Giáo trình
Turbo C nâng cao và C++

PHẦN 1 : TURBO C NÂNG CAO VÀ C++

CHƯƠNG 1 : BIẾN CON TRỎ

§1. KHÁI NIỆM CHUNG

Một con trỏ là một biến chứa địa chỉ của một biến khác. Nếu một biến chứa địa chỉ của một biến khác tthì ta nói biến thứ nhất trỏ đến biến thứ hai .

Cũng như mọi biến khác, biến con trỏ cũng phải được khai báo trước khi dùng. Dạng tổng quát để khai báo một biến con trỏ là :

type *<tên biến>

Trong đó : type là bất kì kiểu dữ liệu cơ bản thích hợp nào được chấp nhận trong C và <tên biến> là tên của một biến con trỏ. Kiểu dữ liệu cơ bản xác định kiểu của những biến mà con trỏ có thể chỉ đến. Ví dụ khai báo biến con trỏ chỉ đến các biến nguyên và biến kiểu kí tự:

```
char *p;
int  *x,*y;
```

Con trỏ có một trị đặc biệt gọi là NULL. Trị này có nghĩa là con trỏ chưa trỏ tới một địa chỉ hợp lệ nào cả. Để dùng được trị này chúng ta phải dùng #include <stdio.h> đầu chương trình

§2. CÁC PHÉP TOÁN VỀ CON TRỎ

C có hai phép toán đặc biệt đối với con trỏ : * và & . Phép toán & là phép toán trả về địa chỉ trong bộ nhớ của biến sau nó. Ví dụ :

```
p = &a;
```

sẽ đặt vào biến p địa chỉ trong bộ nhớ của biến a. Địa chỉ này không có liên quan gì đến trị số của biến a. Nói cách khác địa chỉ của biến a không liên quan gì đến nội dung của biến a.

Phép toán * là phép toán trả về trị của biến đặt tại địa chỉ được mô tả bởi biến đi sau nó. Ví dụ nếu biến a chứa địa chỉ của biến b thì

```
p = *a
```

sẽ đặt trị số của biến b vào biến p

Chương trình 1-1 : Lập chương trình in số 100 lên màn hình

```
main()
  {
    int *p,a,b;
    clrscr();
    a=100;
    p=&a;
    b=*p;
    printf("%d",b);
    getch();
  }
```

§3. TẦM QUAN TRỌNG CỦA DỮ LIỆU KHI KHAI BÁO CON TRỎ

Cần phải bảo đảm là con trỏ luôn luôn trỏ đến một kiểu dữ liệu phù hợp. Ví dụ khi khai báo con trỏ kiểu int , trình biên dịch sẽ hiểu là con trỏ bao giờ cũng chỉ đến một biến có độ dài là 2 byte .
Ta xét một chương trình như sau

Chương trình 1-2

```
main()
  {
    float x=10.1,y;
    int *p;

    clrscr();
    p=&x;
    y=*p;
    printf("%f",y);
    getch();
  }
```

Chương trình này nhằm gán trị của x cho biến y và in ra trị đó. Khi biên dịch chương trình không báo lỗi mà chỉ nhắc nhở :

Suspencious pointer conversion in function main

Tuy nhiên chương trình không gán trị x cho y được. Lí do là ta khai báo một con trỏ int và cho nó trỏ tới biến float x. Như vậy trình biên dịch sẽ chỉ chuyển 2 byte thông tin cho y chứ không phải 4 byte để tạo ra một số dạng float .

§4. CÁC BIỂU THỨC CON TRỎ

1. Các phép gán con trỏ : Cũng giống như bất kì một biến nào khác , ta có thể dùng một con trỏ ở về phải của một phép gán để gán trị của một con trỏ cho một con trỏ khác. Ví dụ ta viết

Chương trình 1-3 :

```
main()
   {
      int x;
      int *p1,*p2;
      clrscr();
      p1 = &x;
      p2 = p1;
      printf(" %p",p2);
      getch();
   }
```

Chương trình này hiện lên địa chỉ của biến x ở dạng hex bằng cách dùng một mã định dạng khác của hàm printf() . %p mô tả rằng sẽ hiện lên một trị chứa trong một biến con trỏ theo dạng reg:xxxx với reg là tên của một trong các thanh ghi segment của CPU còn xxxx là địa chỉ offset tính từ đầu segment .

2. Các phép toán số học của con trỏ : Trong C , ta chỉ có thể dùng hai phép toán số học tác động lên con trỏ là phép + và - . Để hiểu được cái gì sẽ xảy ra khi thực hiện một phép toán số học lên con trỏ ta giả sử p1 là một con trỏ chỉ đến một số nguyên có địa chỉ là 2000 . Sau khi thực hiện biểu thức

p1++ ;

con trỏ sẽ chỉ đến số nguyên nằm ở địa chỉ 2002 vì mỗi khi tăng con trỏ lên 1 nó sẽ chỉ đến số nguyên kế tiếp mà mỗi số nguyên lại có độ dài 2 byte . Điều này cũng đúng khi giảm . Ví dụ :

p1-- ;

sẽ trỏ tới số nguyên ở địa chỉ 1998 . Như vậy mỗi khi con trỏ tăng lên 1 , nó sẽ chỉ đến dữ liệu kế tiếp tại địa chỉ nào đó tuỳ theo độ dài của kiểu dữ liệu. C còn cho phép cộng hay trừ một số nguyên với một con trỏ . Biểu thức :

p1 = p1 + 9;

sẽ làm cho con trỏ chỉ tới phần tử thứ 9 có kiểu là kiểu mà p1 trỏ tới và nằm sau phân tử hiện thời nó đang trỏ đến . Ngoài các phép toán trên , con trỏ không chấp nhận một phép toán nào khác .

3. So sánh các con trỏ : Chúng ta có thể so sánh 2 con trỏ trong một biểu thức quan hệ . Ví dụ cho hai p và q , phát biểu sau đây là hợp lệ :

```
if (p<q)
   printf("p tro den mot vi tri bo nho thap hon q\n");
```

Tuy nhiên cần nhớ rằng phép toán trên là so sánh hai địa chỉ chứa trong p và q chứ không phải nội dung của hai biến mà p và q trỏ tới .

4. Các ví dụ về việc dùng con trỏ :

Chương trình 1-4 : Phân tích chương trình sau :

```
main()
 {
  int i,j,*p;

  i=5;
  p=&i;
  j=*p;
  *p=j+2;
 }
```

Trong chương trình trên ta khai báo hai biến nguyên là i và j và một biến con trỏ p trỏ tới một số nguyên . Chương trình sẽ phân phối bộ nhớ cho 3 biến này ví dụ tại các địa chỉ 100 , 102 và 104 vì mỗi số nguyên dài 2 byte và con trỏ mặc nhiên cũng được mã hoá bằng 2 byte .

100		i
102		j
104		p

lệnh i=5 cho trị số của biến i là 5

100	5	i
102		j
104		p

lệnh p= &i làm cho con trỏ chỉ tới biến i nghĩa là con trỏ p chứa địa chỉ của biến i . Bây giờ p chỉ đến biến i .

100	5	i
102		j
104	100	p

lệnh j=*p đặt nội dung của biến do p chỉ tới (biến i) vào biến j nghĩa là gán 5 cho j

100	5	i
102	5	j
104	100	p

Một trong những vấn đề lí thú khi dùng con trỏ là xem nội dung bộ nhớ của máy tính . Chương trình sau đây cho phép ta vào địa chỉ bắt đầu của RAM mà ta muốn khảo sát và sau đó hiện lên nội dung mỗi byte ở dạng số hex . Trong chương trình có từ khoá far dùng để tham khảo đến các vị trí không nằm trong cùng một segment .

Chương trình 1-5 :

```
main()
  {
    unsigned long int start;
    char *p;
    int t;

    clrscr();
    printf("Nhap vao dia chi bat dau ma ban muon xem : ");
    scanf("%lu",&start);
    p = (char far *) start;
    for(t=0;;t++,p++)
      if(!(t%16))
          {
            printf("%2x\n",*p);
            getch();
          }
  }
```

Trong chương trình ta dùng định dạng %x trong hàm printf() để in ra số dạng hex . Dòng p = (char far *) start; dùng biến đổi số nhập vào thành một con trỏ .

§5. CON TRỎ VÀ MẢNG

Trong chương trước chúng ta đã thấy các ví dụ về mảng . Con trỏ thường được dùng khi xử lí mảng . Chúng ta xét chương trình sau :

Chương trình 1-6 :

```
main()
  {
    int a[10],*pa,x;
    a[0]=11;
    a[1]=22;
    a[2]=33;
    a[3]=44;
    clrscr();
    pa=&a[0];
    x=*pa;
    pa++;
    x=*pa;
```

```
  x=*pa+1;
  x=*(pa+1);
  x=*++pa;
  x=++*pa;
  x=*pa++;
 }
```

*int a[10] , *pa , x;* khai báo một bảng gồm 10 phần tử kiểu int , được liệt kê là a[0],a[1],..,a[9] , một con trỏ để chỉ đến một biến kiểu int và một biến kiểu int là x.
a[0] = 11. . .; từ a[4] đến a[9] chưa được khởi gán . Như vậy chúng sẽ chứa trị ngẫu nhiên đã có tại những vị trí bộ nhớ đã phân phối cho chúng .
pa=&a[0]; đặt vào pa địa chỉ của phần tử đầu tiên của mảng . Biểu thức này có thể viết đơn giản là pa = a ; vì tên của một mảng luôn luôn được trình biên dịch coi là địa chỉ của phần tử đầu tiên của mảng . Tên của mảng không có chỉ số kèm theo có thể được dùng trong chương trình như một hằng địa chỉ .
*x=*pa;* đặt nội dung của biến nguyên mà pa trỏ đến vào (tức là a[0]) vào x . Như vậy x = 11
pa++; pa được tăng lên 1 và bây giờ trỏ vào phần tử thứ 2 của mảng tức là chứa địa chỉ của phần tử a[1]
*x=*pa* ; pa trỏ đến phần tử a[1] nên x = 22
*x = *pa +1* ; x =23
*x = *(pa+1)* ; trước hết pa+1 được thực hiện , nghĩa là pa trỏ vào a[2] , sau đó nội dung của a[2] được gán cho x nên x= 33 .Tuy pa tham gia vào phép toán nhưng trị số của nó không thay đổi .
*x = *++pa*; ++ được thực hiện trước nên pa trỏ tới a[2] . Sau đó trị của a[2] được gán cho x nên x =33
*x= ++*pa*; *pa được thực hiện trước . Do pa chỉ đến a[2] nên *pa=33 và ++*pa=34 . Như vậy x = 34 và a[2]=34
*x=*pa++*; nội dung của pa (tức 34) được đặt vào x . Sau đó nó được tăng lên 1 nên chỉ vào a[3].

Chương trình 1-7:

```
main()
 {
  static int num[]={92,81,70,69,58};
  int dex;
  clrscr();
  for(dex=0;dex<5;dex++)
    printf("%d\n",num[dex]);
  getch();
 }
```

Chương trình 1-8 :

```
main()
 {
  static int num[]={92,81,70,69,58};
  int dex;
  clrscr();
  for(dex=0;dex<5;dex++)
    printf("%d\n",*(num+dex));
```

```
  getch();
 }
```

Hai chương trình chỉ khác nhau ở biểu thức : *(num+dex) . Cách viết này tương đương với num[dex] .Nói cách khác truy cập đến phần tử có chỉ số dex trong mảng num . Chúng ta hiểu *(num+dex) như sau : đầu tiên num là địa chỉ của phần tử đầu tiên của mảng num và ta muốn biết trị số của phần tử có chỉ số dex . Vì vậy num+dex sẽ là địa chỉ của phần tử thứ dex . *(num+dex) xác định nội dung của phần tử (num+dex) . Tóm lại :

(array+index) tương tự array(index)

Có hai cách truy cập mảng là :

theo kí hiệu mảng &array[index]
theo kí hiệu con trỏ array+index

Chương trình 1-9 : Tính nhiệt độ trung bình bằng cách dùng con trỏ

```
main()
 {
  float temp[40];
  float sum=0.0;
  int num,day=0;
  clrscr();
  do
    {
       printf("Cho nhiet do ngay thu %d: ",day+1);
       scanf("%f",temp+day);
    }
  while(*(temp+day++)>0);
  num = day-1;
  for(day=0;day<num;day++)
    sum+=*(temp+day);
  printf("Nhiet do trung binh la : %.3f",sum/num);
  getch();
 }
```

Trong ví dụ trên chúng ta đã dùng biểu thức (temp+day) để truy cập mảng . Tuy nhiên viết while((*temp++)>0) vì temp là hằng con trỏ chứ không phải biến con trỏ . Như vậy chỉ được phép thay đổi trị của biến con trỏ chứ không được thay đổi trị của hằng con trỏ . Chúng ta viết lại chương trình như sau :

Chương trình 1-10 :

```
main()
 {
  float temp[40];
  float sum=0.0;
  int num,day=0;
  float *p;

  clrscr();
  p=temp;
  do
    {
       printf("Cho nhiet do ngay thu %d: ",day+1);
```

```
        scanf("%f",p);
        day++;
     }
   while(*(p++)>0);
   p=temp;
   num=day-1;
   for(day=0;day<num;day++)
     sum+=*(p++);
   printf("Nhiet do trung binh la : %.3f",sum/num);
   getch();
  }
```

Trong chương trình này địa chỉ của temp được đưa vào biến con trỏ p . Sau đó ta tham khảo tới p giống như temp . Ta dùng p trỏ tới mảng và *p là nội dung của địa chỉ đó . Hơn nã do p là biến con trỏ nên ta có thể tăng nó bằng phát biểu p++.

§6. CON TRỎ VÀ CHUỖI

Rất nhiều hàm thư viện trong C làm việc với chuỗi theo con trỏ . Ví dụ hàm strchr() trả về con trỏ trỏ đến lần xuất hiện đầu tiên của một kí tự nào đó trong chuỗi Ví dụ : ptr = strchr(str,'x')
thì biến con trỏ ptr sẽ được gán địa chỉ của lần xuất hiện kí tự 'x' đầu tiên trong chuỗi str .
Sau đây là chương trình cho phép ta gõ vào một câu và một kí tự cần định vị trong câu . Chương trình sẽ cho ta :

- địa chỉ bắt đầu của chuỗi
- địa chỉ của kí tự cần định vị
- độ lệch so với điểm đầu chuỗi

Chương trình 1-11 :

```
#include<string.h>
main()
  {
   char ch,line[81],*ptr;
   clrscr();
   printf("Cho mot cau : ");
   gets(line);
   printf("Cho ki tu can tim : ");
   ch=getche();
   ptr=strchr(line,ch);
   printf("\nChuoi bat dau tai dia chi %u.\n",line);
   printf("Ki tu xuat hien lan dau tai %u.\n",ptr);
   printf("Do la vi tri %d",(ptr-line+1));
   getch();
  }
```

Chuỗi cũng có thể được khởi tạo bằng con trỏ . Ta xét ví dụ sau

Chương trình 1-11 :

```
main()
  {
   char *chao="Xin chao !";
```

```
   char ten[30];

   clrscr();
   printf("Cho ten cua ban : ");
   gets(ten);
   printf(chao);
   puts(ten);
   getch();
  }
```

Trong chương trình trên ta đã khởi tạo chuỗi bằng phát biểu

char *chao = " Xin chao !"

thay cho

static char chao[]=" Xin chao !"

Cả hai cách đều cho cùng một kết quả . Trong phương án dùng con trỏ , chao là biến con trỏ nên có thể thay đổi được . Ví dụ phát biểu :

puts(++chao)

sẽ cho kết quả : in chao !

Nếu ta có một mảng chuỗi ta cũng có thể dùng mảng con trỏ trỏ tới mảng chuỗi này . Ta khởi tạo chúng giống như khởi tạo biến con trỏ đơn .

Chương trình 1-12 :

```
#define         max    5
main()
  {
   int dex;
   int enter=0;
   char name[40];
   static char *list[max]=
         {
           "Hung",
           "Ngan",
           "Van",
           "Hoa",
           "Tien"
         };
   clrscr();
   printf("Cho ten cua ban : ");
   gets(name);
   for(dex=0;dex<max;dex++)
     if (strcmp(list[dex],name)==0)
         enter=1;
   if (enter==1)
     printf("Ban da dang ki hoc lop C");
   else
     printf("Ban chua dang ki vao lop");
   getch();
  }
```

Phát biểu char *list[max] nói rằng list là một mảng con trỏ gồm max phần tử chỉ tới các kí tự . Chúng ta xét tiếp một ví dụ như sau :

Chương trình 1-13 : Nhập vào một dãy tên và sắp xếp lại đúng thứ tự a,b,c

```
#define   maxnum   38
#define   maxlen   81
main()
  {
    static char name[maxnum][maxlen];
    char *ptr[maxnum];
    char *temp;
    int count = 0;
    int in,out;

    clrscr();
    while (count<maxnum)
      {
          printf("Ban cho ten : ");
          gets(name[count]);
          if (strlen(name[count])==0)
            break;
          ptr[count++]=name[count];
      }
    for (out=0;out<count-1;out++)
      for (in=out+1;in<count;in++)
          if (strcmp(ptr[out],ptr[in])>0)
            {
              temp=ptr[in];
              ptr[in]=ptr[out];
              ptr[out]=temp;
            }
    printf("Danh sach da sap xep :\n");
    for(out=0;out<count;out++)
      printf("Ten thu %d : %s\n",out+1,ptr[out]);
    getch();
  }
```

Chương trình này dùng cả mảng chuỗi và mảng con trỏ chuỗi . Con trỏ nằm trong mảng được khai báo như sau :

char *ptr[maxnum]

chuỗi nằm trong mảng hai chiều

static char name[maxnum][maxlen]

Do ta không biết một chuỗi dài bao nhiêu nên phải dùng mảng chuỗi name có tối đa maxnum phần tử , mỗi phần tử có maxlen kí tự . Khi nhập chuỗi phát biểu

ptr[count++] = name[count

sẽ gán địa chỉ của mỗi chuỗi được cất giữ trong mảng name[][] vào phần tử con trỏ ptr . Sau đó mảng con trỏ này được sắp xếp dựa trên mảng name[][] nhương mảng name[][] không thay đổi gì cả .

Ngôn ngữ C có thể xử lí các thành phần của mảng như một mảng . Cụ thể C có thể xem một dòng của mảng hai chiều như là một mảng một chiều. điều này rất tiện lợi như ta đã thấy trong chương trình trên . Câu lệnh ptr[count++] = name[count hoàn toàn hợp lí vì vế

phải chính là địa chỉ của mảng name[count] và mảng này là một thành phần của mảng name[][] là một mảng hai chiều . Ta xem lại khai báo :

static char name[maxnum][maxlen]

rõ ràng ta có thể xem đây là một mảng một chiều có maxnum chuỗi và tham khảo tới phần tử của mảng một chiều bằng 1 chỉ số . Ví dụ :

name[count] với count<=maxnum như thế

name[0] : địa chỉ của chuỗi 1

name[1] : địa chỉ của chuỗi 2

§7. CON TRỎ TRỎ ĐẾN CON TRỎ

Chúng ta có một chương trình in ra một bảng số được viết như sau :

Chương trình 1-14:

```
#define row 4
#define col 5
main()
  {
   static int table[row][col]={
                                {13,15,17,19,21},
                                {20,22,24,26,28},
                                {31,33,35,37,39},
                                {40,42,44,46,48}
                              };
   int c=10;
   int i,j;
   clrscr();
   for(i=0;i<row;i++)
     for(j=0;j<col;j++)
         table[i][j]+=c;
   for(i=0;i<row;i++)
     {
         for(j=0;j<col;j++)
           printf("%5d",table[i][j]);
         printf("\n");
     }
   getch();
  }
```

Trong chương trình trên ta dùng kí hiệu mảng. Bây giờ ta muốn viết chương trình dùng kí hiệu con trỏ thay cho kí hiệu mảng. Vậy thì làm thế nào để mô tả table[i][j] bằng con trỏ . Ta thấy rằng :

- table là địa chỉ của phần tử đầu tiên của toàn bộ mảng , giả định là 1000
- do đây là mảng nguyên nên mỗi phần tử chiếm 2 byte và mỗi dòng chiếm 10 byte vì có 5 phần tử . Như vậy địa chỉ của hai dòng liền nhau cách nhau 10 byte
- do có thể xem mỗi dòng là một mảng một chiều nên các mảng một chiều liền nhau cách nhau 10 byte
- trình biên dịch biết số cột trong mảng qua khai báo nên nó sẽ hiểu table+1 là đem table (trị 1000) cộng với 10 byte thành 1010 . Tương tự table+2 cho ta 1020 .

1000	13	15	17	19	21	table[0]

table==1000	1010	20	22	24	26	28	table[1]
	1020	31	33	35	37	39	table[2]
	1030	40	42	44	46	48	table[3]

Để tham khảo đến từng phần tử của dòng trước hết ta lưu ý địa chỉ của mảng cũng là địa chỉ của phần tử đầu tiên của mảng . Ví dụ với mảng một chiều a[size] thì a và a[0] là như nhau . Trở lại mảng hai chiều địa chỉ của mảng một chiều tạo bởi dòng thứ 3 của mảng table[][] là table[2] hay table+2 .Trong kí hiệu con trỏ địa chỉ của phần tử đầu tiên của mảng một chiều này là &table[2][0] hay *(table+2) . Cả hai cách viết table+2 và *(table+2) đều tham khảo nội dung của cùng một ô nhớ (1020) . Nếu cộng 1 vào table +3 để có table+3 thì ta nhận được địa chỉ của dòng thứ 4 trong mảng table[][] . Nếu cộng 1 vào *(table+2) để có *(table+2)+1 thì có địa chỉ của phần tử thứ 2 trong dòng thứ 3 của mảng table[][] . Tóm lại :

table[i] = *(table+i)
&table[i] = table+i
table[i][j] = *(*table+i)+j)
&table[i][j] = (*(table+i)+j)

Như vậy chương trình trên được viết lại như sau :

Chương trình 1-15 :

```
#define row 4
#define col 5
main()
  {
   static int table[row][col]={
                                {13,15,17,19,21},
                                {20,22,24,26,28},
                                {31,33,35,37,39},
                                {40,42,44,46,48}
                              };
   int c=10;
   int i,j;
   clrscr();
   for(i=0;i<row;i++)
     for(j=0;j<col;j++)
         *(*(table+i)+j)+=c;
   for(i=0;i<row;i++)
     {
         for(j=0;j<col;j++)
           printf("%5d",*(*(table+i)+j));
         printf("\n");
     }
   getch();
  }
```

Bài tập : Lập chương trình tính hiệu độ dài hai chuỗi nhập vào từ bàn phím
Lập chương trình xác định giá trị cực đại của n số nhập vào từ bàn phím
Lập chương trình quản lí hàng gồm ngày , lượng nhập ,lượng xuất và hàng tồn kho

CHƯƠNG 2 : BÀN PHÍM VÀ CURSOR

§1. CÁC MÃ PHÍM MỞ RỘNG

Chúng ta đã thấy bàn phím tạo các mã thông thường cho các chữ cái, các số và dấu chấm câu. Các phím này đều tạo mã ASCII dài 1 byte. Tuy nhiên có nhều phím và tổ hợp phím không được biểu diễn bằng bộ kí tự dài một byte này ví dụ như các phím chức năng từ F1 đến F10 hay các phím điều khiển cursor . Các phím này được mô tả bằng một mã dài 2 byte. Byte đầu tiên có trị số là 0 và byte thứ hai là trị số mã của phím này .

1. Nhận biết các mã mở rộng : Một mã mở rộng phải có 2 byte và byte đầu tiên là 0 nên chương trình cần phải đọc 2 byte này . Sau đây là đoạn chương trình nhận biết các mã mở rộng ***Chương trình 2-1***:

```
#include <string.h>
main()
  {
    char key,key1;
    clrscr();
    while ((key=getche())!='x')
      if (key==0)
        {
           key1=getch();
         printf("%3d%3d",key,key1);
        }
      else
        printf("%3d",key);
   }
```

Chương trình này sẽ hiện thị các mã của các phím được gõ cho dù chúng là mã một byte hay 2 byte . Ta dùng hàm getch() để không hiển thị kí tự vừa gõ lên màn hình . Trong biểu thức kiểm tra của while chương trình đọc mã đầu tiên . Nếu mã này là 0 , chương trình biết đó là mã mở rộng và đọc tiếp phần thứ hai của mã bằng hàm getch() . Sau đó nó hiển thị cả hai phần . Nếu phần đầu khác không chương trình sẽ cho rằng đây không phải là mã mở rộng và hiện thị mã này .

2. Đoán nhận mã mở rộng : Một cách đoán nhận mã mở rộng là dùng phát biểu switch như trong chương trình sau :

Chương trình 2-2 :

```
main()
  {
    int key,key1;
    clrscr();
    while ((key=getche())!='X')
      if (key==0)
        {
         key1=getch();
         switch (key1)
           {
              case 59 : printf("Phim F1 duoc nhan\n");
                    break;
              case 60 : printf("Phim F2 duoc nhan\n");
```

```
                break;
            case 75 : printf("Phim left arrow duoc nhan\n");
                break;
            default : printf("Phim mo rong khac duoc nhan\n");
                break;
          }
       }
     else
       printf("%3d",key);
   getch();
  }
```

§2. ĐIỀU KHIỂN CURSOR VÀ ANSI.SYS

1.Khái niệm chung :Tập tin ansi.sys cung cấp tập đã chuẩn hoá các mã điều khiển cursor . ANSI - America National Standards Institut. Để bảo đảm sự cài đặt của tập tin ansi.sys trong tập tin config.sys ta đặt dòng lệnh :

device = ansi.sys

2. Điều khiển cursor bằng ansi.sys : ansi.sys dùng dãy escape để điều khiển con nháy . Chuỗi escape gồm nhiều kí tự đặc biệt . Ansi.sys tìm chuỗi escape này qua thành phần của chuỗi trong hàm prinft() và giải mã các lệnh theo sau nó . Chuỗi escape luôn luôn giống nhau , gồm kí tự không in được "\x1B"(là mã của kí tự escape) sau đó là dấu [. Sau chuỗi escape có thể có một hay nhiều kí tự . Nhờ chuỗi này con nháy có thể đi lên , xuóng , sang trái , phải hay định vị tại một vị trí nào đó . Ví dụ để di chuyển con nháy xuống dưới ta dùng chuỗi "\x1B[B"

Chương trình 2-3 : Viết chương trình in một chuỗi theo đường chéo :

```
main()
  {
    clrscr();
    printf("Cho mot chuoi tan cung bang dau .:");
    while (getche()!='.')
      printf("\x1B[B");
    getch();
   }
```

3. Dùng #define và chuỗi escape : Chuỗi "\x1B[B" được mã hoá và rất khó đọc . Khi dùng các chương trình phức tạp nên ghi chú rõ ràng bằng cách dùng dẫn hướng #define .

Chương trình 2-4 :

```
#define c_down "\x1B[B"
main()
  {
    while (getche()!='.')
      printf(c_down);
    getch();
   }
```

Tóm tắt các lệnh điều khiển con nháy

Mã	Công dụng

“[2J”	Xoá màn hình và đưa con nháy về home
“[K”	Xoá đến cuối dòng
“[A”	Đưa con nháy lên một dòng
“[B”	Đưa con nháy xuống một dòng
“[C”	Đưa con nháy sang phải một cột
“[D”	Đưa con nháy sang trái một cột
“[%d;%df	Đưa con nháy đến vị trí nào đó
“[s”	Cất giữ vị trí con nháy
“[u”	Khôi phục vị trí con nháy
“[%dA”	Đưa con nháy lên một số dòng
“[%dB”	Đưa con nháy xuống một số dòng
“[%dC”	Đưa con nháy sang phải một số cột
“[%dD”	Đưa con nháy sang trái một dòng và nhiều cột

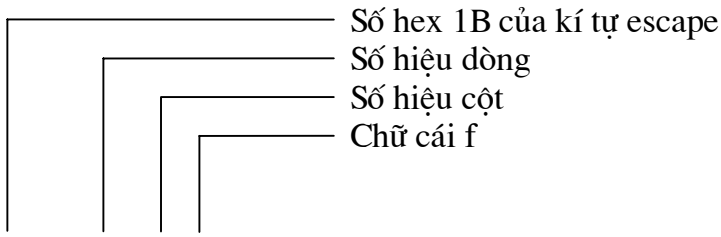
4. Điều khiển con nháy từ bàn phím : Sau đây là chương trình cho phép bạn vẽ các hình đơn giản trên màn hình

Chương trình 2-5 :

```
#define clear     "\x1B[2J"
#define c_left    "\x1B[D"
#define c_right   "\x1B[C"
#define c_up      "\x1B[A"
#define c_down    "\x1B[B"
#define l_arrow   75
#define r_arrow   77
#define u_arrow   72
#define d_arrow   80
#define across    205
#define updown    186
main()
  {
    int key;
    printf(clear);
    while ((key=getch())==0)
      {
        key=getche();
        switch (key)
          {
            case l_arrow : printf(c_left);
                           putch(across);
                           break;
            case r_arrow : printf(c_right);
                           putch(across);
                           break;
            case u_arrow : printf(c_up);
                           putch(updown);
                           break;
            case d_arrow : printf(c_down);
                           putch(updown);
                           break;
          }
```

```
        printf(c_left);
      }
    getch();
 }
```

5. Đưa con nháy đến vị trí bất kì : Chuỗi escape dạng sau sẽ đưa con nháy đến vị trí bất kì trên màn hình



Sau đây là một chương trình ví dụ về cách dùng chuỗi đó

Chương trình 2-6 :

```
#define true    1
#define clear   "\x1B[2J"
#define erase   "\x1B[K"
main()
  {
    int row=1,col=1;
    printf(clear);
    while(true)
      {
       printf("\x1B[23;1f");
       printf(erase);
       printf("Nhap vao so dong va so cot dang(20,40)");
       scanf("%d%d",&row,&col);
       printf("\x1B[%d;%df",row,col);
          printf("*(%d,%d)",row,col);
      }
  }
```

§6. TRÌNH BÀY CHỖ BẤT KÌ TRÊN MÀN HÌNH

Sau đây là chương trình dùng chuỗi định vị cursor .Chương trình cung cấp hai menu định vị dọc theo màn hình .

Chương trình 2-7 :

```
#define size1 5
#define size2 4
#define clear "\x1B[2J"
main()
  {
    static char *menu1[]=
                          {
                           "Open",
                           "Close"
                           "Save"
                           "Print"
                           "Quit"
```

```
                        };
   static char *menu2[]=
                        {
                         "Cut",
                         "Copy",
                         "Paste",
                         "Reformat"
                        };
   void display(char *[],int ,int);

   printf(clear);
   display(menu1,size1,20);
   display(menu2,size2,20);
   getch();
  }
void display(char *arr[],int size,int hpos)
  {
   int j;
   for (j=0;j<size;j++)
     {
        printf("\x1B[%d",j+1,hpos);
        printf("%s\n",*(arr+j));
     }
  }
```

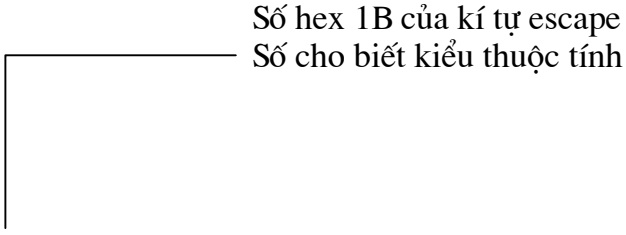
Các mục cho từng menu được cất giữ trong mảng các con trỏ trỏ tới chuỗi . Sau đó chương trình dùng hàm để hiển thị menu . Hàm định vị con nháy nhờ dãy định vị ANSI.SYS , lấy số hiệu dòng từ số hiệu của mục trên menu và số hiệu cột được chương trình chính truyền sang .

§7. CÁC THUỘC TÍNH CỦA KÍ TỰ

Mỗi kí tự hiển thị trên màn hình được cất giữ trong hai byte bộ nhớ . Một byte là mã thông thường của kí tự và byte kia là thuộc tính của nó . Byte thuộc tính ấn định diện mạo của kí tự như chớp nháy , đậm , gạch dưới , đảo màu . Ta có thể dùng chuỗi escape của ANSI để ấn định thuộc tính của kí tự . Theo sau chuỗi kí tự escape và ngoặc vuông là con số và chữ m . Sau đây là danh sách các số tạo hiệu ứng trên màn hình :

2,3,6	màu tối
0	tắt thuộc tính , thường là màu trắng trên nền đen
1	đậm
4	gạch dưới
5	chớp nháy
7	đảo màu
8	không thấy được

Chuỗi escape có dạng như sau :



“ \ x 1 B [10 m”
Chuỗi này được gởi trong tiến trình hiển thị . Mỗi khi bật một thuộc tính , tất cả các kí tự sẽ hiển thị theo thuộc tính mới cho đến khi nó tắt đi . Sau đây là chương trình biểu diễn các thuộc tính của kí tự

Chương trình 2-8 :

```
#define NORMAL   "\x1B[Om"
#define BOLD     "\x1B[1m"
#define UNDER    "\x1B[4m"
#define BLINK    "\x1B[5m"
#define REVERSE  "\x1B[7m"
main()
  {
    printf("normal%s blink %s normal \n\n",BLINK,NORMAL);
    printf("normal%s bold %s normal \n\n",BOLD,NORMAL);
    printf("normal%s underline %s normal \n\n",UNDER,NORMAL);
    printf("normal%s reversed %s normal \n\n",REVERSE,NORMAL);
    printf("%s%s reversed and blink %s \n\n",BLINK,REVERSE,NORMAL);
  }
```

§8. MENU

Ta xây dựng một chương trình gồm 5 mục menu là Open ,Close,Save,Print,Quit . Các phím mũi tên lên xuống sẽ di chuyển vệt sáng đến các mục cần chọn.Phím INS để chọn và thực hiện công việc tương ứng . Mục Quit sẽ kết thúc chương trình .

Chương trình 2-9 :

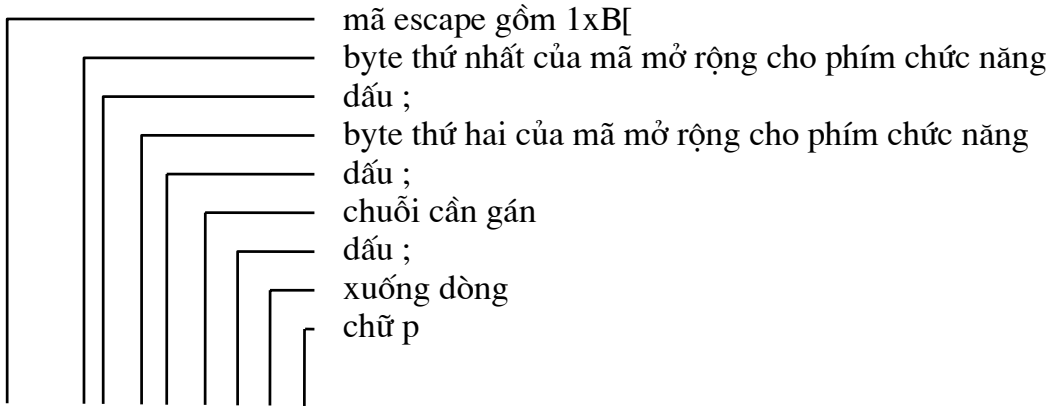
```
#define true   1
#define num  5
#define clear  "\x1B[2J"
#define erase "\x1B[K"
#define normal      "\x1B[Om"
#define      reverse       "\x1B[7m"
#define home"\x1B[1;1f"
#define bottom      "\x1B[20:1f"
#define      u_arro 72
#define color "\x1B[4m"
/*#define l_arro      75
#define r_arro      77*/
#define d_arro 80
#define insert83
main()
  {
    static char *item[num]=
                          {
                           "Open",
                           "Close",
                           "Save",
                           "Print",
                           "Quit"
                          };
```

```
    int curpos;
    int code;
    void display(char *[],int,int);
    int getcode(void);
    void action(int);
    printf(clear);
    curpos=0;
    while(true)
      {
          display(item,num,curpos);
          code=getcode();
          switch (code)
            {
             case u_arro:if (curpos>0)
                              --curpos;
                              break;
             case d_arro:if (curpos<num-1)
                              ++curpos;
                              break;
             case insert:action(curpos);
                         break;
            }
      }
  }

void display(char *arr[],int size,int pos)
  {
    int j;
    printf(home);
    for (j=0;j<size;j++)
      {
          if (j==pos)
            printf(reverse);
          printf("%s\n",*(arr+1));
          printf("%s%5s",color,*(arr+j));
          printf(normal);
          printf("%s"," ");
          printf(home);
      }
  }

int getcode()
  {
    int key;
    while(getch()!=0)
      ;
    return (getch());
  }

void action(int pos)
```

```
{
  switch(pos)
    {
        case 0: printf("Open");
                break;
        case 1: printf("Close");
                break;
        case 2: printf("Save");
                break;
        case 3: printf("Print");
                break;
        case 4: exit();
    }
}
```

§9. GÁN PHÍM CHỨC NĂNG BẰNG ANSI.SYS

Nhờ gán chuỗi vào phím chức năng ta có thể cấu hình lại bàn phím đamg dùng . Dạng thức của chuỗi gán phím chức năng như sau :



\ x 1 B [0 ; 68 ; “s” ; 13 p

Chương trình 2-10:

```
main()
  {
    char str[81];
    int key;
    clrscr();
    printf("Nhap vao mot so cua phim chuc nang :");
    gets(str);
    key=atoi(str);
    printf("Nhap vao mot chuoi de gan phim nay : ");
    gets(str);
    printf("\x1B[0;%d;\"%s\";13p",key+58,str);
  }
```

CHƯƠNG 3 : NHẬP VÀ XUẤT DỮ LIỆU

§1. KHÁI NIỆM CHUNG

1. Khái niệm :Trước đây chúng ta đã xét việc nhập dữ liệu từ bàn phím. Trong nhiều trường hợp thực tế , để thuận lợi , chúng ta phải nhập dữ liệu từ các tập tin trên đĩa . Các hàm thư viện của C cho phép truy cập tập tin và chia là 2 cấp khác nhau :

- các hàm cấp 1 là các hàm ở cấp thấp nhất , truy cập trực tiếp đến các tập tin trên đĩa.C không cung cấp vùng nhớ đệm cho các hàm này
- các hàm cấp 2 là các hàm truy xuất tập tin cao hơn , do chúng được C cung cấp vùng nhớ đệm

Đối với các hàm cấp 1 , tập tin được xem là khối các byte liên tục do đó khi muốn truy cập mẫu tin cụ thể thì phải tính toán địa chỉ của mẫu tin và như vậy công việc vất vả hơn . Ngoài ra phải cung cấp vùng nhớ đệm cho kiểu đọc ghi này. Đối với các hàm cấp hai công việc nhẹ nhàng hơn do :

- trình biên dịch tự động cung cấp vùng kí ức đệm cho chúng
- có thể truy xuất các mẫu tin mà không gặp khó khăn như với các hàm cấp 1

Trong C , các thông tin cần thiết cho các hàm xuất nhập cấp 2 được đặt trong tập tin stdio.h còn các thông tin về hàm nhập xuất cấp 1 thì ở trong tập tin io.h

2. Stream và các tập tin : Ta phải phân biệt hai thuật ngữ là stream và file .Hệ thống xuất nhập của C cung cấp một không gian tưởng tượng giữa người lập trình và các thiết bị được dùng . Cấp trung gian tưởng tượng này gọi là stream và thiết bị cụ thể là tập tin .

a. Các streams : Trong máy tính ta dùng 2 loại stream : văn bản và nhị phân . Một stream văn bản là một loạt kí tự được tổ chức thành dòng mà mỗi dòng được kết thúc bằng kí tự xuống dòng newline("\n") . Khi ghi , một kí tự chuyển dòng LF(mã 10) đucợ chuyển thành 2 kí tự CR(mã 13) và LF . Khi đọc 2 kí tự liên tiếp CR và LF trên tập tin chỉ cho ta một kí tự LF .

Một stream nhị phân là một loạt các byte .

a. Các tập tin : Trong C ,một tập tin là một khái niệm logic mà hệ thống có thể áp dụng cho mọi thứ từ các tập tin trên đĩa cho đến các terminal . Khi bắt đầu thực hiện chương trình , máy tính mở 3 stream văn bản đã được định nghĩa trước là stdin , stdout và stderr . Đối với hầu hết các hệ thống , các thiết bị này là console

§2. NHẬP XUẤT CHUẨN

1. Nhập xuất kí tự , chuỗi kí tự , định dạng và bản ghi : Nhập xuất cấp 2(nhập xuất chuẩn) cung cấp 4 cách đọc và ghi dữ liệu khác nhau (ngược lại nhập xuất cập1 chỉ dùng 1 trong 4 cách này) .

Trước hết dữ liệu có thể đọc ghi mỗi lần một kí tự , tương tự như cách làm việc của putchar() và getche() để đọc dữ liệu từ bàn phím và hiển thị lên màn hình .

Thứ hai , dữ liệu có thể nhập xuất theo chuỗi bằng các dùng các hàm gets() và puts()

Thứ ba , dữ liệu có thể được nhập và xuất theo khuôn dạng bằng các hàm fprintf() và fscanf()

Thứ tư , dữ liệu được đọc và ghi theo khối có chiều dài cố định thường dùng lưu trữ mảng hay cấu trúc bằng các hàm fread() và fwrite() . Tóm lại :

Các hàm dùng chung cho hai kiểu nhị phân và văn bản

fopen : dùng mở tập tin

fclose : đóng tập tin
fclose : đóng tất cả các tập tin
fflush : dùng làm sạch vùng đệm của tập tin
flushall : dùng làm sạch vùng đệm của tất cả tập tin
ferror : cho biết có lỗi (khác không) hay không có lỗi (bằng 0)
perror : thong báo lỗi trên màn hình
foef : cho biết cuối tập tin hay chưa
unlink và remove : dùng để loại tập tin trên đĩa
fseek : di chuyển con trỏ đến vị trí bất kì trên tập tin
ftell : cho biết vị trí hiện tại của con trỏ

Các hàm nhập xuất kí tự

putc và fputc : nhập kí tự vào tập tin
getc và fgetc : đọc kí tự từ tập tin
fprintf : dùng ghi dữ liệu định dạng lên tập tin
fscanf : dùng đọc dữ liệu định dạng từ tập tin
fputs : dùng ghi chuỗi lên tập tin
fgets : dùng đọc chuỗi từ tập tin

Các hàm dùng cho kiểu xuất nhập nhị phân

putw : dùng ghi một số nguyên hai byte lên tập tin
gets : dùng đọc một số nguyên hai byte từ tập tin
fwrite : dùng ghi một mẫu tin lên tập tin
fread : dùng đọc một mẫu tin từ tập tin

2.Dạng văn bản và dạng nhị phân : Cách khác để phân loại các thao tác nhập xuất tập tin là nó được mở theo kiểu văn bản hay nhị phân . Điểm khác biệt giữa hai loại này là kí tự newline và end of line . Điểm thứ hai để phân biệt hai kiểu tập tin là là cách lưu trữ các số vào đĩa . Đối với dạng văn bản thì các số được lưu trữ thành chuỗi các kí tự còn dạng nhị phân thì các số được lưu như trong bộ nhớ , nghĩa là dùng hai byte cho một số nguyên và 4 byte cho một số float .

3. Nhập xuất chuẩn : Chương trình dùng các hàm nhập xuất cấp 2 thường dễ hiểu hơn nên chúng ta sẽ nghiên cứu trước .

a. Nhập xuất kí tự : Để nhập kí tự vào tập tin ta dùng hàm putc() hay fputc().Để đọc kí tự từ tập tin ta dùng hàm getc() hay fgetc() . Chương trình ví dụ này là tạo lập các kí tự bằng cách gõ vào bàn phím mỗi lần một kí tự và ghi vào một tập tin trên đĩa . Chương trình dùng hàm fopen() để mở một tập tin , dùng hàm putc() để ghi lên tập tin , dùng kí tự enter để kết thúc chương trình .

Chương trình 3-1 :

```
#include <stdio.h>
#include <conio.h>
void main()
  {
    FILE *fp;
    char ch;
    printf("Nhap cac ki tu : ");
    fp=fopen("textfile","w");
    while ((ch=getche())!='\r')
      putc(ch,fp);
    fclose(fp);
  }
```

b. Mở một tập tin : Trước khi ghi một tập tin lên đĩa ta phải mở tập tin đó đã . Để mở tập tin , trước hết ta phải khai báo một con trỏ chỉ tới FILE . FILE là một structure chứa đựng các thông tin về cấu trúc của tập tin ví dụ như kích thước , vị trí của bộ đệm dữ liệu hiện hành . Cấu trúc FILE được khai báo trong stdio.h nên ta cần include tập tin này . Ngoài ra stdio.h còn xác định các tên và các biến khác được dùng trong chương trình hướng đến các tập tin . Do vậy trong chương trình ta có câu lệnh :

FILE *fp ;

Sau đó ta mở tập tin bằng lệnh :

fopen("textfile","w");

Khi viết như vậy sẽ làm cho hệ điều hành biết là mở một tập tin tên là textfile trong thư mục hiện hành để viết lên tập tin đó (nhờ "w") . Ta có thể cho tên đường dẫn đầy đủ nếu muốn mở tập tin ở thư mục bất kì . Hàm fopen() trả về một con trỏ chỉ đến cấu trúc FILE cho tập tin và con trỏ này được cất giữ trong biến fp . Chuỗi "w" được gọi là kiểu , nó có nghĩa là ghi lên tập tin . Các kiểu mở tập tin là :

- "r","rt" mở để đọc , tập tin phải có trên đĩa
- "w","wt" mở để ghi , nếu trên đĩa đã có tập tin thì nội dung bị ghi đè , nếu chưa có thì tập tin được tạo lập
- "a","at" mở để nối thêm, thông tin được ghi vào cuối tập tin cũ nếu đã có tập tin hay tạo mới tập tin
- "r+",''r+t" mở để vừa đọc và ghi , tập tin phải có trên đĩa
- "rb" mở một tập tin để đọc theo kiểu nhị phân . Tập tin phải có sẵn trên đĩa
- "r+b" mở một tập tin để đọc theo kiểu nhị phân . Tập tin phải có sẵn trên đĩa
- "w+","w+t" mở để vừa đọc và ghi , nội dung tập tin đã có trên đĩa sẽ bị ghi đè lên
- "wb" mở để ghi theo kiểu nhị phân , nếu trên đĩa đã có tập tin thì nội dung bị ghi đè , nếu chưa có thì tập tin được tạo lập
- "a+","a+t" mở để đọc và nối thêm , nếu tập tin chưa có thì nó sẽ được tạo ra
- "ab" mở để đọc và nối thêm theo kiểu nhị phân , nếu tập tin chưa có thì nó sẽ được tạo ra

c. Ghi lên tập tin : Khi tập tin đã được mở , ta có thể ghi lên tập tin từng kí tự một bằng cách dùng hàm :

putc(ch,fp)

Hàm putc() tương tự các hàm putch() và putchar() . Hàm putc() ghi lên tập tin có cấu trúc FILE được ấn định bởi biến fp nhận được khi mở tập tin . Tiến trình ghi được tiến hành cho đến khi nhấn enter .

d. Đóng tập tin : Khi không đọc ghi nữa ta cần đóng tập tin . Câu lệnh đóng tập tin là :

fclose(fp);

Ta báo cho hệ thống biết là cần đóng tập tin chỉ bởi fp .

e. Đọc tập tin : Nếu ta có thể ghi lên tập tin thì ta cũng có thể đọc từ tập tin . Ta có ví dụ sau :

Chương trình 3-2 :

```
#include <stdio.h>
#include <conio.h>
main()
  {
    FILE *fp;
    int ch;
    clrscr();
    fp=fopen("textfile","r");
    while ((ch=getc(fp))!=EOF)
```

```
    printf("%c",ch);
  fclose(fp);
  getch();
 }
```

f. Kết thúc tập tin : Sự khác nhâu chủ yếu giữa chương trình đọc và ghi là chương trình đọc phải phân biệt được đâu là kí tự EOF . Nó không phải là một kí tự àm là một số nguyên do hệ điều hành gửi tới . Khi hết tập tin ta gặp mã kết thúc tập tin EOF (định nghĩa trong stdio.h bằng -1) và hàm foef() cho trị khác không . Người ta chọn -1 làm mã kết thúc vì nếu chưa gặp cuối tập tin thì sẽ đọc được một byte mà mã sẽ nằm trong khoảng 0-255 . Như vậy giá trị -1 không trùng với bất kì kí tự nào nào được đọc từ tập tin . Trong khi chương trình đang đọc và hiển thị các kí tự thì nó tìm kiếm mộ giá trị -1 hay EOF . Khi thấy giá trị này , chương trình sẽ kết thúc . Chúng ta dùng một biến nguyên cất giữ một kí tự đọc được , do đó ta có thể hiểu dấu EOF như là một trị nguyên có trị là -1 . Nếu dùng một biến kiểu char , chúg ta có thể dùng tất cả các kí tự từ 0..255 - đó là tổ hợp 8 bit . Do đó nếu dùng biến nguyên , ta bảo đảm rằng chỉ có một giá trị 16 bit là -1 , đó là dấu EOF .

g. Sự phiền phức khi mở tập tin : Hai chương trình ta trình bày trên có một lỗi tiểm ẩn . Nếu tập tin đã được chỉ định không mở được thì chương trình không chạy . Lỗi này có thể là do tập tin chưa có (khi đọc) hay đĩa không còn đủ chỗ(khi ghi). Do đó vấn đề là phải kiểm tra xem tập tin có mở được hay không , nếu tập tin không mở được thì hàm fopen() trả về trị 0(0 là NULL trong stdio.h) . Khi này C coi đây không phải là địa chỉ hợp lệ . Như vậy ta viết lại chương trình trên như sau

Chương trình 3-3 :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
void main()
 {
  FILE *fp;
  int ch;
  clrscr();
  if ((fp=fopen("file","r"))==NULL)
    {
       printf("Khong mo duoc tap tin\n");
       getch();
       exit(1);
    }
  while ((ch=getc(fp))!=EOF)
    printf("%c",ch);
  fclose(fp);
 }
```

h. Đếm số kí tự : Khả năng đọc và ghi tập tin trên cơ sở các kí tự cho phép triển khai một số ứng dụng . Chúng ta xem xét chương trình đếm số kí tự sau :

Chương trình 3-4 :

```
#include <stdio.h>
#include <conio.h>
main(int argc,char *argv)
 {
  FILE *fp;
```

```
   char string[8];
   int count = 0;
   clrscr();
   if (argc!=2)
     {
        printf("Format c:\<ten chuong trinh>    <ten tap tin>");
        getch();
        exit(1);
     }
   if ((fp=fopen(argv[1],"r"))==NULL)
     {
        printf("Khong mo duoc tap tin\n");
        getch();
        exit(1);
     }
   while (getc(fp)!=EOF)
     count++;
   fclose(fp);
   printf("Tap tin %s co %d ki tu",argv[1],count);
   getch();
 }
```

i. Đếm số từ : Ta có thể sửa chương trình trên thành chương trình đếm số từ .

Chương trình 3-5 :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main(int argc,char *argv[])
  {
   FILE *fp;
   char ch,string[81];
   int count = 0;
   int white=1;
   clrscr();
   if (argc!=2)
     {
        printf("  Format c:\<Ten chuong trinh>  <tentaptin>\n");
        getch();
        exit(1);
     }
   if ((fp=fopen(argv[1],"r"))==NULL)
     {
        printf("Khong mo duoc tap tin\n");
        getch();
        exit(1);
     }
   while ((ch=getc(fp))!=EOF)
     switch(ch)
        {
         case ' ': /*nếu có dấu trống , dòng mới hay tab*/
```

```
            case '\t':
            case '\n': white++;
                       break;
            default:if(white)
                    {
                     white=0;
                     count++;
                    }
           }
   fclose(fp);
   printf("Tap tin %s co %d tu",argv[1],count);
   getch();
   return 0;
  }
```

k.Vào ra chuỗi : Đọc hay ghi chuỗi trên tập tin cũng tương tự như đọc hay ghi từng kí tự riêng lẻ . Ta xét một chương trình ghi chuỗi

Chương trình 3-6 :

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>
void main()
  {
   FILE *fp;
   char string[8];
   clrscr();
   if ((fp=fopen("a.txt","w"))==NULL)
     {
        printf("Khong mo duoc tap tin\n");
        getch();
        exit(1);
     }
   while (strlen(gets(string))>0)
     {
        fputs(string,fp);
        fputs("\n",fp);
     }
   fclose(fp);
  }
```

Trong chương trình mỗi chuỗi kết thúc bằng cách gõ enter và kết thúc chương trình bằng cách gõ enter ở đầu dòng mới . Do fputs() không tự động thêm vào mã kết thúc để chuyển dòng mới nên ta phải thêm vào tập tin mã này . Chương trình đọc một chuỗi từ tập tin :

Chương trình 3-7 :

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>
void main()
```

```
{
  FILE *fp;
  char string[81];
  clrscr();
  if ((fp=fopen("a.txt","r"))==NULL)
    {
       printf("Khong mo duoc tap tin\n");
       getch();
       exit(1);
    }
  while (fgets(string,81,fp)!=NULL)
    printf("%s",string);
  fclose(fp);
  getch();
}
```

Hàm fgets() nhận 3 đối số : địa chỉ nơi đặt chuỗi , chiều dài tối đa của chuỗi , và con trỏ chỉ tới tập tin .

l. Vấn đề sang dòng mới : Trong chương trình đếm kí tự ta thấy số kí tự đếm được bao giờ cũng nhỏ hơn số byte có trong tập tin này nhận được bằng lệnh dir của DOS . Khi ta ghi một tập tin văn bản vào đĩa , C tự động ghi vào đĩa cả hai mã CR và LF khi gặp mã sang dòng mới "\n" . Ngược lại khi đọc tập tin từ đĩa , các mã CR và LF được tổ hợp thành mã sang dòng mới . Chương trình sau minh hoa thêm về kĩ thuật vào ra chuỗi , nội dung tương tự lệnh type của DOS

Chương trình 3-8 :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main(int argc,char *argv[])
  {
   FILE *fp;
   char string[81];
   clrscr();
   if (argc!=2)
     {
        printf("Format c:\<ten chuong trinh>  <ten tap tin>");
        getch();
        exit(1);
     }
   if ((fp=fopen(argv[1],"r"))==NULL)
     {
        printf("Khong mo duoc tap tin\n");
        getch();
        exit(1);
     }
   while (fgets(string,81,fp)!=NULL)
     printf("%s",string);
   fclose(fp);
   getch();
   return 0;
```

```
}
```

m. Các tập tin chuẩn và máy in : Trên đây ta đã nói đến cách thức tiếp nhận một con trỏ tham chiếu dến một tập tin trên đĩa của hàm fopen() , C định nghĩa lại tê chuẩn của 5 tập tin chuẩn như sau :

Tên	Thiết bị
in	Thiết bị vào chuẩn (bàn phím)
out	Thiết bị ra chuẩn (màn hình)
err	Thiết bị lỗi chuẩn (màn hình)
aux	Thiết bị phụ trợ chuẩn(cổng nối tiếp)
prn	Thiết bị in chuẩn (máy in)

Ta có thể dùng các tên này để truy cập đến các thiết bị . Chương trình sau dùng hàm fgets(0 và fputs() để in nội dung một tập tin ra máy in

Chương trình 3-9 :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
main(int argc,char *argv[])
 {
  FILE *fp1,*fp2;
  char string[81];
  clrscr();
  if (argc!=2)
    {
       printf("Format c:\<ten chuong trinh>  <ten tap tin>");
       getch();
       exit(1);
    }
  if ((fp1=fopen(argv[1],"r"))==NULL)
    {
       printf("Khong mo duoc tap tin\n");
       getch();
       exit(1);
    }
  if ((fp2=fopen("prn","w"))==NULL)
    {
       printf("Khong mo duoc may in\n");
       getch();
       exit(1);
    }
  while (fgets(string,81,fp1)!=NULL)
    fputs(string,fp2);
  fclose(fp1);
  fclose(fp2);
  getch();
  return 0;
 }
```

Trong chương trình trên máy in được coi là tập tin có tên là prn

n. Nhập xuất định dạng : Trước đây ta đã đề cập đến nhập xuất kí tự . Những số có định dạng cũng có thể ghi lên đĩa như các kí tự . Ta xét chương trình sau

Chương trình 3-10 :

```
#include <stdio.h>
#include <conio.h>
main()
  {
    FILE *p;
    int i,n;
    float x[4],y[4];
    clrscr();
    p=fopen("test.txt","w");
    printf("Cho so cap so can nhap n = ");
    scanf("%d",&n);
    fprintf(p,"%d\n",n);
    printf("Cho cac gia tri x va y\n");
    for (i=0;i<n;i++)
      {
          scanf("%f%f",&x[i],&y[i]);
          fprintf(p,"%f     %f\n",x[i],y[i]);
      }
    fclose(p);
  }
```

```
#include <stdio.h>
#include<conio.h>
#include <string.h>
void main()
  {
    FILE *fp;
    char name[40];
    int code;
    float height;
    int n,i;
    clrscr();
    fp=fopen("b.txt","w");
    printf("Cho so nguoi can nhap : ");
    scanf("%d",&n);
    for (i=0;i<n;i++)
      {
          printf("Nhap ten , ma so va chieu cao : ");
          scanf("%s%d%f",name,&code,&height);
          fprintf(fp,"%s  %d  %f",name,code,height);
      }
    fclose(fp);
  }
```

Chương trình 3-11 :

```
#include <stdio.h>
#include <conio.h>
```

```
void main()
  {
    FILE *p;
    int i,n;
    float x[4],y[4];
    clrscr();
    p=fopen("test.txt","r");
    fscanf(p,"%d",&n);
    for (i=0;i<n;i++)
      {
          fscanf(p,"%f%f",&x[i],&y[i]);
          printf("\n%.3f%8.3f",x[i],y[i]);
      }
    fclose(p);
    getch();
  }
```

```
#include <stdio.h>
#include<conio.h>
#include <string.h>
void main()
  {
    FILE *fp;
    char name[2];
    int code,n,i;
    float height;
    clrscr();
    fp=fopen("b.txt","r");
    fscanf(fp,"%d",&n);
    for (i=0;i<n;i++)
      {
          fscanf(fp,"%s%d%f\n",name,&code,&height);
          printf("%s%3d%8.3f\n",name,code,height);
      }
    fclose(fp);
    getch();
  }
```

§3. KIỂU NHỊ PHÂN VÀ KIỂU VĂN BẢN

1. Mã sang dòng theo hai kiểu : Trong dạng văn bản , một kí tự chuyển dòng tương ứng với 2 mã CR và LF khi ghi vào tập tin trên đĩa . Ngược lại khi đọc , tổ hợp CR/LF trên đĩa tương ứng với kí tự sang dòng mới . Tuy nhiên nếu mở tập tin theo kiểu nhị phân thì 2 mã CR và LF là phân biệt nhau . Từ đó số kí tự mà chương trình đếm được khác với trường hợp mở tập tin bằng kiểu văn bản

Chương trình 3-12 : Chương trình đếm số kí tự bằng cách mở tập tin theo kiểu nhị phân

```
#include <stdio.h>
#include <stdlib.h>
```

```
#include <conio.h>
void main(int argc,char *argv[])
  {
    FILE *fp;
    char string[81];
    int count=0;
    clrscr();
    if (argc!=2)
      {
          printf("Format c:\<ten chuong trinh>  <ten tap tin>");
          getch();
          exit(1);
      }
    if ((fp=fopen(argv[1],"rb"))==NULL)
      {
          printf("Khong mo duoc tap tin\n");
          getch();
          exit(1);
      }
    while (getc(fp)!=EOF)
      count++;
    fclose(fp);
    printf("Tap tin %s co %d ki tu",argv[1],count);
    getch();
  }
```

2. Mã kết thúc tập tin theo 2 kiểu : Sự khác biệt thứ hai khi mở tập tin theo kiểu nhị phân hay kiểu kí tự còn là ở chỗ nhìn nhận kí tự kết thúc tập tin . Nói chung các tập tin đều được quản lí theo kích thước của nó và khi đọc hết số byte đã chỉ ra trong kích thước tập tin thì dấu hiệu EOF sẽ được thông báo , dấu hiệu đó ứng với mã 1Ah(hay 26 ở hệ 10) . Khi đóng tập tin văn bản , mã 1A sẽ được tự động chèn vào cuối tập tin để làm dấu hiệu kết thúc tập tin (tương đương mã Ctrl-Z) . Do vậy nếu bằng cáh nào đó ta chèn mã 1A vào một vị trí giữa tập tin , thì khi mở tập tin theo kiểu văn bản và đọc đến mã này chương trình đọc sẽ ngừng hẳn vì chính lúc đó hàm đọc phát sinh giá trị -1 để báo cho chương trình là đã kết thúc tập tin . Nếu đã lưu số vào tập tin theo dạng nhị phân thì khi mở tập tin cần phải mở theo dạng nhị phân . Nếu không sẽ có một số nào đó là 1A và việc đọc tập tin theo kiểu văn bản sẽ kết thúc ngoài ý định . Tương tự , với tập tin mở theo kiểu nhị phân mã 10 không được nhìn nhận là mã sang dòng mới vì không được xem là tương ứng với tổ hợp CR/LF nữa.

3. Chương trình minh hoạ : Chúng ta xét một chương trình dùng kiểu nhị phân để khảo sát tập tin .

Chương trình 3-13 :

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#define length          10
#define true   0
#define false  -1
void main(int agrc,char *argv[])
  {
```

```
  FILE *fp;
  int ch;
  int j,noteof;
  unsigned char string[length+1];
  clrscr();
  if (agrc!=2)
    {
       printf("Dang c:\<ten chuong trinh> <ten tap tin>");
       getch();
       exit(1);
    }
  if ((fp=fopen(argv[1],"rb"))==NULL)
    {
       printf("Khong mo duoc tap tin\n");
       getch();
       exit(1);
    }
  noteof=true;
  do
    {
       for (j=0;j<length;j++)
         {
          if ((ch=getc(fp))==EOF)
            noteof=false;
          printf("%3x",ch);
          if (ch>31)
            *(string+j)=ch;/* ki tu in duoc*/
          else
            *(string+j)='.';/* ki tu khong in duoc*/
         }
       *(string+j)='\0';
       printf("  %s\n",string);
    }
  while (noteof==true);
  fclose(fp);
  getch();
}
```

4. Các hàm fread và fwrite :

a. Ghi cấu trúc bằng fwrite : Ta xét một chương trình ghi cấu trúc lên đĩa . Trong chương trình ta dùng hàm fread() . Hàm này có 4 đối số : địa chỉ để ghi cấu trúc , kích thước của cấu trúc , số cấu trúc sẽ ghi và con trỏ chỉ tới tập tin .

Chương trình 3-14 :

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
void main()
 {
  char chso[10];
  FILE *fp;
```

```
  struct nguoi {
                  char ten[30];
                  int so;
                  float cao;
                 }nv;
  clrscr();
  if((fp=fopen("nhanvien.rec","wb"))==NULL)
    {
      printf("Khong mo duoc file\n");
      getch();
      exit(1);
    }
  do
    {
      printf("\nCho ten : ");
      gets(nv.ten);
      printf("Cho ma so : ");
      gets(chso);
      nv.so=atoi(chso);
      printf("Cho chieu cao :");
      gets(chso);
      nv.cao=atof(chso);
      fwrite(&nv,sizeof(nv),1,fp);
      printf("Tiep tuc khong y/n?");
    }
  while(getch()=='y');
  fclose(fp);
 }
```

b. Đọc cấu trúc bằng fread : Ta dùng hàm fread() để đọc cấu trúc ghi trên một tập tin . Các đối số của fread() cũng giống như fwrite() . Hàm fread() trả về số của những mục đã được đọc tới . Nếu tập tin đã kết thúc nó cho trị âm . Ta xét ví dụ sau :

Chương trình 3-15 :

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
void main()
 {
  FILE *fp;
  struct nguoi {
                  char ten[30];
                  int so;
                  float cao;
                 }nv;
  clrscr();
  if((fp=fopen("nhanvien.rec","rb"))==NULL)
    {
      printf("Khong mo duoc file\n");
      getch();
      exit(1);
    }
```

```
   do
     {
         printf("\nTen :%s\n",nv.ten);
         printf("Ma so :%03d\n",nv.so);
         printf("Chieu cao :%.2f\n",nv.cao);

     }
   while (fread(&nv,sizeof(nv),1,fp)==1);
   fclose(fp);
   getch();
  }
```

c. Ghi mảng bằng fwrite() : Hàm fwrite() cũng dùng ghi mảng lên đĩa . Ta xét ví dụ sau :

Chương trình 3-16 :

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
int table[10]={1,2,3,4,5,6,7,8,9,10};
void main()
  {

   FILE *fp;
   clrscr();
   if((fp=fopen("table.rec","wb"))==NULL)
     {
         printf("Khong mo duoc file\n");
         getch();
         exit(1);
     }
   fwrite(table,sizeof(table),1,fp);
   fclose(fp);
  }
```

d. Đọc mảng bằng fread() : Sau khi ghi mảng lên đĩa ta có thể đọc các phần tử của mảng từ đĩa bằng hàm fread().

Chương trình 3-17 :

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

void main()
  {

   FILE *fp;
   int a[10];
   int i;
   clrscr();
   if((fp=fopen("table.rec","rb"))==NULL)
     {
```

```
        printf("Khong mo duoc file\n");
        getch();
        exit(1);
     }
   for (i=0;i<10;i++)
     {
        fread(a,sizeof(a),10,fp);
        printf("\%3d",a[i]);
     }
   fclose(fp);
   getch();
  }
```

e. Ví dụ về cơ sở dữ liệu : Ta xét chương trình quản lí nhân viên với các tập tin trên đĩa như sau :

Chương trình 3-18 :

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#define true 1
struct nguoi {
              char ten[30];
              int so;
              float cao;
             };
struct nguoi nv[10];
int n=0;
char numstr[10];

void main()
  {
   char ch;
   void newname(void);
   void listall(void);
   void wfile(void);
   void rfile(void);
   clrscr();
   while (true)
     {
        printf("\nGo 'e' de nhap nhan vien moi\n");
        printf("Go 'l'de liet ke nhan vien\n");
        printf("Go 'w' de ghi len dia\n");
        printf("Go 'r'de doc file tu dia\n");
        printf("Go 'q' de ket thuc chuong trinh\n\n");
        ch=getch();
        switch (ch)
          {
           case 'e':newname();
                    break;
           case 'l':listall();
```

```
                    break;
            case 'w':wfile();
                    break;
            case 'r':rfile();
                    break;
            case 'q': exit(1);
            default : printf("Nhap sai ki tu , chon lai!");
           }
      }
  }

void newname()
  {
    char numstr[81];
    printf("\nBan ghi so %d\nCho ten : ",n+1);
    gets(nv[n].ten);
    printf("Cho ma so co 3 chu so : ");
    gets(numstr);
    nv[n].so=atoi(numstr);
    printf("Cho chieu cao :");
    gets(numstr);
    nv[n++].cao=atof(numstr);
  }
void listall()
  {
    int j;
    if (n<1)
      printf("Danh sach rong\n");
    for (j=0;j<n;j++)
      {
          printf("\nBan ghi so %d\n",j+1);
          printf("Ten :%s\n",nv[j].ten);
          printf("Ma nhan vien : %3d\n",nv[j].so);
          printf("Chieu cao :%4.2f\n",nv[j].cao);
      }
  }

void wfile()
  {
    FILE *fp;
    if (n<1)
      {
          printf("Danh sach rong , khong ghi\n");
          getch();
          exit(1);
      }
    if ((fp=fopen("nv.rec","wb"))==NULL)
      {
          printf("Khong mo duoc file\n");
          getch();
```

```
        exit(1);
     }
   else
     {
        fwrite(nv,sizeof(nv[0]),n,fp);
        fclose(fp);
        printf("Da ghi %3d ban ghi len dia\n",n);
     }
  }

void rfile()
  {
   FILE *fp;
   if ((fp=fopen("nv.rec","rb"))==NULL)
     {
        printf("Khong mo duoc file\n");
        getch();
        exit(1);
     }
   else
     {
        while(fread(&nv[n],sizeof(nv[n]),1,fp)==1)
          {
           clrscr();
           printf("Ban ghi so %3d\n",n+1);
           printf("Ten nhan vien :%s\n",nv[n].ten);
           printf("Ma nhan vien :%3d\n",nv[n].so);
           printf("Chieu cao cua nhan vien :%.2f\n",nv[n].cao);
           getch();
           n++;
          }
        fclose(fp);
        printf("Xong ! Tong so ban ghi da doc %3d\n",n);
     }
  }
```

§4. CÁC FILE NGẪU NHIÊN

Các tập tin đề cập trước đây là các tập tin tuần tự , nghĩa là tập tin mà khi đọc hay ghi đề theo chế độ tuần tự từ đầu đến cuối tập tin . Đối với tập tin tuần tự ta không thể đọc hay ghi một cách trực tiếp tại một vị trí bất kì trên tập tin . Tập tin ngẫu nhiên cho phép ta truy cập ngẫu nhiên vào những vị trí cần thiết trên tập tin . Các hàm dùng khi truy cập tập tin ngẫu nhiên là :

rewind() : di chuyển con trỏ tập tin về đầu tập tin

Cú pháp : void rewind(FILE *fp);

fseek() : di chuyển con trỏ tập tin về vị trí mong muốn

Cú pháp : int fseek(FILE *fp , long sb , int xp)

fp - con trỏ tập tin

sb - số byte cần di chuyển

xp - vị trí xuất phát mà việc dịch chuyển đươợ bắt đầu từ đó . xp có thể có các giá trị sau :

xp=SEEK_SET hay 0 : xuất páht từ đầu tập tin

xp=SEEK_CUR hay 1 : xuất phát từ vị trí con trỏ hiện tại

xp=SEEK_END hay 2 : xuất páht từ cuối tập tin

ftell() : cho biết vị trí hiện tại của con trỏ tập tin

Ta xét chương trình ví dụ sau :

Chương trình 3-19 :

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
void main()
  {
    struct nguoi {
                   char ten[30];
                   int so;
                   float cao;
                  }nv;
    int recno;
    FILE *fp;
    long int offset;
    clrscr();
    if ((fp=fopen("nhanvien.rec","r"))==NULL)
      {
         printf("Khong mo duoc file\n");
         getch();
         exit(1);
      }
    printf("Ban muon doc ban ghi thu may : ");
    scanf("%d",&recno);
    recno--;
    offset=recno*sizeof(nv);
    if (fseek(fp,offset,0)!=0)
      {
         printf("Khong di chuyen duoc con tro file toi do\n");
         getch();
         exit(1);
      }
    fread(&nv,sizeof(nv),1,fp);
    printf("Ten :%s\n",nv.ten);
    printf("Ma nhan vien : %3d\n",nv.so);
    printf("Chieu cao :%4.2f\n",nv.cao);
    getch();
  }
```

§5. LỖI VÀO RA

Nói chung , khi mở tập tin thành công ta có thể ghi lên nó . Tuy nhiên , nhiều trường hợp không mở được tập tin nhưng ta không biết lỗi do đâu . Để xác định llỗi ta dùng hàm

ferror() . Hàm này có đối số là con trỏ tập tin . Hàm sẽ có giá trị không nếu không có lỗi gì . Ngược lại hàm cho giá trị khác không . Ta cũng có thể dùng hàm perror() để chỉ nội dung lỗi .

Chương trình 3-20 :

```
#include <stdio.h>
#include<conio.h>
#include <string.h>
#include <stdlib.h>
void main()
  {
    FILE *fp;
    char name[40],numstr[10];
    int code;
    float height;
    int n,i;
    clrscr();
    fp=fopen("a:\newfile.txt","w");
    printf("Cho so nguoi can nhap : ");
    gets(numstr);
    n=atoi(numstr);
    for (i=0;i<n;i++)
      {
          printf("Nhap ten : ");
          gets(name);
          printf("Nhap ma so : ");
          gets(numstr);
          code=atoi(numstr);
          printf("Nhap chieu cao : ");
          gets(numstr);
          height=atof(numstr);
          fprintf(fp,"%s  %d  %f",name,code,height);
          if (ferror(fp))
            {
              perror("Loi ghi file ");
              getch();
              exit(1);
            }
      }
    fclose(fp);
  }
```

Sau lỗi do ta ghi , trình biên dịch sẽ thông báo lỗi cụ thể trong câu " Loi ghi file : no such file ỏ directory"

§6. VÀO RA Ở MỨC HỆ THỐNG

1.Các tập tin tiêu đề và biến chuẩn : Trong cách vào ra ở mức hệ thống , ta phải khởi tạo bộ đệm rồi đặt dữ liệu vào đó trước ghi hay đọc . Vào ra ở mức hệ thống có lợi ở chỗ lượng mã ít hơn vào ra chuẩn và tốc độ sẽ nhanh hơn . Để dùng các hàm cấp 1 ta phải cần các tập tin tiêu đề sau :

io.h - chứa các prototype của các hàm cấp 1
fcntl.h - chứa các định nghĩa quyền truy cập
sys/stat.h - chá các định nghĩa thuộc tính
dó.h - chứa các thuộc tính theo DOS

2. Tóm tắt các hàm : creat - tạo tập tin mới
_creat - tạo tập tin mới theo kiểu nhị phân
open - mở tập tin
_open - mở tập tin đã tồn tại
close và _close - đóng tập tin
chmod - thay đổi thuộc tính của tập tin
_chmode - thay đổi thuộc tính của tập tin theo kiểu DOS
perror - thông báo lỗi (stdlib.h)
write - ghi một dãy các byte
read - đọc một dãy các byte
lseek - dùng di chuyển con trỏ vị trí

3. Đọc tập tin theo cách vào ra hệ thống : Ta có chương trình đọc tập tin từ đĩa và hiển thị lên màn hình theo cách vào ra hệ thống .

Chương trình 3-21 :

```
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <io.h>
#define BUFFSIZE  512
char buff[BUFFSIZE];
void main(int argc,char *argv[])
  {
   int inhandle,bytes,i;
   clrscr();
   if (argc!=2)
     {
        printf("Dang <ten chuong trinh>   <ten tap tin>");
        getch();
        exit(1);
     }
   if ((inhandle=open(argv[1],O_RDONLY|O_BINARY))<0)
     {
        printf("Khong mo duoc file\n");
        getch();
        exit(1);
     }
   while ((bytes=read(inhandle,buff,BUFFSIZE))>0)
     for (i=0;i<bytes;i++)
        putch(buff[i]);
   close(inhandle);
  }
```

4. Khởi tạo bộ đệm : Trong chương trình ta phải định nghĩa bộ đệm bằng phát biểu

```
#define BUFFSIZE  512
char buff[BUFFSIZE]
```

Nhờ đó ta đọc được dữ liệu từ đĩa vào bộ đệm buff . Với DOS , kích thước bộ đệm nên chọn là bội số của 512.

5. Mở một tập tin : Cũng giống như vào ra bằng hàm cấp 2 , ta phải mở tập tin trước khi đọc hay ghi bằng phát biểu :

inhandle=open(argv[1],O_RDONLY | O_BINARY);

Biểu thức này thiết lập sự liên lạc giữa tập tin và hệ điều hành . Trong biểu thức ta cần một hằng lag oflag là dấu hiệu cho biết mức độ dùng tập tin .

oflag	ý nghĩa
O_APPEND	Đặt con trỏ ở cuối tập tin
O_CREAT	Tạo tập tin mới để ghi(không có hiệu quả nếu tập tin đã có)
O_RDONLY	Mở một tập tin để chỉ đọc
O_RDWR	Mở một tập tin để chỉ đọc hay ghi
O_TRUNC	Mở và cắt bỏ bớt tập tin
O_WRONLY	Mở tập tin để ghi
O_BINARY	Mở tập tin kiểu nhị phân
O_TEXT	Mở tập tin kiểu văn bản

6. Danh số của tập tin : Trong vào ra chuẩn , con trỏ tập tin sẽ nhận được ngay khi gọi hàm fopen() còn trong nhập xuất bằng hàm cấp 1 ta nhậ được giá trị nguyên gọi là danh số của tập tin . Đây là số gán cho một tập tin cụ thể để tham chiếu đến tập tin này . Nếu hàm open() cho ta giá trị -1 nghĩa là danh số không đúng và phát sinh lỗi .

7. Đọc tập tin vào bộ đệm : Để đọc tập tin vào bộ đệm ta dùng lệnh :

byte = read(inhandle , buff , BUFSIZE);

Hàm này có 3 đối số : danh số của tập tin , địa chỉ của bộ đệm và số byte cực đại cần đọc . Giá trị của hàm read() chỉ ra số byte đã đọc được .

8. Đóng tập tin : Để đóng tập tin ta dùng lệnh

close(inhandle);

9. Thông báo lỗi : Khi hàm open() cho giá trị -1 , nghĩa là có lỗi . Dạng lỗi sẽ được đọc bằng perror() . Ta có chương trình ví dụ

Chương trình 3-22 :

```
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <io.h>
#define BUFFSIZE  512
char buff[BUFFSIZE];
void main(int argc,char *argv[])
  {
    int inhandle,bytes,i;
    clrscr();
    if (argc!=2)
      {
          printf("Dang <ten chuong trinh>   <ten tap tin>");
          getch();
          exit(1);
      }
    if ((inhandle=open(argv[1],O_RDONLY|O_BINARY))<0)
```

```
    {
        perror("Khong mo duoc file\n");
        getch();
        exit(1);
    }
  while ((bytes=read(inhandle,buff,BUFFSIZE))>0)
    for (i=0;i<bytes;i++)
        putch(buff[i]);
  close(inhandle);
 }
```

10. Thao tác trên bộ đệm : Việc đưa tập tin vào bộ đệm có lợi là cho phép truy cập trên bộ đệm thay vì trên tập tin . Làm như vậy nhanh hơn truy cập trên đĩa .
Chương trình sau cho phép tìm một từ trong một tập tin văn bản .
Chương trình 3-23 :

```
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <string.h>
#include <io.h>
#define BUFFSIZE  1024
char buff[BUFFSIZE];

void main(int argc,char *argv[])
 {
  int inhandle,bytes;
  void search(char *,int);
  clrscr();
  if (argc!=3)
    {
        printf("Dang <ten chuong trinh> <ten tap tin> <tu can tim>");
        getch();
        exit(1);
    }
  if ((inhandle=open(argv[1],O_TEXT))<0)
    {
        printf("Khong mo duoc file %s\n",argv[1]);
        getch();
        exit(1);
    }
  while ((bytes=read(inhandle,buff,BUFFSIZE))>0)
    search(argv[2],bytes);
  close(inhandle);
  printf("Khong tim thay");
  getch();
 }

void search(char *cau,int buflen)
 {
```

```
  char *p,*ptr;
  ptr=buff;
  while ((ptr=memchr(ptr,cau[0],buflen))!=NULL)
    if (memcmp(ptr,cau,strlen(cau))==0)
        {
          printf("Tu xuat hien lan dau trong cau tai vi tri %d:\n",ptr-buff+1);
          for (p=ptr;p<ptr+strlen(cau);p++)
            putch(*p);
          exit(1);
        }
    else
        ptr++;
 }
```

11. Hàm dùng bộ đệm : Hàm search() là chương trình con minh hoạ cách dùng bộ đệm . Ta có một hàm memchr() dạng :

ptr = memchr(ptr , cau[0] , buflen);

Hàm này dùng để tìm vị trí của kí tự cau[0] trong chuỗi chỉ bởi ptr và độ f\dài của phần cần tìm trong bộ đệm là buflen . Chương trình sẽ truyền argv[2] cho cau . Hàm này cho giá trị NULL khi không tìm thấy kí tự cần tìm . Ngược lại nó sẽ cho địa chỉ của kí tự đã tìm thấy trong bộ đệm . Việc so sánh các chuỗi cau và chuỗi ptr được tiến hành nhờ hàm memcmp trong câu lệnh :

if ((memcmp(ptr,cau,strlen(cau))==0)

Hàm này cũng có 3 đối số là : chuỗi thu nhất ptr , chuỗi thu hai cau và đo dai can so sanh strlen(cau)

12. Ghi lên tập tin : Ghi thông tin lên tập tin phức tạp hơn đọc từ tập tin . Ta có chương trình ví dụ sau dùng để chép từ một tập tin này sang tập tin khác.

Chương trình 3-24 :

```
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <fcntl.h>
#include <io.h>
#include <sys\stat.h>
#define BUFFSIZE  4096
char buff[BUFFSIZE];

void main(int argc,char *argv[])
 {
  int inhandle,outhandle,bytes;

  clrscr();
  if (argc!=3)
    {
      printf("Dang <ten chuong trinh> <ten tap tin 1> <ten tap tin 2>");
      getch();
      exit(1);
    }
  if ((inhandle=open(argv[1],O_RDWR|O_BINARY))<0)
    {
```

```
        printf("Khong mo duoc file %s\n",argv[1]);
        getch();
        exit(1);
    }
  if ((outhandle=open(argv[2],O_CREAT|O_WRONLY|O_BINARY,S_IWRITE))<0)
    {
        printf("Khong mo duoc file %s\n",argv[2]);
        getch();
        exit(1);
    }
  while ((bytes=read(inhandle,buff,BUFFSIZE))>0)
    write(outhandle,buff,bytes);
  close(inhandle);
  close(outhandle);
  printf("Da chep xong");
  getch();
}
```

Trong ví dụ trên ta mở một lúc 2 tập tin với danh số là inhamdle và outhandle Biểu thức mở tập tin nguồn không có gì đặc biệt còn biểu thức mở tập tin đích có dạng :

outhandle = open(argv[2] ,O_CREAT | O_WRONLY | O_BINARY , S_IWRITE)

với O_CREAT để tạo tập tin trên đĩa

O_WRONLY để chỉ ghi lên tập tin

O_BINARY để mở tập tin theo kiểu nhị phân

Khi mở tập tin với O_CREAT , đối thứ 3 của open() là một trong 3 trị :

S_IWRITE : chỉ cho phép ghi lên tập tin

S_IREAD : chỉ cho phép đọc từ tập tin

S_IWRITE | S_IREAD : cho phép đọc và ghi lên tập tin

Để dùng các trị này phải khai báo #include <sys\stat.h> sau khai báo #include<fcntl.h> . Hàm write() có đối tương tự như read() . Trong vòng lặp while hàm read() báo số byte đọc được qua biến bytes và hàm write() sẽ biết số bytes cần ghi vào tập tin đích . Trong chương trình ta dùng bộ đệm với kích thước khá lớn để chương trình chạy nhanh .

CHƯƠNG 4 : BỘ NHỚ VÀ HIỂN THỊ KÍ TỰ

§1. KHÁI NIỆM CHUNG

Trong phần này ta sẽ xem xét việc xử lí hiển thị kí tự bằng cách xâm nhập trực tiếp vào bộ nhớ (direc memory access-DMA) . Ta sẽ tìm hiểu cách xâm nhập trực tiếp màn hình . Cách này nhanh hơn là dùng các hàm của C .

§2. CÁC TOÁN TỬ BITWISE

1. Toán tử bitwise and (&) : C dùng 6 toán tử bitwise được tóm tắt trong bảng sau

Phép toán	Kí hiệu
and	&
or	|
xor	^
dịch phải	>>
dịch trái	<<
đảo	~

Các phép toán này có thể áp dụng cho dữ liệu kiểu int , char nhưng không áp dụng cho số float .

Toán tử & (khác với and logic &&) cần hai toán hạng có kiểu giống nhau . Các toán hạng này được and bit với bit . Toán tử & thường dùng kiểm tra xem một bit cụ thể nào đó có trị là bao nhiêu . Ví dụ để kiểm tra bit thứ 3 cuả biến ch có trị 1 hay 0 ta dùng phép toán :

ch &0x08;

2. Toán tử or : Toán tử or (khác or logic | |) thường dùng kết hợp các bit từ các biến khác nhau vào một biến duy nhất . Ví dụ ta có hai biến là ch1 và ch2 và giả sử các bit từ 0..3 của ch1 chứa các trị mong muốn còn các bit 4..7 của ch2 chứa các trị mong muốn . Khi viết :

a = ch1|ch2;

thì cả 8 bit của a đều chứa trị mong muốn .

3. Toán tử dịch phải >> : Toán tử này làm việc trên một biến duy nhất . Toán tử này dịch từng bit trong toán hạng sang phải . Sô bit dịch chuyển được chỉ định trong số đi theo sau toán tử . Việc dịch phải một bit đồng nghĩa với việc chia toán hạng cho 2 . Ví dụ : 0 1 1 1 0 0 1 0 dịch sang phải 1 bit sẽ là

0 0 1 1 1 0 0 1

4. Đổi từ số hex sang số nhị phân : Ta dùng các toán tử bitwise để đổi một số từ hệ hex sang hệ 2 . Chương trình như sau :

Chương trình 4-1 :

```
#include <conio.h>
#include <stdio.h>

void main()
  {
    int i,num,bit;
    unsigned int mask;
    char string[10],ch;
    clrscr();
```

```
  do
   {
      mask=0x8000;
      printf("\nBan cho mot so : ");
      scanf("%x",&num);
      printf("Dang nhi phan cua so %x la : ",num);
      for (i=0;i<16;i++)
        {
         bit=(mask&num)? 1 : 0;
         printf("%d",bit);
         if (i==7)
           printf("  ");
         mask >>= 1;
        }
      printf("\nBan muon tinh tiep khong(c/k)?");
      ch=getch();
   }
  while (ch=='c');
  getch();
}
```

Trong chương trình trên ta dùng vòng lặp for để duyệt qua 16 bit của biến nguyên từ trái qua phải . Lõi của vấn đề là các phát biểu :

```
bit = (mask&num)? 1 : 0;
mask >>=1
```

Trong phát biểu đầu tiên mask là biến chỉ có một bit 1 duy nhất ở phía trái nhất . Mask này được & với num để xem bit trái nhất của num có là 1 hay là 0 . Nếu kết quả khác 0 (true) bit tương ứng của num là 1 còn ngược lại bit tương ứng là 0 . Sau mỗi lần & mask được dịch trái 1 bit để xác định bit tiếp theo của num là 0 hay 1 .

5. Các toán tử bitwise khác :

a. Toán tử xor ^ : Toán tử xor trả về trị 1 khi chỉ có 1 bit chứ không phải 2 bit có trị là 1

0	0	0
0	1	1
1	0	1
1	1	0

Toán tử xor cần khi lật bit nghĩa là hoán chuyển giữa 1 và 0 vì 1 xor với 1 là 0 và 1 xor với 0 là 1 . Ví dụ để lật bit thứ 3 trong biến ch ta dùng :

```
ch ^ 0x08
```

b. Toán tử dịch phải << : Toán tử này tương tự toán tử dịch trái . Giá trị của bit chèn vào bên phải luôn luôn bằng 0 . Dịchphải tương ứng với việc nhân số đó cho 2 .

c. Toán tử đảo : Toán tử này là toán tử một ngôi . Nó tác động lên các bit của toán hạng và đảo trị của bit từ 1 sang 0 và từ 0 sang 1 . Đảo 2 lần một số ta lại nhận được số cũ .

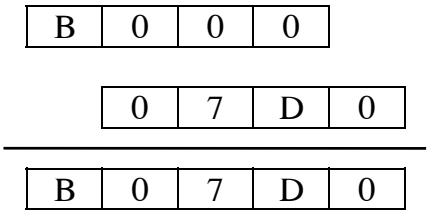
§3. BỘ NHỚ MÀN HÌNH

1. Khái niệm chung : Màn hình thông thường có 25 dòng và mỗi dòng có 80 kí tự . Như vậy toàn bộ màn hình có 2000 kí tự . Mỗi kí tự trên màn hình tương ứng với một địa chỉ cụ thể trong bộ nhớ màn hình . Mỗi kí tự dùng 2 byte của bộ nhớ này : byte thứ nhất chứa mã ASCII của kí tự (từ 0 đến 255 nay từ 0 đến ff)và byte thứ 2 chứa thuộc tính của nó . Như vậy để hiển thị 2000 kí tự , bộ nhớ màn hình phải cần 4000 byte . Bộ nhớ màn hình đơn sắc bắt đầu tại B000h và kết thúc tại B0F9F . Nếu ta có màn hình màu thì vùng nhớ cho chế độ văn bản sẽ bắt đầu từ B8000h và kết thúc tại B8F9F . Khi muốn hiển thị kí tự trên màn hình , đoản trình thư vện C sẽ gọi đoản trình ROM-BIOS để đặt kí tự cần hiển thị vào địa chỉ tương ứng trong bộ nhớ nàm hình. Nếu muốn có tốc độ nhanh , hãy chèn trực tiếp các giá trị trên vào vùng nhớ màn hình .

2. Con trỏ far : Khi biết địa chỉ , cách thông dụng để đưa giá trị vào bộ nhớ là dùng con trỏ . Do vậy nếu ta cần đưa kí tự vào vị trí đầu tiên của vùng nhớ màn hình thì ta viết :

```
int *ptr ;
ptr = 0xB800;
*(ptr)=ch;
```

Đoạn chương trình trên có vẻ hợp lí nhưng lại không làm việc vì biến con trỏ thông thường có hai byte trong khi địa chỉ B0000h lại dài 5 chữ số (2,5 byte) . Lí do dẫn đến tình trạng này là do con trỏ thường dùng để chứa đại chỉ nằm trong một đoạn duy nhất mà thôi . Trong họ 8086 , một đoạn dài 10000h hay 65535 byte . Bên trong các đoạn địa chạy từ 0h đến FFFFh . Thông thường các dữ liệu của chương trình C nằm trong một đoạn nên để thâm nhập các địa chỉ nằm ngoài đoạn ta phải dùng một cơ chế khác . Bên trong 8086 , tình trạng này được khắc phục bằng cách dùng các thanh ghi gọi là thanh ghi đoạn . Các địa chỉ nằm ngoài đoạn đucợ tạo lập bằng tổ hợp địa chỉ đoạn và địa chỉ offset .



Trong hình trên địa chỉ đoạn B000h được dịch trái 4 bit rồi cộng với địa chỉ offset 07D0 tạo ra địa chỉ tuyệt đối B07D0h.

3. Dùng địa chỉ đoạn : offset trong C : Như vậy khi địa chỉ nằm bên ngoài đoạn dữ liệu , C dùng tổ hợp đoạn : offset và yêu cầu dạng biểu diễn 32 bit(4 byte , 8 chữ số hex) với 4 chữ số cho địa chỉ đoạn và 4 chữ số cho địa chỉ offset . Do vậy C coi địa chỉ tuyệt đối B07D0 là 0xB00007D0 (B000 và theo sau là 07D0) . Trong C con trỏ 32 được tính bằng cách dịch địa chỉ đoạn sang trái 16 bit và cộng với địa chỉ offset . Do con trỏ thông thường không thể cất giữ địa chỉ dài 32 bit nên ta phải dùng con trỏ far Con trỏ này cất giữ địa chỉ dài 4 byte . Vì vậy chương trình sẽ là :

```
int far *ptr ;
ptr = 0xB8000000;
*(ptr)=ch;
```

4. Dùng một kí tự để tô màn hình : Chúng ta dùng con trỏ far để ghi lên màn hình 2000 bản sao của một kí tự . Điều này tương tự như dùng putch() . Chương trình kết thúc ghi gõ x

Chương trình 4-2 :

```
#include <conio.h>
```

```
#include <stdio.h>
#define length         2000
void main()
  {
    int far *fptr;
    int add;
    char ch;
    clrscr();
    printf("Go vao mot ki tu , go lai de thay doi");
    fptr=(int far*)0xB8000000;
    while((ch=getche())!='x')
      for (add=0;add<length;add++)
          *(fptr+add)=ch|0x0700;
  }
```

Trong chương trình phát biểu :

```
*(fptr+add)=ch|0x0700;
```

dùng để điền đầy kí tự lên màn hình . Biến ch là kí tự muốn đặt vào bộ nhớ . Hằng số 0x0700 là byte thuộc tính , có nghĩa là không chớp nháy , không đậm , chữ trắng trên nền đen .

Phát biểu khác cần giải thích :

```
fptr=(int far*)0xB8000000;
```

Ta dùng dấu ngoặc vì hằng 0xB8000000 và biến int far fptr có kiểu khác nhau : hằng có vẻ là số nguyên dài còn fptr lại là con trỏ chỉ đến kiểu int . Do đó để tránh nhắc nhở của trình biên dịch ta cần biến đổi kiểu làm hằng trở tthành con trỏ far chỉ đến int. Màn hình có thể được xem như một mảng hai chiều gồm các hàng và cột . Địa chỉ tương ứng trong bộ nhớ được tính từ phép nhân số hiệu hàng với số lượng cột trong một hàng rồi cộng kết quả và số hiệu cột với địa chỉ bắt đầu của vùng nhớ màn hình . Ta có chương trình sau :

Chương trình 4-3 :

```
#include <conio.h>
#include <stdio.h>
#define rowmax 25
#define colmax 80

void main()
  {
    int far *fptr;
    int row,col;
    char ch;
    clrscr();
    printf("Go vao mot ki tu , go lai de thay doi");
    fptr=(int far*)0xB8000000;
    while((ch=getche())!='x')
      for (row=0;row<rowmax;row++)
          for (col=0;col<colmax;col++)
            *(fptr+row*colmax+col)=ch|0x0700;
  }
```

5.Trình xử lí văn bản theo dòng: Để biết thên về sự tiện lợi của con trỏ far ta xét thêm một trình xử lí văn bản theo dòng . Trình xử lí này chỉ làm việc trên một dòng văn bản . Ta sẽ tiến hành theo 2 bước : đầu tiên là một chương trình cho phép người dùng gõ vào một dòng

và di chuyển con nháy tới lui . Có thể xoá kí tự nhờ di chuyển con nháy tới đó và ghi đè lên nó . Chương trình như sau :

Chương trình 4-4 :

```
#include <conio.h>
#include <dos.h>
#define colmax 80
#define rarrow 77
#define larrow 75
#define video 0x10
#define ctrlc '\x03'
int col=0;
int far *fptr;
union REGS reg;

void main()
  {
    char ch;
    void clear(void);
    void cursor(void);
    void insert(char);
    fptr=(int far*)0xB8000000;
    clear();
    cursor();
    while((ch=getch())!=ctrlc)
      {
          if (ch==0)
            {
              ch=getch();
              switch (ch)
                {
                    case rarrow : if (col<colmax)
                                  ++col;
                                  break;
                    case larrow : if (col>0)
                                  --col;
                                  break;
                }
            }
          else
            if (col<colmax)
              insert(ch);
          cursor();
      }
  }

void cursor()
  {
    reg.h.ah=2;
    reg.h.dl=col;
    reg.h.dh=0;
```

```
  reg.h.bh=0;
  int86(video,&reg,&reg);
 }

void insert(char ch)
 {
  *(fptr+col)=ch|0x0700;
  ++col;
 }

void clear()
 {
  int j;
  for (j=0;j<2000;j++)
    *(fptr+j)=0x0700;
 }
```

Để xoá màn hình ta điền số 0 vào vùng nhớ màn hình với thuộc tính 07 . Sau đó con nháy được di chuyển về đầu màn hình nhờ phục vụ ấn định vị trí con nháy như sau :

ngắt 10h
ah=0;
dh=số hiệu dòng
dl= số hiệu cột
bh=số hiệu trang , thường là 0

Phát biểu switch dùng để đoán nhận các phím được nhận là phím thường hay phím chức năng . Phím mũi tên dùng tăng giảm col và gọi hàm cursor() để di chuyển con nháy tới đó . Nếu kí tự gõ vào là kí tự thường , nó được chèn vào nhờ hàm insert() .

6. Byte thuộc tính : Một kí tự trên màn hình được lưu giữ bởi 2 byte : một byte là mã của kí tự và byte kia là thuộc tính của nó . Byte thuộc tính được chia làm nhiều phần , bit nào bằng 1 thì thì thuộc tính tương ứng được bật . Bit thứ 3 điều khiển độ sáng còn bit thứ 7 điều khiển độ chớp nháy . Các bit còn lại là : 6 - thành phần đỏ của màu nền ; 5 - thành phần green của màu nền ; 4 - thành phần blue của màu nền ; 2 - thành phần đỏ của màu chữ ; 1 - thành phần green của màu chữ ; 0 - thành phần blue của màu chữ . Ta lập một chương trình để điền đầy màn hình bằng các kí tự chớp nháy .

Chương trình 4-5 :

```
#include <conio.h>
#include <stdio.h>
#define rowmax 25
#define colmax 80

void main()
 {
  int far *fptr;
  int row,col;
  char ch;
  clrscr();
  printf("Go vao mot ki tu , go lai de thay doi");
  fptr=(int far*)0xB8000000;
  while((ch=getche())!='x')
   for (row=0;row<rowmax;row++)
```

```
        for (col=0;col<colmax;col++)
          *(fptr+row*colmax+col)=ch|0x8700;
  }
```

Để bật chớp nháy ta để bit thứ 7 thành 1 , 3 bit màu nền 0 , 1 và 2 được đặt trị 1 nên nền sẽ là đen . Byte thuộc tính lúc này là 10000111 = 87h.

7. Chương trình điền thuộc tính : Để hiểu sâu hơn thuộc tính của kí tự ta xét chương trình sau

Chương trình 4-6 :

```
#include <conio.h>
#include <stdio.h>
#define rowmax 25
#define colmax 80
void main()
  {
    int far *fptr;
    char ch,attr=0x07;
    void fill(char,char);
    clrscr();
    printf("Go n cho chu binh thuong,\n");
    printf("Go b cho chu xanh nuoc bien,\n");
    printf("Go i cho chu sang,\n");
    printf("Go c cho chu chop nhay,\n");
    printf("Go r cho chu dao mau\n");
    while((ch=getche())!='x')
      {
        switch (ch)
          {
            case 'n':attr=0x07;
                     break;
            case 'b':attr=attr&0x88;
                     attr=attr|0x01;
                     break;
            case 'i':attr=attr^0x08;
                     break;
            case 'c':attr=attr^0x80;
                     break;
            case 'r':attr=attr&0x88;
                     attr=attr|0x70;
                     break;
          }
        fill(ch,attr);
      }
  }
void fill(char ch,char attr)
  {
    int far *fptr;
    int row,col;
    fptr=(int far*)0xB8000000;
    for (row=0;row<rowmax;row++)
```

```
    for (col=0;col<colmax;col++)
        *(fptr+row*colmax+col)=ch|attr<<8;
 }
```

Trong hàm fill() ta có lệnh

```
*(fptr+row*colmax+col)=ch|attr<<8;
```

vì attr là kí tự nên phải dịch trái 8 bit trước khi kết hợp với ch .

8. Trở lại xử lí văn bản : Bây giờ chúng ta đã biết thuộc tính của kí tự và ta sẽ mở rộng chương trình xử lí văn bản bằng cách thêm vào việc chèn và huỷ bỏ kí tự ,đổi màu .

Chương trình 4-7 :

```
#include <conio.h>
#include <dos.h>
#define colmax 80
#define rarrow 77
#define larrow 75
#define video 0x10
#define norm 0x07
#define blue 0x01
#define bkspc 8
#define altu 22
#define ctrlc '\x03'
int col=0;
int length=0;
int far *fptr;
union REGS reg;

void main()
 {
  char ch,attr=norm;
  void clear(void);
  void cursor(void);
  void insert(char,char);
  void del(void);
  fptr=(int far*)0xB8000000;
  clear();
  cursor();
  while((ch=getch())!=ctrlc)
   {
      if (ch==0)
       {
        ch=getch();
        switch (ch)
          {
            case rarrow : if (col<length)
                            ++col;
                          break;
            case larrow : if (col>0)
                            --col;
                          break;
            case altu   : attr=(attr==norm)? blue:norm;
          }
```

```
          }
        else
          switch (ch)
            {
              case bkspc: if (length>0)
                            del();
                          break;
              default : if (length<colmax)
                          insert(ch,attr);
            }
          cursor();
      }
  }

void cursor()
  {
    reg.h.ah=2;
    reg.h.dl=col;
    reg.h.dh=0;
    reg.h.bh=0;
    int86(video,&reg,&reg);
  }

void insert(char ch,char attr)
  {
    int i;
    for (i=length;i>col;i--)
      *(fptr+i)=*(fptr+i-1);
    *(fptr+col)=ch|attr<<8;
    ++length;
    ++col;
  }

void del()
  {
    int i;
    for (i=col;i<=length;i++)
      *(fptr+i-1)=*(fptr+i);
    --length;
    --col;
  }

void clear()
  {
    int j;
    for (j=0;j<2000;j++)
      *(fptr+j)=0x0700;
  }
```

Khi gõ tổ hợp phím Alt+U sẽ lật biến attr qua lại giữa norm(thuộc tính 07) và blue (cho chữ màu xanh - thuộc tính 01) . Hàm insert(0 có vòng lặp for dùng để thâm nhập trực

tiếp bộ nhớ và con trỏ far để dịch các kí tự sang trái khi cần chèn . Tiến trìmh dịch phải bắt đầu từ cuối câu để tránh ghi đè lên .

§4. CÁC KIỂU BỘ NHỚ TRONG C

1. Địa chỉ đoạn và offset : Trong C kiểu bộ nhớ là khái niệm để chỉ về lượng các phần bộ nhớ khác nhau mà chương trình có thể chiếm . C cho phép 6 kiểu bộ nhớ là tiny , small , compact , medium , large và huge . Kiểu bộ nhớ mặc định là small .

Bộ vi xử lí dùng các thanh ghi 16 bit để ghi địa chỉ . Thanh ghi 16 bit lưu được ffffh byte hay 65536 hay 64 Kb địa chỉ . Vùng nhớ có kích thước này gọi là đoạn . Để truy cập địa chỉ nằm ngoài đoạn , bộ vi xử lí phải dùng hai thanh ghi là thanh ghi đoạn và thanh ghi offset . Địa chỉ thực được tính bằng cách dịch địa chỉ của thanh ghi đoạn sang trái 4 bit rồi cộng với thanh ghi offset . Làm như vậy ta đánh địa chỉ được fffffh hay 1048576 = 1Mb .

2. Hai loại chỉ thị của bộ vi xử lí : Bộ vi xử lí dùng hai kĩ thuật khác nhau để tham chiếu dữ liệu trong bộ nhớ . Nếu vị trí cần tham chiếu nằm trong đoạn 64Kb và đoạn này đã được chỉ định trong thanh ghi đoạn thì bộ vi xử lí chỉ cần dùng một lệnh duy nhất để truy cập dữ liệu . Cách này tương ứng với việc dùng con trỏ near trong C và thực hiện rất nhanh . Trái lại nếu bộ vi xử lí cần tham chiếu ô nhớ nằm ngoài đoạn thì đầu tiên nó phải thay đổi địa chỉ đoạn và sau đoa là địa chỉ offset . Điều này tương ứng với việc dùng con trỏ far trong C và thực hiện khá chậm .

3. Các kiểu Compact , small , medium và large : Có 4 loại chỉ thị của bộ vi xử lí ứng với 4 kiểu bộ nhớ trong C

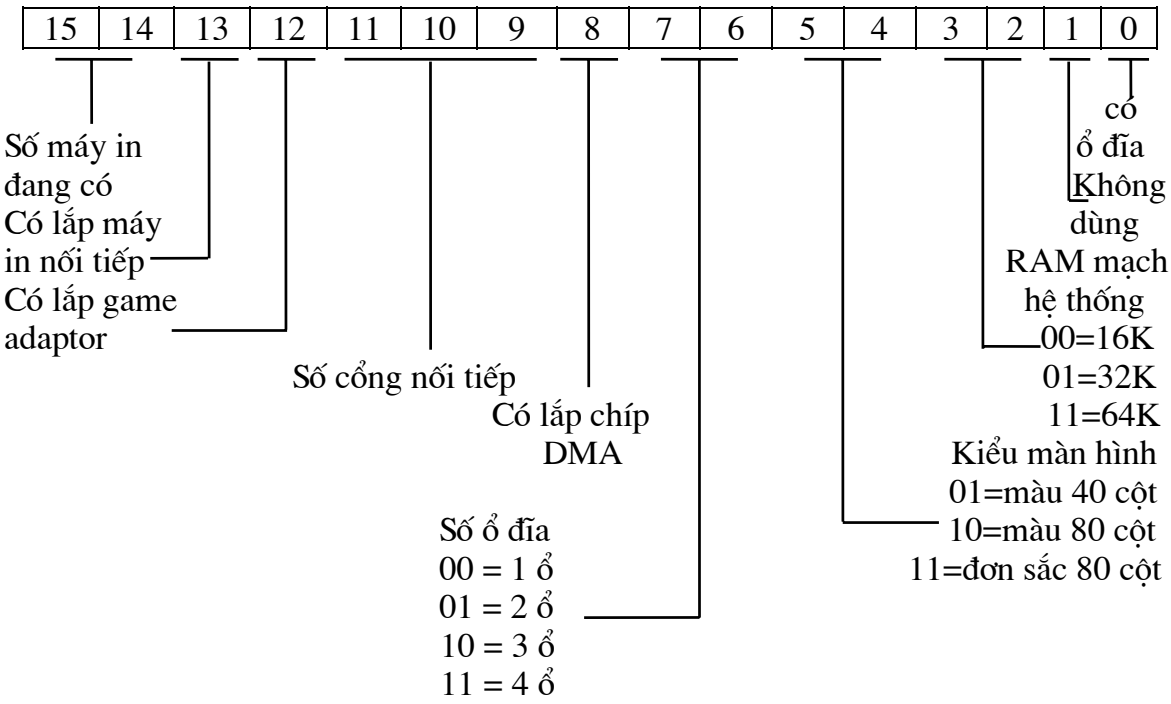
Kiểu	Mã	Dữ liệu
small	near	near
medium	far	near
compact	near	far
large	far	far

Nếu mã chương trình nằm gọn trong một đoạn 64 K và mã dữ liệu nằm gọn trong một đoạn 64 K khác thì kiểu bộ nhớ small là thích hợp . Nếu mã chương trình lớn hơn 64 K và mã dữ liệu nằm gọn trong một đoạn 64 K khác thì hãy dùng kiểu bộ nhớ medium. Nếu mã chương trình nhỏ hơn 64 K và mã dữ liệu lớn hơn 64 K thì hãy dùng kiểu bộ nhớ compact. Nếu cả mã chương trình và mã dữ liệu lớn hơn 64 K thì hãy dùng kiểu bộ nhớ large .

4. Kiểu tyni và kiểu huge : Kiểu tyni được dùng trong các trường hợp đặc biệt với lượng bộ nhớ cho cả mã chương trình lẫn mã dữ liệu nằm gọn trong một đoạn . Kiểu này được dùng để tạo ra tập tin dạng *.com . Kiểu huge được dùng chô một mục dữ liệu (thường là mảng) mà bản thân nó lớn hơn 64K .

§5. TỪ CHỨA DANH MỤC THIẾT BỊ

Đây là một vùng bộ nhớ dài 2 byte nằm trong vùng nhớ thấp có địa chỉ tuyệt đối là 410h chứa thông tin về thiết bị được nối với máy tính. Để truy cậo từ này ta dùng con trỏ far . Con trỏ sẽ chỉ tới đoạn 0000 , địa xhỉ offset là 0410h và được biểu diễn trong C là 00000410 hay 0x410



Để xem xét từng bit và nhóm bit trong từ này chúng ta sẽ dùng các toán tử bitweise . Nói chung ta sẽ dịch từ chứa danh mục thiết bị sang phải và đưa các bit cần quan tâm vào phía phải của từ và che các bit không quan tâm ở phái trái bằng toán tử and . Ngoài từ chứa danh mục thiết bị ta có thể đọc từ chứa kích thước bộ nhớ tại địa chỉ 413h .

Chương trình 4-8 :

```
#define eqlist 0x410
#define memsiz 0x413
#include <conio.h>
#include <stdio.h>
void main()
  {
    int far *fptr;
    unsigned int eq,data;
    clrscr();
    fptr=(int far *)eqlist;
    eq=*(fptr);
    data=eq>>14;
    printf("So may in la : %d\n",data);
    if (eq&0x2000)
      printf("Co may in noi tiep\n");
    data=(eq>>9)&7;
    printf("So cong noi tiep la :%d\n",data+1);
    if (eq&1)
      {
          data=(eq>>6)&3;
          printf("So dia mem la :%d\n",data);
      }
    else
```

```
  printf("Khong co dia mem\n");
 data=(eq>>4)&3;
 switch (data)
  {
      case 1: printf("Man hinh mau 40 cot\n");
              break;
      case 2: printf("Man hinh mau 80 cot\n");
              break;
      case 3: printf("Man hinh don sac 80 cot\n");
  }
 fptr=(int far *)memsiz;
 printf("Dung luong bo nho :%dKbyte\n",*(fptr));
 getch();
}
```

CHƯƠNG 5 : TRUY CẬP TRỰC TIẾP BỘ NHỚ

§1.CÁC HÀM TRUY CẬP THEO ĐỊA CHỈ PHÂN ĐOẠN

1. Hàm pokeb() : Dùng để gửi một kí tự value vào bộ nhớ tại địa chỉ phân đoạn off . Nguyên mẫu của hàm trong dos.h là :

void pokeb(unsigned seg,unsigned off , char value)

2. Hàm peekb() : Nhận một byte tại địa chỉ seg:off . Nguyên mẫu của hàm trong dos.h là :

char peekb(unsigned seg,unsigned off)

3. Hàm poke() : Gửi một số nguyên value vào bộ nhớ tại địa chỉ seg:off . Nguyên mẫu của hàm trong dos.h là :

void poke(unsigned seg,unsigned off , int value)

4. Hàm peek() : Nhận một word tại địa chỉ seg:off . Nguyên mẫu của hàm trong dos.h là :

void peek(unsigned seg,unsigned off)

5. Hàm movedata() : Sao n byte từ địa chỉ seg_gui:off_gui đến địa chỉ seg_nhan:off_nhan . Nguyên mẫu của hàm trong menu.h là :

void movedata(unsigned seg_gui,unsigned off_gui , unsigned seg_nhan,unsigned off_nhan , int n)

§2. ĐỔI TỪ ĐỊA CHỈ PHÂN ĐOẠN SANG ĐỊA CHỈ THỰC

1. Đổi từ địa chỉ thực : Để đổi từ địa chỉ thực sang địa chỉ phân đoạn ta dùng macro sau :

unsigned FP_SEG(địa chỉ thực)

unsigned FP_OFF(địa chỉ thực)

2. Đổi từ địa chỉ phân đoạn : Để đổi từ địa chỉ phân đoạn sang địa chỉ thực ta dùng macro :

void far *MK_FP(seg:off)

Ví dụ : Sau khi thực hiện các câu lệnh:

```
char buf[100]
unsigned ds,dx;
ds= FP_SEG(buf)
dx= FP_OFF(buf)
```

thì ds:dx chứa địa chỉ của nảmg buf .

Sau khi thực hiện câu lệnh :

```
char *pchar;
pchar = (char *) MK_FP(0xb800:0)
```

thì pchar trỏ tới đầu bộ nhớ màn hình . Khi đó ta có thể dùng các lệnh gán để truy cập trực tiếp tới bộ nhớ màn hình .

Chương trình 5-1 : Lập chương trình xác định địa chỉ của một ngắt .

```
#include <dos.h>
#include <conio.h>
#include <stdio.h>
void main()
  {
    unsigned char far *p;
    int n,k;
    unsigned seg,off;
    clrscr();
    p=(unsigned char far*)MK_FP(0,0);
```

```
  while(1)
    {
        printf("\nSo hieu ngat(Bam 0 de ket thuc): ");
        scanf("%d",&n);
        if (n==0)
          break;
        k=(n-1)*4;
        off=p[k]+256*p[k+1];
        seg=p[k+2]+256*p[k+3];
        printf("\nDia chi cua ngat %x : %x",seg,off);
    }
 }
```

Số hiệu của ngắt được đánh số từ 0 nhưng n được nhập từ 1 , mỗi ngắt chiếm 4 byte nên ta có

```
k=(n-1)*4;
```

CHƯƠNG 6 : ĐỒ HOẠ TRONG C

§1. KHÁI NIỆM CHUNG

Turbo C có khoảng 100 hàm đồ hoạ . Các hàm này được chia làm hai kiểu :

- Loại theo kiểu văn bản (ví dụ hàm tạo cửa sổ)
- Loại theo kiểu đồ hoạ

§2. HÀM THEO KIỂU VĂN BẢN

Các hàm này được dùng với màn hình đơn sắc hay màn hình đồ hoạ . Ta phải đặt vào đầu chương trình dòng #include <conio.h> .

1. Cửa sổ : Mục đích của các hàm đồ hoạ theo kiểu văn bản là tạo ra các cửa sổ . Cửa sổ là vubgf hình chữ nhật trên màn hình dùng để giới hạn vùng xuất dữ liệu . Nếu ta soạn thảo văn bản trong cửa sổ thì con nháy chỉ di chuyển trong phạm vi của cửa sổ chứ không phải toàn bộ màn hình . Ta xét một chương trình tạo ra cửa sổ và điền đầy vào đó dòng " Xin chao "

Chương trình 6-1 :

```
#include <conio.h>
#include <dos.h>
#define left 10
#define top 8
#define right 52
#define bot 21

void main()
  {
    int i;
    clrscr();
    window(left,top,right,bot);
    textcolor(RED);
    textbackground(GREEN);
    for (i=0;i<100;i++)
      {
          cputs(" Xin chao ");
          delay(100);
      }
    gotoxy(15,8);
    cputs("Ket thuc");
    getche();
  }
```

Trong chương trình ta có hàm :

window(x1,y1,x2,y2) dùng để ấn định một cửa sổ có toạ độ góc trên trái là x1,y1 và góc dưới phải là x2,y2

textcolor(RED) để ấn định màu chữ là đỏ

textbackcolor(GREEN) để ấn định màu nền văn bản là xanh lá cây

gotoxy(x,y) để di chuyển con nháy về toạ độ x,y

cputs(string) để đặt chuỗi string trong một cửa sổ . Khi gặp biên của cửa sổ chuỗi sẽ được xuống dòng . Màu trong chế độ đồ hoạ được quy định như sau :

Số	Màu
0	BLACK
1	BLUE
2	GREEN
3	CYAN
4	RED
5	MAGENTA
6	BROWN
7	LIGHTGRAY
8	DARKGRAY
9	LIGHTBLUE
10	LIGHTGREEN
11	LIGHTCYAN
12	LIGHTRED
13	LIGHTMAGENTA
14	YELLOW
15	WHITE

2. Dời chỗ văn bản : Muốn dời chỗ một vùng hình chữ nhật của văn bản từ nơi này sang nơi khác ta dùng hàm movetext() . Ta xét chương trình sau tạo ra một cửa sổ , điền đầy cửa sổ bằng một đoạn văn bản và dời cửa sổ sang vị trí khác trên màn hình

Chương trình 6-2 :

```
#include <conio.h>
#include <dos.h>
#define left 26
#define top 7
#define right 65
#define bot 20
#define desleft 1
#define destop 1
#define numcolor 16

void main()
  {
    int i;
    clrscr();
    window(left,top,right,bot);
    textbackground(GREEN);
    for (i=0;i<100;i++)
      {
          textcolor(i%numcolor);
          cputs(" Xin chao ");
          delay(200);
      }
    delay(2000);
    movetext(left,top,right,bot,desleft,destop);
```

```
  getche();
 }
```

Hàm movetext(x1,y1,x2,y2,x0,y0) dùng di chuyển cửa sổ x1,y1,x2,y2 đến vị trí mới mà toạ độ góc trên trái bây giờ là x0,y0 .

3.Lưu trữ và phục hồi màn hình văn bản : Ta có thể lưu trữ một vùng văn bản hình chữ nhật trên màn hình và sau đó phục hồi lại tại một vị trí nào đó trên màn hình . Nhờ vậy ta có thể tạo một cửa sổ nhỏ trên đầu văn bản hiện hành . Ta xét ví dụ sau

Chương trình 6-3 :

```
#include <conio.h>
#include <dos.h>
#define left 1
#define top 1
#define right 80
#define bot 25
int buf[80][25];
void main()
 {
  int i,x,y;

  clrscr();
  for (i=0;i<300;i++)
    cputs(" Turbo C ");
  getche();
  gettext(left,top,right,bot,buf);
  clrscr();
  getch();
  puttext(left,top,right,bot,buf);
  getch();
 }
```

Chương trình lưu toàn bộ màn hình vào vùng đệm có tên là buf nhớ hàm gettext(x1,y1,x2,y2,buf) lưu vn trong hình chữ nhật x1,y1,x2,y2 vào biến buf . Hàm puttext(x1,y1,x2,y2,buf) đặt lại văn bản trong hình chữ nhật x1,y1,x2,y2 lưu bởi biến buf ra màn hình .

3. Một số hàm đồ hoạ văn bản khác :

- void clreol(void) : xoá đến cuối dòng
- int cprintf(const char *format) đưa kí tự ra một cửa sổ
- void textattr(int newattr) ấn định màu cùng lúc cho văn bản và nền
- void gettextinfo(struct text_info *r) : đọc các thông tin như kiểu màn hình , vị trí và kích thước cửa sổ , màu nền và văn bản ,vị trí con nháy
- void normvideo(void) trả lại độ sáng cũ
- void insline(void) : chèn thêm một dòng
- void delline(void) xoá một dòng
- void hightvideo(void) tăng độ sáng
- void lowvideo(void) : giảm độ sáng
- void textmode(int newmode) chuyển đổi giữa các kiểu văn bản . Hàm dùng các đối số sau :

Trị	Hằng	Ý nghĩa

-1	LASTMODE	Kiểu văn bản trước đó
0	BW40	Đen trắng 40 cột
1	C40	Màu 40 cột
2	BW80	Đen trắng 80 cột
3	C80	Màu 80 cột
7	MONO	Đơn sắc 80 cột

§3. CÁC HÀM ĐỒ HOẠ

1. Khởi tạo kiểu đồ hoạ : Để khởi tạo đồ hoạ ta dùng hàm initgraph() được khai báo trong graphics.h với cú pháp :

void far initgraph(int *graphdrive , int *graphmode , char *path);

với các biến graphdrive chứa trình điều khiển đồ hoạ
graphmode kiểu đồ hoạ
path đường dẫn đến thư mục chứa các drive đồ hoạ . Trong phần này ta phải dùng hai dấu \\ vì dấu \ đã được dùng cho kí tự escape .

Để thuận tiện ta khởi tạo đồ hoạ tự động bằng cách viết :

graphdrive = detect;
initgraph(graphdrive , graphmode , path);

Ta có chương trình vẽ đường thẳng và đường tròn như sau :

Chương trình 6-4 :

```
#include <graphics.h>
#include <conio.h>
void main()
  {
   int gd,gm;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   line(0,0,100,100);
   circle(100,100,50);
   getche();
   closegraph();
  }
```

2. Lỗi đồ hoạ : Để biết lỗi đồ hoạ ta dùng hàm int far graphresult(void) . Sau khi biết mã lỗi ta chuyển nó sang cho hàm grapherrormsg() . Hàm này trả về con trỏ chỉ đến lỗi . Sau đây là chương trình minh hoạ

Chương trình 6-5 :

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
void main()
  {
   int gd,gm,ge;
   char *ep;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   ge=graphresult();
```

```
  if (ge)
    {
       printf("Ma loi %d",ge);
       ep=grapherrormsg(ge);
       puts(ep);
       getch();
       exit(1);
    }
  line(0,0,100,100);
  circle(100,100,50);
  getche();
  closegraph();
}
```

3. Đường thẳng và màu sắc : Để thiết lập dạng , mẫu và bề dày của đường thẳng ta dùng hàm void far setlinestyle(int style,int pattern, int thickness) . Tham biến style có thể là :

Trị	Hằng	Y nghĩa
0	SOLID_LINE	Đường đặc
1	DOTTED_LINE	Đường chấm
2	CENTER_LINE	Đường gạch
3	DASHED_LINE	Đường gạch dài
4	USERBIT_LINE	Đường tự tạo

Tham biến thickness có thể nhân một trong hai giá trị sau :

Trị	Hằng	Y nghĩa
1	NORM_WIDTH	dãy 1 điểm ảnh
2	THICK_WIDTH	dãy 3 điểm ảnh

Để xác định màu cho đường thẳng ta dùng hàm void setcolor(int color) . Ta có chương trình sau

Chương trình 6-6 :

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>

void main()
  {
   int gd,gm,ge;
   int x1=0,y1=0;
   int x2=199,y2=199;
   int xc=100,yc=100;
   int r=90;
   char *ep;
   gd=DETECT;
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   ge=graphresult();
   if (ge)
```

```
    {
        printf("Ma loi %d",ge);
        ep=grapherrormsg(ge);
        puts(ep);
        getch();
        exit(1);
    }
  setlinestyle(1,1,1);
  setcolor(LIGHTGREEN);
  line(x1,y1,x2,y2);
  circle(xc,yc,r);
  getche();
  closegraph();
 }
```

4. Ellipse và đa giác : Để vẽ ellipse ta dùng hàm

void far ellipse(int x,int y , int gd,int gc,int xr , int yr)

x,y - toạ độ tâm ellipse

gd,gc - góc bắt đầu vẽ và góc kết thúc vẽ

xr,yr - toạ độ tâm ellipse

Chương trình 6-7 : Vẽ một loạt ellipse

```
#include <graphics.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
void main()
 {
  int gd,gm,ge;
  int x=150,y=150;
  int g1=0,g2=360;
  int xr=150,yr;
  char *ep;
  gd=DETECT;
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  ge=graphresult();
  if (ge)
    {
        printf("Ma loi %d",ge);
        ep=grapherrormsg(ge);
        puts(ep);
        getch();
        exit(1);
    }
  setcolor(RED);
  for (yr=0;yr<100;yr+=10)
    ellipse(x,y,g1,g2,xr,yr);
  getche();
  closegraph();
 }
```

Để vẽ đa giác ta dùng hàm

void far drawpoly(int number , int far *addrlist)
number - số đỉnh đa giác cộng thêm 1
addrlist - mảng chứa toạ độ các đỉnh , toạ độ điểm đầu và cuối phải trùng nhau

Chương trình 6-8 : Vẽ một hình hộp chữ nhật

```
#include <graphics.h>
#include <conio.h>
#define left 50
#define top 50
#define right 150
#define bot 180
int a[]={150,50,180,20,180,135,150,180};
int b[]={50,47,150,47,180,17,95,17,50,47};

void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    rectangle(left,top,right,bot);
    setcolor(2);
    drawpoly(4,a);
    drawpoly(5,b);
    getche();
    closegraph();
  }
```

5. Tô màu và mẫu tô : Turbo C có nhiều hàm để tô màu . Hàm thông dụng nhất để tô bên trong một đa giác và mẫu tô hiện hành là void far fillpoly(int number , int far * addlist) . Màu và mẫu tô được thiết lập nhờ hàm void far setfillstyle(int pattern , int color) . Biến pattern có thể nhận một trong các trị sau :

Trị	Hằng	Ý nghĩa
0	EMPTY_FILL	Rỗng
1	SOLID_FILL	Màu đặc
2	LINE_FILL	Đường ngang
3	LTSLASH_FILL	//// chéo mảnh
4	SLASH_FILL	//// chéo dày
5	BKSLASH_FILL	\\\\ chéo ngược
6	LTBKSLASH_FILL	\\\\ chéo ngược mảnh
7	HATCH_FILL	Sọc dưa thưa
8	XHATCH_FILL	Sọc dưa dày
9	INTERLEAVE_FILL	Đường xen kẽ
10	WIDE_DOT_FILL	Chấm thưa
11	CLOSE_DOT_FILL	Chấm dày
12	USER_FILL	Mẫu tự do

Biến color được chọn theo danh sách đã liệt kê trong phần setcolor(). Nếu dùng giá trị không hợp lệ cho pattern và color thì hàm graphresult() sẽ trả về mã lỗi là -11 . Hàm floodfill() dùng để to màu một hình kín . Nó cần biết điểm bắt đầu tô . Hàm sẽ tô cho đến khi gặp đường biên có màu xác định bằng biến border . Có thể tô bên trong hay ngoài hình vẽ tuỳ điểm bắt đầu . Nếu tô một vùng không kín thì màu tô sẽ lan ra trong lẫn ngoài vật thể . Sau đây là chương trình tô vòng tròn .

Chương trình 6-9 :

```
#include <graphics.h>
#include <conio.h>
#define x 200
#define y 200
#define r 150

void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    circle(x,y,r);
    setfillstyle(CLOSE_DOT_FILL,BLUE);
    floodfill(x,y,RED);
    getche();
    closegraph();
  }
```

Màu dùng để tô có thể giống hay khác với màu dùng cho đường viền của vùng . Tuy vậy màu dùng cho tham biến border của floodfill() phải giống màu vè vật thể (trong chương trình là màu RED)

6. Đồ thị : Turbo C có nhiều hàm giúp đơn giản hoá việc vẽ đồ thị các hàm là bar() , bar3d() và pieslice() .

void bar (int top , int left , int right , int bottom)

void far bar3d(int left , int top , int right , int right , int bottom , int depth , int topflag)

topflag = 0 - có nắp , topflag = 1 - không có nắp

void far pieslice(int x , int y , int startangle , int endangle , int r)

Ta có chương trình minh hoạ

Chương trình 6-10 :

```
#include <graphics.h>
#include <conio.h>
#define n 10
#define bwidth 10
#define sep 12
#define di (bwidth+sep)
#define shft 15
#define width ((n+1)*di)
#define left 5
#define depth 3
#define topflag 1
#define bot 170
```

```
#define top 5
#define ppd (float)(bot-top)/100

void main()
  {
    int gd,gm,i;
    int data[n]={41,47,54,62,63,59,75,83,89,96};
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    setcolor(RED);
    rectangle(top,left,left+width,bot);
    for (i=0;i<n;i++)
      {
          setfillstyle(SOLID_FILL,1+i%3);
          bar3d(left+shft+i*di,bot-data[i]*ppd,left+shft+i*di+bwidth,bot,depth,topflag);
      }
    getche();
    closegraph();
  }
```

Sau đây là chương trình dùng pieslice()

Chương trình 6-11 :

```
#include <graphics.h>
#include <conio.h>
#define n 6
#define r 90
int data[n]={11,19,44,32,15,7};

void main()
  {
    int gd,gm,i,x,y;
    float datasum,startangle,endangle,relangle;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    x=getmaxx()/2;
    y=getmaxy()/2;
    setcolor(RED);
    for (i=0,datasum=0;i<n;i++)
      datasum+=data[i];
    endangle=0;
    relangle=10;
    for (i=0;i<n;i++)
      {
          startangle=(i+1)*relangle;
          setfillstyle(SOLID_FILL,i%4);
          pieslice(x,y,startangle,endangle,r);
          getch();
      }
```

```
  getche();
  closegraph();
 }
```

7. Viewport : Viewport là một vùng nhì thấy được của màn hình . Khi mới khởi động viewport là toàn bộ màn hình . Để xác định một viewport ta dùng hàm setviewport() có cú pháp :

void far setviewport(int left , int top , int right , int bot , int clip)

Tham biến clip cho biết hình vẽ có hiện ra ngoài viewport hay không . Nếu clip <>0 thì không thấy được hình bên ngoài viewport . Để xoá một viewport ta dùng hàm void far clearviewport(void)

Chương trình 6-12 :

```
#include <graphics.h>
#include <conio.h>
void main()
 {
  int gd,gm,i;
  int left=0,top=0,right=150,bot=150;
  int x1=0,y1=0,x2=199,y2=199;
  int x=100,y=100;
  int clip=1;
  int r=90;
  gd=DETECT;
  clrscr();
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  setviewport(left,top,right,bot,clip);
  setcolor(RED);
  rectangle(left,top,right,bot);
  line(x1,y1,x2,y2);
  circle(x,y,r);
  getch();
  closegraph();
 }
```

8. Vẽ theo toạ độ tương đối : Trong C ta có thể dùng toạ độ tương đối so với điểm hiện hành CP-current point . Để vẽ đường thẳng ta dùng hàm void far lineto(int x, int y) . Hàm này vẽ đường thẳng từ điểm CP đến điểm mới có toạ độ là x,y . Hàm void far linerel(int dx , int dy) vẽ đường thẳng từ CP(xc,yc) đến điểm có toạ độ (xc+dx,yc+dy) . Thường ta hay kết hợp với hàm void far moveto(int x, int y) để di chuyển điểm hiện hành tới điểm mới có toạ độ (x,y)

Chương trình 6-13 : Vẽ một bàn cờ

```
#include <graphics.h>
#include <conio.h>
#define max 160
#define grid 20
#define size 18
void main()
 {
  int gd,gm,i,j;
  void square(int );
```

```
   gd=DETECT;
   clrscr();
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   for (i=0;i<max;i+=grid)
     for (j=0;j<max;j+=grid)
          {
           moveto(j,i);
           square(size);
          }
   getch();
   closegraph();
  }

void square(int side)
  {
   linerel(side,0);
   linerel(0,side);
   linerel(-side,0);
   linerel(0,-side);
  }
```

9. Điểm ảnh : Để đặt một điểm ảnh lên màn hình ta dùng hàm :

void far putpixel(int x , in y, int color)

Chương trình 6-14 : Lập chương trình vẽ hình sin bằng putpixel

```
#include <graphics.h>
#include <conio.h>
#include <math.h>
void main()
  {
   int gd,gm,x,y;
   double g,sg;
   gd=DETECT;
   clrscr();
   initgraph(&gd,&gm,"c:\\bc\\bgi");
   line (1,100,200,100);
   for (x=0;x<200;x++)
     {
          g=((double)x/200)*(2*3.14159);
          sg=sin(g);
          y=100-100*sg;
          putpixel(x,y,RED);
     }
   getch();
   closegraph();
  }
```

Để xác định màu của một điểm ta dùng hàm int getpixel(int x,int y)

10 . Ảnh bit và làm ảnh chuyển động : Để cất gữi một hình ảnh vào bộ nhớ ta dùng hàm :

void far getimage(int left , int top , int right , int bot , void far * addbuf)

left , top , right , bot - các góc của hình chữ nhật chứa ảnh
addbuf - địa chỉ bộ nhớ dùng chứa ảnh

Hàm này cần biết kích thước của hình . Kích thước này được xác định theo hàm :

unsigned far imagesize(int left , int top , int right , int bot)

Giá trị của hàm được truyền cho hàm malloc() để cấp phát bộ nhớ . Con trỏ do hàm malloc() trả về được truyền cho hàm putimage để khôi phục lại hình đã cất . Cú pháp của putimage() là :

void far putimage(int left , int top , void far * addbuf,int putop)
left,top là góc trên trái của vùng sẽ đưa ảnh ra
addbuf - địa chỉ bộ nhớ dùng chứa ảnh
putop là các đưa ảnh ra . Các hằng putop là :

Trị	Hằng	Ý nghĩa
0	COPY_PUT	Thay hình cũ bằng hình mới
1	XOR_PUT	XOR hình cũ bằng hình mới
2	OR_PUT	OR hình cũ bằng hình mới
3	AND_PUT	AND hình cũ bằng hình mới
5	NOT_PUT	Thay hình cũ bằng đảo hình mới

Chương trình 6-15 : Lập chương trình thể hiện quả bóng dội

```
#include <graphics.h>
#include <conio.h>
#include <alloc.h>
#define left 0
#define top 0
#define right 639
#define bottom 479
#define r 8

void main()
  {
    int gd,gm,x,y;
    int dx,dy,oldx,oldy;
    void far *buf;
    unsigned size;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    rectangle(left,top,right,bottom);
    x=y=r+10;
    setcolor(LIGHTGREEN);
    setfillstyle(SOLID_FILL,LIGHTGREEN);
    circle(x,y,r);
    floodfill(x,y,LIGHTGREEN);
    size=imagesize(x-r,y-r,x+r,y+r);
    buf=malloc(size);
    getimage(x-r,y-r,x+r,y+r,buf);
    dx=1;
    dy=1;
```

```
  while (!kbhit())
    {
       putimage(x-r,y-r,buf,COPY_PUT);
       delay(5);
       oldx=x;
       oldy=y;
       x=x+dx;
       y=y+dy;
       if (x<=left+r+2||x>=right-r-2)
         dx=-dx;
       if (y<=top+r+1||y>=bottom-r-2)
         dy=-dy;
       putimage(oldx-r,oldy-r,buf,XOR_PUT);
       delay(5);
    }
  closegraph();
}
```

§4. VĂN BẢN TRONG ĐỒ HOẠ

1. Các fonts : Để chọn fonts chữ ta dùng hàm :

void far settextstyle(int font , int direction , int charsize)

Các fonts chứa trong các tập tin trong bảng sau

Trị	Hằng	Tập tin
0	DEFAULT_FONT	Cài sẵn
1	TRIPLEX_FONT	trip.chr
2	SMALL_FONT	litt.chr
3	SANSERIF_FONT	sans.chr
4	GOTHIC_FONT	goth.chr
5	SCRIPT_FONT	scrip.chr
6	SIMPLEX_FONT	simp.chr
7	TRIPLEX_SCR_FONT	tscr.chr
8	COMPLEX_FONT	lcom.chr
9	EUROPEAN_FONT	euro.chr
10	BOLD_FONT	bold.chr

Đối direction có thể nhận một trong hai trị :

0 (HORIZ_DIR) - từ trái sang phải

1 (VERT_DIR) - từ trên xuống dưới

Khi đối charsize có trị là 1 , kích thước chữ là nhỏ nhất . Khi kích thước là 2 , chữ sẽ tăng gấp đôi v.v. Để in chuỗi ra màn hình trong chế độ đồ hoạ ta dùng các hàm :

void far outtext(char far * string);

void far outtextxy(int x , int y , char far *string);

Chương trình 6-16 : Dùng hàm settextstyle() để viết chữ

```
#include <graphics.h>
#include <conio.h>
```

```
#define FONTSIZE 4
void main()
  {
    int gd,gm;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    settextstyle(GOTHIC_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,0,"Gothic");
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,40,"Triplex");
    settextstyle(SMALL_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,80,"Small");
    settextstyle(SANS_SERIF_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,100,"Sanserif");
    settextstyle(DEFAULT_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,160,"Default");
    settextstyle(EUROPEAN_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,200,"Euro");
    settextstyle(BOLD_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,240,"Bold");
    settextstyle(COMPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,300,"Complex");
    settextstyle(SCRIPT_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,340,"Script");
    settextstyle(SIMPLEX_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,370,"Simplex");
    settextstyle(TRIPLEX_SCR_FONT,HORIZ_DIR,FONTSIZE);
    outtextxy(0,420,"Triplex script");
    getch();
    closegraph();
  }
```

2. Justify và định kích thước văn bản : Hàm định vị trí văn bản là ;

void far settextjustify(int horiz , int vert);

Đối horiz nhận các biến trong bảng sau

Trị	Hằng	Ý nghĩa
0	LEFT_TEXT	CP nằm bên trái văn bản
1	CENTER_TEXT	CP nằm bên chính giữa văn bản
2	RIGHT_TEXT	CP nằm bên phải văn bản

Đối vert nhận một trong các giá trị sau :

Trị	Hằng	Ý nghĩa
0	BOTTOM_TEXT	CP nằm ở đáy văn bản
1	CENTER_TEXT	CP nằm bên chính giữa văn bản
2	TOP_TEXT	CP nằm ở đỉnh văn bản

Chương trình 6-17 :

```
#include <graphics.h>
#include <conio.h>
#define cl 150
#define lead 40
#define fontsize 3
void main()
  {
    int gd,gm,i;
    gd=DETECT;
    clrscr();
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    settextstyle(TRIPLEX_FONT,HORIZ_DIR,fontsize);
    line(cl,0,cl,200);
    for (i=0;i<lead*5;i+=lead)
      line(0,i,300,i);
    moveto(cl,0);
    outtext("Default");
    moveto(cl,lead);
    settextjustify(LEFT_TEXT,TOP_TEXT);
    outtext("Left-top");
    moveto(cl,lead*2);
    settextjustify(RIGHT_TEXT,TOP_TEXT);
    outtext("Right-top");
    moveto(cl,lead*3);
    settextjustify(CENTER_TEXT,TOP_TEXT);
    outtext("Center-top");
    moveto(cl,lead*4);
    settextjustify(CENTER_TEXT,BOTTOM_TEXT);
    outtext("Center-bottom");
    getch();
    closegraph();
  }
```

Để thay đổi kích thước và tỉ lệ chữ ta dùng hàm :

void far setusercharsize(int multx , int divx , int multy , int divy);

multx - nhân chiều rộng của kí tự

divx - chia chiều rộng của kí tự

multy - nhân chiều cao của kí tự

divx - chia chiều cao của kí tự

Chương trình 6-18 : Tạo một đồ thị có ghi chú

```
#include <graphics.h>
#include <conio.h>
#include <stdlib.h>
#define n 12
#define bwidth 20
#define sep 24
#define shft 30
#define left 5
```

```
#define depth 6
#define topflag 1
#define bot 300
#define top 5
#define ticks 10
#define twidth 10
#define maxdata 100
#define xtitle 40
#define ytitle 40
#define font SANS_SERIF_FONT
#define di (bwidth+sep)
#define width (((n+1)*di))
#define pbt ((float)(bot-top))
#define ppd ((float)(bot-top)/maxdata)

void main()
  {
    int gd,gm,i;
    float a,b,c,d;
    int data[n]={41,47,54,62,63,59,75,83,89,96,55,2};
    char month[12][4]={"Jan","Feb","Mar","Apr","May","Jun","Jul",
                       "Aug","Sep","Oct","Nov","Dec"};
    char string[40];
    clrscr();
    gd=DETECT;
    initgraph(&gd,&gm,"c:\\bc\\bgi");
    rectangle(left,top,left+width,bot);
    setusercharsize(4,3,4,3);
    settextstyle(font,HORIZ_DIR,0);
    moveto(xtitle,ytitle);
    outtext("1998 Sales");
    setusercharsize(2,3,2,2);
    settextstyle(font,HORIZ_DIR,0);
    for (i=0;i<ticks;i++)
      {
          line(left,bot-i*pbt/10,left+twidth,bot-i*pbt/10);
          line(left+width-twidth,bot-i*pbt/10,left+width,bot-i*pbt/10);
          moveto(left+width+sep,bot-(i+1)*pbt/10);
          itoa(i*(maxdata/ticks),string,10);
          outtext(string);
      }
    setusercharsize(2,3,2,2);
    settextstyle(font,VERT_DIR,0);
    for (i=0;i<n;i++)
      {
          setfillstyle(SOLID_FILL,i);
          bar3d(left+shft+i*di,bot-data[i]*ppd,left+shft+i*di+bwidth,bot,depth,topflag);
          moveto(left+i*di+bwidth-1,bot+5);
          outtext(month[i]);
      }
```

```
  getch();
  closegraph();
 }
```

§5. VÍ DỤ KẾT THÚC

Chương trình 6-19 : Lập chương trình vẽ một mặt Mallbrot

```
#include <graphics.h>
#include <conio.h>
#include <stdlib.h>
#define ymax 400
#define xmax 400
#define maxcount 16
void main()
 {
  int gd,gm;
  int x,y,count;
  float xscale,yscale;
  float left,top,xside,yside,zx,zy,cx,cy,tempx;
  clrscr();
  gd=DETECT;
  initgraph(&gd,&gm,"c:\\bc\\bgi");
  left=-2.0;
  top=1.25;
  xside=2.5;
  yside=-2.5;
  xscale=xside/xmax;
  yscale=yside/ymax;
  rectangle(0,0,xmax+1,ymax+1);
  for (y=1;y<=ymax;y++)
   {
      for (x=1;x<=xmax;x++)
       {
        cx=x*xscale+left;
        cy=y*yscale+top;
        zx=zy=0;
        count=0;
        while((zx*zx+zy*zy<4) && (count<maxcount))
         {
            tempx=zx*zx-zy*zy+cx;
            zy=2*zx*zy+cy;
            zx=tempx;
            count++;
         }
        putpixel(x,y,count);
        if (kbhit())
         exit(0);
       }
```

```
    }
  getch();
  closegraph();
}
```

CHƯƠNG 7 : MỘT SỐ VẤN ĐỀ VỀ ĐA THỨC VÀ HÀM SỐ

§1. MỘT SỐ KHÁI NIỆM CHUNG

1. Khái niệm về phương pháp tính : Phương pháp tính là môn học về những lí luận cơ bản và các phương pháp giải gần đúng,cho ra kết quả bằng số của các bài toán thường gặp trong toán học cũng như trong kĩ thuật.

Chúng ta thấy rằng hầu hết các bài toán trong toán học như giải các phương trình đại số hay siêu việt,các hệ phương trình tuyến tính hay phi tuyến,các phương trình vi phân thường hay đạo hàm riêng,tính các tích phân,... thường khó giải đúng được,nghĩa là khó tìm kết quả dưới dạng các biểu thức.

Một số bài toán có thể giải đúng được nhưng biểu thức kết quả lại cồng kềnh,phức tạp khối lượng tính toán rất lớn.Vì những lí do trên,việc giải gần đúng các bài toán là vô cùng cần thiết.

Các bài toán trong kĩ thuật thường dựa trên số liệu thực nghiệm và các giả thiết gần đúng.Do vậy việc tìm ra kết quả gần đúng với sai số cho phép là hoàn toàn có ý nghĩa thực tế.

Từ lâu người ta đã nghiên cứu phương pháp tính và đạt nhiều kết quả đáng kể. Tuy nhiên để lời giải đạt được độ chính xác cao,khối lượng tính toán thường rất lớn.Với các phương tiện tính toán thô sơ,nhiều phương pháp tính đã được đề xuất không thể thực hiện được vì khối lượng tính toán quá lớn.Khó khăn trên đã làm phương pháp tính không phát triển được.

Ngày nay nhờ máy tính điện tử người ta đã giải rất nhanh các bài toán khổng lồ,phức tạp,đã kiểm nghiệm được các phương pháp tính cũ và đề ra các phương pháp tính mới. Phương pháp tính nhờ đó phát triển rất mạnh mẽ.Nó là cầu nối giữa toán học và thực tiễn.Nó là môn học không thể thiếu đối với các kĩ sư.

Ngoài nhiệmvụ chính của phương pháp tính là tìm các phương pháp giải gần đúng các bài toán,nó còn có nhiệm vụ khác như nghiên cứu tính chất nghiệm,nghiên cứu bài toán cực trị,xấp xỉ hàm v.v. Trong phần này chúng ta sẽ nghiên cứu một loạt bài toán thường gặp trong thực té và đưa ra chương trình giải chúng.

2. Các đặc điểm của phương pháp tính : Đặc điểm về phương pháp coả môn học này là hữu hạn hoá và rời rạc hoá.

Phương pháp tính thường biến cái vô hạn thành cái hữu hạn,cái liên tục thành cái rời rạc và sau cùng lại trở về với cái vô hạn,cái liên tục.Nhưng cần chú ý rằng quá trình trở lại cái vô hạn,cái liên tục phải trả giá đắt vì khối lượng tính toán tăng lên rất nhiều.Cho nên trong thực tế người ta dừng lại khi nghiệm gần đúg sát với nghiệm đúng ở một mức độ nào đó.

Đặc diểm thứ hai của môn học là sự tiến đến kết quả bằng quá trình liên tiếp.Đó là quá trình chia ngày càng nhỏ hơn,càng dày đặc hơn hoặc quá trình tính toán bước sau dựa vào các kết quả của các bước trước.Công việc tính toán lặp đi lặp lại này rất thích hợp với máy điện toán.

Khi nghiên cứu phương pháp tính người ta thường triệt để lợi dụng các kết quả đạt được trong toán học.Cùng một bài toán có thể có nhiều phương pháp tính khác nhau.Một phương pháp tính được coi là tốt nếu nó đạt các yêu cầu sau :

- phương pháp tính được biểu diễn bằng một dãy hữu hạn các bước tính cụ thể.Các bước tính toán cụ thể này của phương pháp tính được gọi là thuật toán. Thuật toán càng đơn giản càng tốt.
- đánh giá được sai số và sai số càng nhỏ càng tốt.
- thuật toán thực hiện được trên máy điện toán và thời gian chạy máy ít nhất

3. Các loại sai số : Trong việc thiếtlập và giải các bài toán thực tế ta thường gặp các loại sai số.

Giả sử ta xét bài toán A nào đó.Nghiên cứu các quy luật liên hệ giữa các đại lượng trong bài toán dẫn đến phương trình có dạng tổng quát :

$$y = Bx$$

Trong đó : x - đại lượng đã biết

y - đại lượng chưa biết

B - quy luật biến đổi từ x sang y

Bài toán thực tế thường rất phức tạp.Để đơn giản và có thể diễn đạt nó bằng toán học,người ta đưa ra một số giả thiết không hoàn toàn chính xác để nhận được phương trình trên.

Vì vậy nếu gọi y_1 là giá trị đúng của y thì khi đó $y \neq y_1$. Giá trị $|y - y_1|$ được gọi là *sai số giả thiết của bài toán*.

Do x là số liệu ban đầu của bài toán,thu được từ đo lường,thí nghiệm nên nó chỉ là giá trị gần đúng.Sai số này được gọi là *sai số của các số liệu ban đầu*.

Để giải gần đúng phương trình trên ta thường thay B bằng C hay x bằng t để phương trình đơn giản hơn và có thể giải được.Bằng cách đó ta tìm được y_2 gần đúng với y.Giá trị $|y_2 - y|$ được gọi là *sai số phương pháp của bài toán*.

Cuối cùng khi thực hiện các phép tính ta thường thu gọn các kết quả trung gian hay kết quả cuối cùng nên đáp số của bài toán là y_3.Giá trị $|y_3 - y|$ là *sai số tính toán*.

Trong phần này chúng ta quan tâm tới sai số phương pháp.

4. Xấp xỉ và hội tụ : Xét bài toán

$$y = Bx$$

Giả sử y là nghiệm đúng của bài toán mà ta chưa biết.Bằng phương pháp nào đó ta lấy y_1 thay cho y và khi đó y_1 gọi là xấp xỉ thứ nhất của nghiệm và viết :

$$y_1 \approx y$$

Cũng bằng phương pháp tương tự,ta xây dựng được một dãy các xấp xỉ $y_1,y_2,y_3,..y_n$.Nếu ta có :

$$\lim_{n\to\infty} y_n = y$$

thì ta nói dãy xấp xỉ hội tụ tới nghiệm y.

§2. TÍNH GIÁ TRỊ CỦA ĐA THỨC THEO SƠ ĐỒ HORNER

1. Sơ đồ Horner : Giả sử chúng ta cần tìm giá trị của một đa thức tổng quát dạng :

$$P(x) = a_0x^n + a_1x^{n-1} + a_2x^{n-2} +....+ a_n \qquad (1)$$

tại một trị số x nào đó. Trong (1) các hệ số a_i là các số thực đã cho. Chúng ta viết lại (1) theo thuật toán Horner dưới dạng :

$$P(x_o) = (...((a_0x + a_1)x+ a_2x)+...+ a_{n-1})x + a_n \qquad (2)$$

Từ (2) ta nhận thấy :

$$P_0 = a_0$$
$$P_1 = P_0x + a_1$$
$$P_2 = P_1x + a_2$$
$$P_3 = P_2x + a_3$$
.....
$$P(x) = P_n = P_{n-1}x + a_n$$

Tổng quát ta có :

$$P_k = P_{k-1}x + a_k \text{ với } k =1,2...n \text{ ; } P_0 = a_0$$

Do chúng ta chỉ quan tâm đến trị số của P_n nên trong các công thức truy hồi về sau chúng ta sẽ bỏ qua chỉ số k của P và viết gọn P := Px + ak với k = 0...n.Khi ta tính tới k = n thì P chính là giá trị cần tìm của đa thức khi đã cho x. Chúng ta thử các bước tính như sau :

Ban đầu			$P = 0$
Bước 0	$k = 0$		$P = a_o$
Bước 1		$k = 1$	$P = a_o x + a_1$
Bước 2		$k = 2$	$P = (a_o x + a_1)x + a_2$
.....			
Bước n-1	$k = n - 1$		$P = P(x_o) = (...((a_o x + a_1)x+a_2 x)+...+a_{n-1})x$
Bước n		$k = n$	$P = P(x_o) = (...((a_o x + a_1)x+a_2 x)+...+a_{n-1})x + a_n$

Sau đây là chương trình thực hiên thuật toán trên

Chương trình 7-1

```
#include <conio.h>
#include <stdio.h>
#define m  10

void main(void)
 {
   int  k,n;
   float p,x;
   float a[m];

   clrscr();
   printf("\nCho bac cua da thuc n = ");
   scanf("\%d",&n);
   printf("Vao cac he so a:\n");
   for (k=1;k<=n+1;k++)
           {
        printf("a[%d] = ",k-1);
        scanf("%f",&a[k]);
     };
   printf("Cho gia tri x = ");
   scanf("%f",&x);
   p=0.0;
   for (k=1;k<=n+1;k++)
     p=p*x+a[k];
   printf("Tri so cua da thuc P tai x =%.2f la :%.5f",x,p);
   getch();
 }
```

2. Sơ đồ Horner tổng quát : Giả sử chúng ta có đa thức :

$P_n(x) = a_0 x^n + a_1 x^{n-1} + a_2 x^{n-2} +....+ a_n$ (1)

Khai triển Taylor của đa thức tại $x = x_o$ có dạng :

$$P_n(x) = P_n(x_0) + \frac{P'(x_0)}{1!}(x - x_0) + \frac{P''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{P^{(n)}(x_0)}{2!}(x - x_0)^n \qquad (2)$$

Mặt khác chúng ta có thể biến đổi đa thức về dạng :

$$P_n(x) = (x - x_o)P_{n-1}(x) + P_n(x_o) \qquad (3)$$

Trong đó $P_{n-1}(x)$ là đa thức bậc n-1 và có dạng :

$$P_{n-1}(x) = b_o x^{n-1} + b_{o-1}x^{n-2} + b_2 x^{n-3} + \ldots + b_{n-1} \quad (4)$$

Thuật toán để tìm các hệ số nhận được bằng cách so sánh (1) và (3) :

$b_o = a_o$

$b_i = a_i + b_{i-1}x_o$

$b_n = P_n(x_o)$

So sánh (2) và (3) ta có :

$$(x - x_0)P_{n-1}(x_0) + P_n(x_0) = P_n(x_0) + \frac{P'(x_0)}{1!}(x - x_0) + \frac{P''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{P^{(n)}(x_0)}{2!}(x - x_0)^n$$

hay :

$$(x - x_0)P_{n-1}(x) = \frac{P'(x_0)}{1!}(x - x_0) + \frac{P''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{P^{(n)}(x_0)}{2!}(x - x_0)^n$$

và khi chia hai vế cho $(x - x_0)$ ta nhận được :

$$P_{n-1}(x) = \frac{P'(x_0)}{1!} + \frac{P''(x_0)}{2!}(x - x_0) + \cdots + \frac{P^{(n)}(x_0)}{2!}(x - x_0)^{n-1} \quad (5)$$

So sánh (4) và (5) ta nhận được kết quả :

$$P_{n-1}(x_0) = \frac{P'(x_0)}{1!}$$

Trong đó $P_{n-1}(x)$ lại có thể phân tích giống như $P_n(x)$ dạng (3) để tìm ra $P_{n-1}(x_o)$.Quá trình này được tiếp tục cho đến khi ta tìm hết các hệ số của chuỗi Taylor của $P_n(x)$

Tổng quát thuật toán thể hiện ở bảng sau :

$P_n(x)$	a_o	a_1	a_2	a_3	...	a_{n-1}	a_n
$x = x_o$	0	$b_o x_o$	$b_1 x_o$	$b_2 x_o$		$b_{n-2} x_o$	$b_{n-1} x_o$
$P_{n-1}(x)$	b_o	b_1	b_2	b_3	...	b_{n-1}	$b_n = P_n(x_o)$

Để hiểu rõ hơn chúng ta lấy một ví dụ cụ thể sau : Khai triển đa thức sau tại $x_0 = 2$

$P(x) = x^5 - 2x^4 + x^3 - 5x + 4$

Ta lập bảng tính sau :

	1	-2	1	0	-5	4
2	0	2	0	2	4	2
	1	0	1	2	-1	2 = P(2)/0!
2	0	2	4	10	24	
	1	2	5	12	23 = P'(2)/1!	
2	0	2	8	26		
	1	4	13	38 = P''(2)/2!		
2	0	2	12			
	1	6	25 = P'''(2)/3!			
2	0	2				
	1	8 = P''''(2)/4!				

2 0

1 = P''''(2)/4!

Như vậy :

$P_n(x) = (x-2)^5 + 8(x-2)^4 + 25(x-2)^3 + 38(x-2)^2 + 23(x-2) + 2$

Chương trình sau dùng để xác định các hệ số của chuỗi Taylor của đa thức P(x) tại $x_0 = 2$.

Chương trình 7-2

```
#include <conio.h>
#include <stdio.h>
#define  m  10

void main(void)
  {
    float a[m],b[m],c[m];
    int n,i,j,k;
    float x;

    clrscr();
    printf("Cho bac cua da thuc n = ");
    scanf("%d",&n);
    printf("Cho gia tri x = ");
    scanf("%f",&x);
    printf("Vao cac he so a\n");
    for (k=n;k>=0;k--)
      {
          printf("a[%d] = ",n-k);
          scanf("%f",&a[k]);
      }
    printf("\n");
    b[n] = a[n];
    c[n] = a[n];
    for (k=0;k<=n-1;k++)
      {
           for (i=n-1;i>=k;i--)
             b[i] = b[i+1]*x + a[i];
           c[k] = b[k];
           for (j=n;j>=k+1;j--)
             a[j] = b[j];
      }
    printf("\nSo do Horner tong quat");
    printf("\nKhai trien tai x = %.4f\n",x);
    for (k=n;k>=0;k--)
          printf("%10.4f\t",c[k]);
    getch();
  }
```

§3. CÁC PHÉP TÍNH TRÊN ĐA THỨC

1. Phép cộng hai đa thức : Giả sử chúng ta có hai đa thức A(x) bậc n và B(x) bậc m với n>m. Khi cộng hai đa thức này,chúng ta cộng lần lượt các hệ số cùng bậc của chúng với nhau.Ta có chương trình sau :

Chương trình 7-3

```
#include <conio.h>
#include <stdio.h>
#define t 10

void main(void)
  {
    int k,n,m;
    float a[t],b[t],c[t];

    clrscr();
    printf("Cho bac cua da thuc A n = ");
    scanf("%d",&n);
    printf("Vao cac he so a\n");
    for (k=1;k<=n+1;k++)
      {
                  printf("a[%d] = ",k-1);
                  scanf("%f",&a[k]);
      }
    printf("Cho bac cua da thuc B m = ");
    scanf("%d",&m);
         printf("Vao cac he so b\n");
    for (k=1;k<=m+1;k++)
      {
                 printf("b[%d] = ",k-1);
                 scanf("%f",&b[k]);
      }
    printf("\n");
    for (k=1;k<=n+1;k++)
    if (k<=n-m)
      c[k] = a[k];
    else
      c[k] = a[k] + b[k-n+m];
    printf("Cac he so cua da thuc tong C la :\n");
    for (k=1;k<=n+1;k++)
           printf("%.4f\t",c[k]);
         getch();
  }
```

2. Phép nhân hai đa thức : Để thấy rõ thuật toán xác định các hệ số của đa thức C(x) là kết quả của phép nhân hai đa thức A(x) và B(x) ta cho một ví dụ cụ thể :

$A(x) = a_o x^5 + a_1 x^4 + a_2 x^3 + a_3 x^2 + a_4 x + a_5$

$B(x) = b_o x^3 + b_1 x^2 + b_2 x + b_3$

$C(x) = A(x).B(x)$

$= a_o b_o x^8 + (a_o b_1 + a_1 b_o)x^7 + (a_o b_2 + a_1 b_1 + a_2 b_o)x^6 + (a_o b_3 + a_1 b_2 + a_2 b_1 + a_3 b_o)x^5 + (a_1 b_3 + a_2 b_2 + a_3 b_1 + a_4 b_o)x^4 + (a_2 b_3 + a_3 b_2 + a_4 b_1 + a_5 b_o)x^3 + (a_3 b_3 + a_4 b_2 + a_5 b_1)x^2 + a_5 b_2 x + a_5 b_3$

Các hệ số của đa thức kết quả là :

$C_o = a_o b_o$

$C_1 = a_o b_1 + a_1 b_o$

$C_2 = a_o b_2 + a_1 b_1 + a_2 b_o$

$C_3 = a_o b_3 + a_1 b_2 + a_2 b_1 + a_3 b_o$

$C_4 = a_1 b_3 + a_2 b_2 + a_3 b_1 + a_4 b_o$

$C_5 = a_2 b_3 + a_3 b_2 + a_4 b_1 + a_5 b_o$

$C_6 = a_3 b_3 + a_4 b_2 + a_5 b_1$

$C_7 = a_5 b_2$

$C_8 = a_5 b_3$

Ta nhận thấy là hệ số C_k của C(x) là tổng các tích các hệ số của đơn thức bậc i của A(x) và bậc (k-i) của B(x). Chỉ số i = 0 khi k <= m+1 và i = k+m khi k > m+1.Chỉ số j = k khi k <= n+1 và j = n +1 khi k > n + 1. Chương trình tính tích hai đa thức :

Chương trình 7-4

```
#include <conio.h>
#include <stdio.h>
#define t  10

void main()
  {
        int k,n,m,l,i,j,p;
        float a[t],b[t],c[2*t];

        clrscr();
        printf("Cho bac cua da thuc A n = ");
        scanf("%d",&n);
        printf("Vao cac he so a\n");
        for (k=1;k<=n+1;k++)
          {
                printf("a[%d] = ",k-1);
                scanf("%f",&a[k]);
          }
        printf("Cho bac cua da thuc B m = ");
        scanf("%d",&m);
        printf("Vao cac he so b\n");
        for (k=1;k<=m+1;k++)
          {
                printf("b[%d] = ",k-1);
                scanf("%f",&b[k]);
          }
        printf("\n");
        l=n+m;
```

```
        for (k=1;k<=l+1;k++)
          {
                if (k<=(n+1))
                  j=k;
                else
                  j=n+1;
                if (k<=(m+1))
                  p=1;
                else
                  p= k-m;
                c[k]=0;
                for (i=p;i<=j;i++)
                  c[k] = c[k] + a[i]*b[k-i+1];
          }
        printf("Cac he so cua da thuc tich C voi bac %d la :\n",l);
        for (k=1;k<=l+1;k++)
          printf("%.4f\t",c[k]);
        getch();
  }
```

3. Chia hai đa thức : Giả sử ta có hai đa thức là $A_n(x)$ và $B_m(x)$ với n ≥ m.Thương hai đa thức này là :

$$\frac{A_n(x)}{B_m(x)} = Q_{n-m}(x) + \frac{R_{m-1}(x)}{B_m(x)}$$

Chương trình sau thực hiện việc chia 2 đa thức :

Chương trình 7-5

```
#include <stdio.h>
#include <conio.h>
#include <math.h>
#define t 10
void main()
  {
        int k,n,m,l,i,j,jp;
        float a[t],b[t],q[t],r[t],epsi;

        clrscr();
        printf("Cho bac cua da thuc A n = ");
        scanf("%d",&n);
        printf("Vao cac he so a\n");
        for (k=1;k<=n+1;k++)
          {
                printf("a[%d] = ",k-1);
                scanf("%f",&a[k]);
          }
        printf("\n");
        printf("Cho bac cua da thuc B m = ");
        scanf("%d",&m);
```

```
        printf("Vao cac he so b\n");
        for (k=1;k<=m+1;k++)
          {
                printf("b[%d] = ",k-1);
                scanf("%f",&b[k]);
          }
        printf("\n");
        printf("Cho gia tri sai so epsilon epsi = ");
        scanf("%f",&epsi);
        if ((m+1)>1)
          {
                l=n-m+1;
                for (i=0;i<=t;i++)
                  r[i]=a[i];
                j=n;
                for (k=1;k<=l;k++)
                  {
                        q[k]=r[1]/b[1];
                        for (i=1;i<=j;i++)
                          if ((i<m+1))
                                r[i]=r[i+1]-q[k]*b[i+1];
                          else
                                r[i]=r[i+1];
                          j=j-1;
                  }
                while ((abs(r[i])<epsi)&&(j>0))
                  {
                        for (i=1;i<=j;i++)
                          r[i]=r[i+1];
                        j=j-1;
                  }
                if (abs(r[1])<epsi)
                  r[1]=0.0;
                jp=j+1;
        }
else
        {
                l=n+1;
                for (k=1;k<=l;k++)
                  q[k]=a[k]/b[1];
                jp=1;
                r[1]=0.0;
          }
        printf("\n");
        printf("Cac he so cua thuong Q(x) bac %d la : ",l);
        for (k=1;k<=l;k++)
          printf("%.3f\t",q[k]);
        printf("\n");
        printf("Cac he so cua phan du R(x) bac %d la : ",jp-1);
        for (k=1;k<=jp;k++)
```

```
        printf("%.3f",r[k]);
      getch();
}
```

CHƯƠNG 8 : GIẢI GẦN ĐÚNG PHƯƠNG TRÌNH ĐẠI SỐ VÀ SIÊU VIỆT

§1.KHÁI NIỆM CHUNG

Nếu phương trình đại số hay siêu việt khá phức tạp thì ít khi tìm được nghiệm đúng.Bởi vậy việc tìm nghiệm gần đúng và ước lượng sai số là rất cần thiết.

Ta xét phương trình :

$$f(x) = 0 \qquad (1)$$

với f(x) là hàm cho trước của biến x.Chúng ta cần tìm giá trị gần đúng của nghiệm của phương trình này.

Quá trình giải thường chia làm hai bước: bước sơ bộ và bước kiện toàn nghiệm.

Bước giải sơ bộ có 3 nhiệm vụ:vây nghiệm, tách nghiệm và thu hẹp khoảng chứa nghiệm.

Vây nghiệm là tìm xem các nghiệm của phương trình có thể nằm trên những đoạn nào của trục x.Tách nghiệm là tìm các khoảng chứa nghiệm soa cho trong mỗi khoảng chỉ có đúng một nghiệm.Thu hẹp khoảng chứa nghiệm là làm cho khoảng chứa nghiệm càng nhỏ càng tốt.Sau bước sơ bộ ta có khoảng chứa nghiệm đủ nhỏ.

Bước kiện toàn nghiệm tìm các nghiệm gần đúng theo yêu cầu đặt ra.

Có rất nhiều phương pháp xác định nghiệm của (1).Sau đây chúng ta xét từng phương pháp.

§2.PHƯƠNG PHÁP LẶP ĐƠN

Giả sử phương trình (1) được đưa về dạng tương đương :

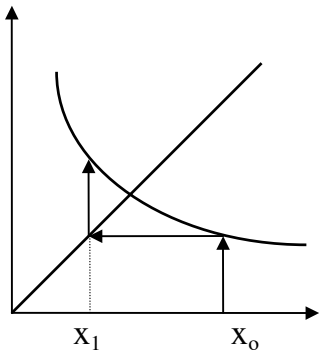
$$x = g(x) \qquad (2)$$

từ giá trị x_o nào đó gọi là giá trị lặp đầu tiên ta lập dãy xấp xỉ bằng công thức :

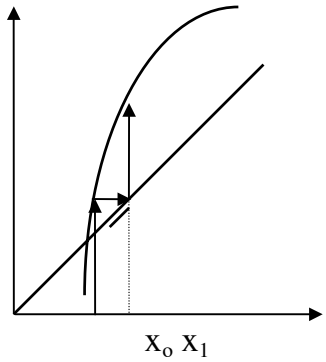
$$x_n = g(x_{n-1}) \qquad (3)$$

với n = 1,2,....

Hàm g(x) được gọi là hàm lặp.Nếu dãy $x_n \to \alpha$ khi $n \to \infty$ thì ta nói phép lặp (3) hội tụ.



Hình a



Hình b

Ta có định lí:Xét phương pháp lặp (3),giả sử :

- [a,b] là khoảng phân li nghiệm α của phương trình (1) tức là của (2)
- mọi x_n tính theo (3) đều thuộc [a,b]
- g(x) có đạo hàm thoả mãn :

$$|g'(x)| \le q < 1 \ , a < x < b \qquad (4)$$

trong đó q là một hằng số thì phương pháp lặp (3) hội tụ
Ta có thể minh hoạ phép lặp trên bằng hình vẽ a và b.

Cách đưa phương trình f(x) = 0 về dạng x = g(x) được thực hiện như sau:ta thấy f(x) = 0 có thể biến đổi thành x = x + λf(x) với λ ≠ 0.Sau đó đặt x + λf(x) = g(x) sao cho điều kiện (4) được thoả mãn.

Ví dụ:xét phương trình

$$x^3 + x - 1000 = 0$$

Sau bước giải sơ bộ ta có nghiệm $x_1 \in (9,10)$
Nếu đưa phương trình về dạng:

$$x = 1000 - x^3 = g(x)$$

thì dễ thấy $| g'(x) | > 1$ trong khoảng (9,10) nên không thoả mãn điều kiện (4)
Chúng ta đưa phương trình về dạng

$$x = \sqrt[3]{1000 - x}$$

thì ta thấy điều kiện (4) được thoả mãn.Xây dựng dãy xấp xỉ

$$x_{n+1} = \sqrt[3]{1000 - x_n}$$

với x_o chọn bất kì trong (9,10)

Trên cơ sở phương pháp này chúng ta có các chương trình tính toán sau:
Chương trình giải phương trình exp((1/3)*ln(1000-x)) với số lần lặp cho trước

Chương trình 8-1

```
//lap don
#include <conio.h>
#include <stdio.h>
#include <math.h>

void main()
 {
        int i,n;
        float x,x0;
        float f(float);

        clrscr();
        printf("Cho so lan lap n = ");
        scanf("%d",&n);
        printf("Cho gia tri ban dau cua nghiem x0 = ");
        scanf("%f",&x0);
        x=x0;
        for (i=1;i<=n;i++)
          x=f(x);
        printf("Nghiem cua phuong trinh la :%.4f",x);
        getch();
 }

float f(float x)
 {
        float a=exp((1./3.)*log(1000-x));
        return(a);
```

```
  }
```

và chương trình giải bài toán bằng phương pháp lặp với sai số cho trước

Chương trình 8-2

```
//lap don
#include <conio.h>
#include <stdio.h>
#include <math.h>
void main()
  {
        int i;
        float epsi,x,x0,y;
        float f(float);

        clrscr();
        printf("Cho sai so epsilon = ");
        scanf("%f",&epsi);
        printf("Cho gia tri ban dau cua nghiem x0 = ");
        scanf("%f",&x0);
        x=x0;
        y=f(x);
        if (abs(y-x)>epsi)
          {
                x=y;
                y=f(x);
          }
        printf("Nghiem cua phuong trinh la %.6f",y);
        getch();
  }

float f(float x)
  {
        float a=exp((1./3.)*log(1000-x));
        return(a);
  }
```

Cho giá trị đầu x_o = 1.Kết quả tính toán x = 9.966555

§3.PHƯƠNG PHÁP CHIA ĐÔI CUNG

Giả sử cho phương trình f(x) = 0 với f(x) liên tục trên đoạn [a,b] và f(a).f(b) < 0.Chia đoạn [a,b] thành 2 phần bởi chính điểm chia (a + b)/2.

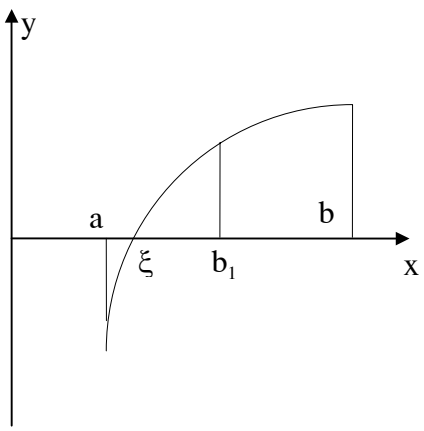
1.Nếu f((a+b)/2) = 0 thì ξ = (a+b)/2

2.Nếu f((a+b)/2) ≠ 0 thì chọn [a,(a + b)/2] hay [(a + b)/2,b] mà giá trị hàm hai đầu trái dấu và kí hiệu là $[a_1,b_1]$.Đối với $[a_1,b_1]$ ta lại tiến hành như [a,b]

Cách làm trên được mô tả trong chương trình sau dùng để tìm nghiệm của phương trình :

$$x^4 + 2x^3 - x - 1 = 0$$

trên đoạn [0,1]



Chương trình 8-3

```
//chia doi cung
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define  epsi  0.00001
void main()
 {
       float x0,x1,x2;
       float y0,y1,y2;
  float f(float);
       int maxlap,demlap;

  clrscr();
       printf("Tim nghiem cua phuong trinh phi tuyen");
       printf("\nbang cach chia doi cung\n");
       printf("Cho cac gia tri x0,x1,maxlap\n");
       printf("Cho gia tri  x0 = ");
       scanf("%f",&x0);
       printf("Cho gia tri  x1 =  ");
       scanf("%f",&x1);
       printf("Cho so lan lap maxlap =  ");
       scanf("%d",&maxlap);
       y0=f(x0);
       y1=f(x1);
       if ((y0*y1)>0)
         {
              printf("Nghiem khong nam trong doan x0 - x1\n");
              printf(" x0 = %.2f\n",x0);
              printf(" x1 = %.2f\n",x1);
              printf(" f(x0) = %.2f\n",y0);
              printf(" f(x1) = %.2f\n",y1);
         }
       demlap=0;
       do
         {
```

```
            x2=(x0+x1)/2;
            y2=f(x2);
            y0=f(x0);
            if (y0*y2>0)
              x0=x2;
            else
              x1=x2;
            demlap=demlap+1;
    }
        while(((abs((y2-y0))>epsi)||(demlap<maxlap)));
        if (demlap>maxlap)
          {
            printf("Phep lap khong hoi tu sau %d lan lap ",maxlap);
            printf(" x0 = %.2f\n",x0);
            printf(" x1 = %.2f\n",x1);
            printf(" f(x2) = %.2f\n",y2);
          }
        else
          {
            printf("Phep lap hoi tu sau %d lan lap\n",demlap);
            printf("Nghiem x = %.2f",x2);
          }
        getch();
  }

float f(float x)
  {
        float a=x*x*x*x+2*x*x*x-x-1 ;
        return(a);
  }
```

Kết quả tính cho nghiệm:x = 0.87

§4.PHƯƠNG PHÁP DÂY CUNG

Giả sử f(x) liên tục trên trên đoạn [a,b] và f(a).f(b) < 0.Cần tìm nghiệm của f(x) = 0.Để xác định ta xem f(a) < 0 và f(b) > 0.Khi đó thay vì chia đôi đoạn [a,b] ta chia [a,b] theo tỉ lệ -f(a)/f(b).Điều đó cho ta nghiệm gần đúng :

$$x_1 = a + h_1$$

Trong đó

$$h_1 = \frac{-f(a)}{-f(a)+f(b)}(b-a)$$

Tiếp theo dùng cách đó với đoạn [a,x_1] hay [x_1,b] mà hai đầu hàm nhận giá trị trái dấu ta được nghiệm gần đúng x_2 v.v.

Về mặ hình học,phương pháp này có nghĩa là kẻ dây cung của đường cong f(x) qua hai điểm A[a,f(a)] và B[b,f(b)].Thật vậy phương trình dây cung AB có dạng :

$$\frac{x-a}{b-a}=\frac{y-f(a)}{f(b)-f(a)}$$

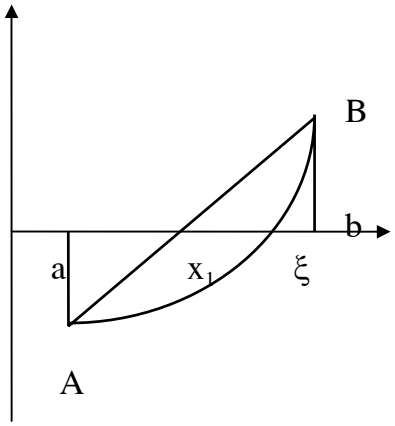
Cho $x = x_1$ $y = 0$ ta có

$$x_1 = a - \frac{f(a)}{f(b)-f(a)}(b-a)$$

Trên cơ sở của phương pháp ta có chương trình tính nghiệm của phương trình

$$x^4 + 2x^3 - x - 1 = 0$$

trên đoạn [0,1]



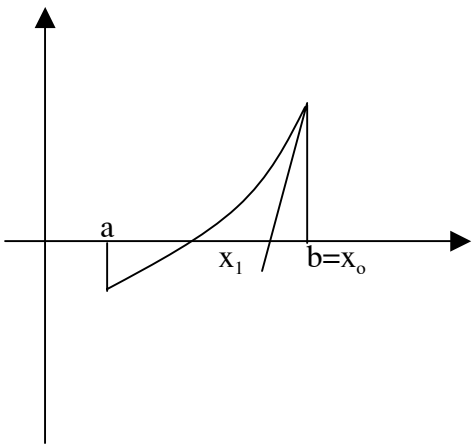
Chương trình 8-4

```
//phuong phap day cung
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define epsi  0.00001

void main()
 {
        float a,b,fa,fb,dx,x;
        float f(float);

        clrscr();
        printf("Tim nghiem cua phuong trinh phi tuyen\n");
        printf("bang phuong phap day cung\n");
        printf("Cho cac gia tri a,b\n");
        printf("Cho gia tri cua a = ");
        scanf("%f",&a);
        printf("Cho gia tri cua b = ");
        scanf("%f",&b);
        fa=f(a);
        fb=f(b);
        dx=fa*(b-a)/(fa-fb);
        while (fabs(dx)>epsi)
         {
                x=a+dx;
                fa=f(x);
                if((fa*fb)<=0)
                  a=x;
                else
                  b=x;
                fa=f(a);
                fb=f(b);
                dx=fa*(b-a)/(fa-fb);
         }
```

```
        printf("Nghiem x = %.3f",x);
        getch();
 }

float f(float x)
 {
        float e=x*x*x*x+2*x*x*x-x-1;
        return(e);
 }
```

Kết quả tính cho nghiệm:x = 0.876

§5.PHƯƠNG PHÁP LẶP NEWTON



Phương pháp lặp Newton (còn gọi là phương pháp tiếp tuyến) được dùng nhiều vì nó hội tụ nhanh.Giả sử f(x) có nghiệm là ξ đã được tách trên đoạn [a,b] đồng thời f'(x) và f''(x) liên tục và giữ nguyên dấu trên đoạn [a,b].Khi đã tìm được xấp xỉ nào đó $x_n \in [a,b]$ ta có thể kiện toàn nó theo phương pháp Newton.Từ mút B ta vẽ tiếp tuyến với đường cong.Phương trình đường tiếp tuyến là

$$y - f(x_0) = f'(b)(x - x_0)$$

Tiếp tuyến này cắt trục hoành tại điểm có y=0,nghĩa là :

$$-f(x_0) = f'(b)(x_1 - x_0)$$

hay :

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Từ x_1 ta lại tiếp tục vẽ tiếp tuyến với đường cong thì giao điểm x_i sẽ tiến tới nghiệm của phương trình.

Việc chọn điểm ban đầu x_o rất quan trọng.Trên hình vẽ trên ta thấy nếu chọn điểm ban đầu x_o = a thì tiếp tuyến sẽ cắt trục tại một điểm nằm ngoài đoạn [a,b].Chọn x_o = b sẽ thuận lợi cho việc tính toán.Chúng ta có định lí :

Nếu f(a).f(b) < 0 ; f(x) và f''(x) khác không và giữ nguyên dấu xác định khi x ∈ [a,b] thì xuất phát từ $x_o \in [a,b]$ thoả mãn điều kiện $f(x_o).f''(x_o) > 0$ có thể tính theo phương pháp Newton nghiệm ξ duy nhất với độ chính xác tuỳ ý.

Khi dùng phương pháp Newton cần lấy x_o là đầu mút của đoạn [a,b] để tại đó $f(x_o).f''(x_o) > 0$.Áp dụng lí thuyết trên chúng ta xây dựng chương trình tính sau:

Chương trình 8-5

```
//phuong phap Newton
#include <conio.h>
#include <stdio.h>
```

```
#include <math.h>
#include <stdlib.h>
#define n 50
#define epsi 1e-5

void main()
  {
        float t,x0;
        float x[n];
        int i;
        float f(float);
        float daoham(float);

        clrscr();
        printf("Tim nghiem cua phuong trinh phi tuyen\n");
        printf("bang phuong phap lap Newton\n");
        printf("Cho gia tri cua x0 = ");
        scanf("%f",&x0);
        i=1;
        x[i]=x0;
        do
          {
                x[i+1] = x[i]-f(x[i])/daoham(x[i]);
                t = fabs(x[i+1]-x[i]);
                x[i]=x[i+1];
                i=i+1;
                if (i>100)
                  {
                        printf("Bai toan khong hoi tu\n");
                        getch();
                        exit(1);
                        }
                else
        ;
          }
        while (t>=epsi);
        printf("Nghiem x = %.5f",x[i]);
        getch();
  }

float f(float x)
  {
        float a=x*x-x-2;
        return(a);
  }

float daoham(float x)
  {
         float d=2*x-1;
         return(d);
```

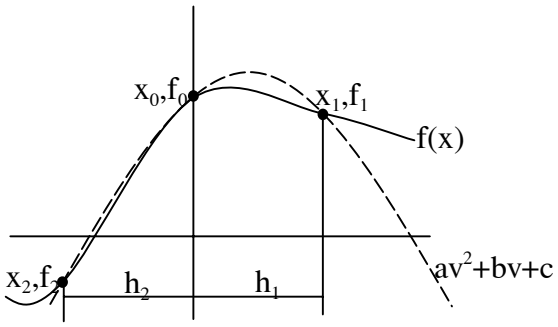
}

Chương trình này được dùng xác định nghiệm của hàm đã được định nghĩa trong function.Trong trường hợp này phương trình đó là:x^2 - x -1 = 0.Kết quả tính với giá trị đầu x_o = 0 cho nghiệm x = 2.

§6.PHƯƠNG PHÁP MULLER

Trong phương pháp dây cung khi tìm nghiệm trong đoạn [a,b] ta xấp xỉ hàm bằng một đường thẳng.Tuy nhiên để giảm lượng tính toán và để nghiệm hội tụ nhanh hơn ta có thể dùng phương pháp Muller.Nội dung của phương pháp này là thay hàm trong đoạn [a,b] bằng một đường cong bậc 2 mà ta hoàn toàn có thể tìm nghiêm chín xác của nó.Gọi các điểm đó có hoành độ lần lượt là $a = x_2, b = x_1$ và ta chọn thêm một điểm x_0 nằm trong đoạn $[x_2,x_1]$.Gọi

$$h_1 = x_1 - x_0$$
$$h_2 = x_0 - x_2$$
$$v = x - x_0$$
$$f(x_0) = f_0$$
$$f(x_1) = f_1$$
$$f(x_2) = f_2$$
$$\gamma = \frac{h_2}{h_1}$$



Qua 3 điểm này ta có một đường parabol :

$$y = av^2 + bv + c$$

Ta tìm các hệ số a,b,c từ các giá trị đã biết v:

$$v = 0 (x = x_0) \quad a(0)^2 + b(0) + c = f_0$$
$$v = h_1 (x = x_1) \quad ah_1^2 + bh_1 + c = f_1$$
$$v = h_2 (x = x_2) \quad ah_2^2 + bh_2 + c = f_2$$

Từ đó ta có :

$$a = \frac{\gamma f_1 - f_0(1+\gamma) + f_2}{\gamma h_1^2 (1+\gamma)}$$

$$b = \frac{f_1 - f_0 - ah_1^2}{h_1}$$

$$c = f_0$$

Sau đó ta tìm nghiệm của phương trình $av^2 + bv + c = 0$ và có :

$$n_{1,2} = x_0 - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}$$

Tiếp đó ta chọn nghiệm gần x_0 nhất làm một trong 3 điểm để tính xấp xỉ mới.Các điểm này được chọn gần nhau nhất.

Tiếp tục quá trình tính đến khi đạt độ chính xác yêu cầu thì dừng lại.

Ví dụ:Tìm nghiệm của hàm f(x) = sin(x) - x/2 trong đoạn [1.8,2.2].Ta chọn $x_0 = 2$

Ta có : $x_0 = 2$ $f(x_0) = -0.0907$ $h_1 = 0.2$

$x_1 = 2.2$ $f(x_1) = -0.2915$ $h_2 = 0.2$

$x_2 = 1.8$ $f(x_2) = 0.07385$ $\gamma = 1$

Vậy thì :

$$a = \frac{1\times(-0.2915)-(-0.0907)\times(1+1)+0.07385}{1\times 0.2^2\times(1+1)} = -0.45312$$

$$b = \frac{-0.2915-(-0.097)-(-0.45312)\times 0.2^2}{0.2} = -0.91338$$

$$c = -0.0907$$

Ta có nghiệm gần x_0 nhất là :

$$n_1 = 2.0 - \frac{2\times(-0.0907)}{-0.91338-\sqrt{(-0.91338)^2-4\times(-0.45312)\times(-0.0907)}} = 1.89526$$

Với lần lặp thứ hai ta có :

$x_0 = 1.89526$ $f(x_0) = 1.9184\times10^{-4}$ $h_1 = 0.10474$

$x_1 = 2.0$ $f(x_1) = -0.0907$ $h_2 = 0.09526$

$x_2 = 1.8$ $f(x_2) = 0.07385$ $\gamma = 0.9095$

Vậy thì :

$$a = \frac{0.9095\times(-0.0907)-(1.9184\times10^{-4})\times 1.9095+0.07385}{0.9095\times 0.10474^2\times 1.9095} = -0.4728$$

$$b = \frac{-0.0907-1.9184\times10^{-4}-(-0.4728)\times 0.10474^2}{0.10474} = -0.81826$$

$$c = 1.9184\times10^{-4}$$

Ta có nghiệm gần x_0 nhất là :

$$n_1 = 1.89526 - \frac{2\times 1.9184\times10^{-4}}{-0.81826-\sqrt{(0.81826)^2-4\times(-0.4728)\times 1.9184\times10^{-4}}} = 1.89594$$

Ta có thể lấy $n_1 = 1.895494$ làm nghiệm của bài toán

Chương trình giải bài toán bằng phương pháp Muller như sau:

Chương trình 8-6

```
//phuong phap Muller
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

void main()
 {
        float x0,x1,x2,h1,h2,eps;
        float a,b,c,gamma,n1,n2,xr;
        int dem;
        float f(float);

        clrscr();
        printf("PHUONG PHAP MULLER\n");
        printf("\n");
        printf("Cho khoang can tim nghiem [a,b]\n");
        printf("Cho gia tri duoi a = ");
        scanf("%f",&x2);
```

```
        printf("Cho gia tri tren b = ");
        scanf("%f",&x1);
        if (f(x1)*f(x2)>0)
          {
                printf("\n");
                printf("Nghiem khong nam trong doan nay\n");
                getch();
                exit(1);
          }
        eps=1e-5;
        x0=(x1+x2)/2;
        dem=0;
        do
          {
                dem=dem+1;
                h1=x1-x0;
                h2=x0-x2;
                gamma=h2/h1;
                a=(gamma*f(x1)-
f(x0)*(1+gamma)+f(x2))/(gamma*(h1*h1)*(1+gamma));
                b=(f(x1)-f(x0)-a*(h1*h1))/h1;
                c=f(x0);
                if ((a==0)&&(b!=0))
                  {
                        n1=-c/b;
                        n2=n1;
                  }
                if ((a!=0)&&(b==0))
                  {
                        n1=(-sqrt(-c/a));
                        n2=(sqrt(-c/a));
                  }
                if ((a!=0)&&(b!=0))
                  {
                        n1=x0-2*c/(b+(sqrt(b*b-4*a*c)));
                        n2=x0-2*c/(b-(sqrt(b*b-4*a*c)));
                  }
                if (fabs(n1-x0)>fabs(n2-x0))
                  xr=n2;
                else
                  xr=n1;
                if (xr>x0)
                  {
                        x2=x0;
                        x0=xr;
                  }
                else
                  {
                        x1=x0;
                        x0=xr;
```

```
                }
            }
        while (fabs(f(xr))>=eps);
        printf("\n");
        printf("Phuong trinh co nghiem x = %.5f sau %d lan lap",xr,dem);
        getch();
}

float f(float x)
 {
        float a=sin(x)-x/2;
        return(a);
 }
```

§7.PHƯƠNG PHÁP LẶP BERNOULLI

Có nhiều phương pháp để tìm nghiệm của một đa thức.Ta xét phương trình :

$$a_o x^n + a_1 x^{n-1} + \cdots + a_n = 0$$

Nghiệm của phương trình trên thoả mãn định lí:*Nếu max{| a_1 |,| a_2 |,...,| a_n |} = A thì các nghiệm của phương trình thoả mãn điều kiện | x | < 1 + A/ | a_0 |*

Phương pháp Bernoulli cho phép tính toán nghiệm lớn nhất α của một đa thức $P_n(x)$ có n nghiệm thực phân biệt.Sau khi tìm được nghiệm lớn nhất α ta chia đa thức $P_n(x)$ cho $(x - \alpha)$ và nhận được đa thức mới $Q_{n-1}(x)$.Tiếp tục dùng phương pháp Bernoulli để tìm nghiệm lớn nhất của $Q_{n-1}(x)$.Sau đó lại tiếp tục các bước trên cho đến khi tìm hết các nghiệm của $P_n(x)$.

Chúng ta khảo sát phương trình phương trình sai phân φ có dạng như sau :

$$\varphi = a_o y_{k+n} + a_1 y_{k+n-1} + \ldots\ldots + a_n y_k = 0 \qquad (1)$$

Đây là một phương trình sai phân tuyên tính hệ số hằng.Khi cho trước các giá trị đầu $y_o, y_1, .. y_{n-1}$ ta tìm được các giá trị y_n, y_{n+1},..Chúng được gọi là nghiệm của phương trình sai phân tuyến tính (1).

Đa thức

$$P_n(x) = a_0 x^n + a_1 x^{n-1} + .. + a_{n-1} x + a_n \qquad (2)$$

với cùng một hệ số a_i như (1) được gọi là đa thức đặc tính của phương trình sai phân tuyến tính (1).Nếu (2) có n nghiệm phân biệt $x_1, x_2, .., x_n$ thì (1) có các nghiệm riêng là

$$y_i = x_i^k$$

Nếu y_i là các nghiệm của phương trình sai phân là tuyến tính (1),thì

$$y_k = c_1 x_1^k + c_2 x_2^k + \ldots + c_n x_n^k \qquad (3)$$

với các hệ số c_i bất kì cũng là nghiệm của phương trình sai phân tuyến tính hệ số hằng (1).

Nếu các nghiệm là sao cho :

$$| x_1 | \geq | x_2 | \geq \ldots | x_n |$$

Vậy thì

$$y_k = c_1 x_1^k [1 + \frac{c_1}{c_2} (\frac{x_2}{x_1})^k + \ldots]$$

và

$$y_{k+1} = c_1 x_1^{k+1} [1 + \frac{c_1}{c_2} (\frac{x_2}{x_1})^{k+1} + \ldots]$$

do đó : $$\frac{y_{k+1}}{y_k}=x_1\frac{[1+\frac{c_2}{c_1}(\frac{x_2}{x_1})^{k+1}+...]}{[1+\frac{c_2}{c_1}(\frac{x_2}{x_1})^{k}+...]}$$

do $x_1 > x_2$

nên: $(\frac{x_2}{x_1})^k,(\frac{x_2}{x_1})^k.....\to 0$ khi $k\to\infty$

vậy thì : $\frac{y_{k+1}}{y_k}\to\infty$ khi $k\to\infty$

Nghĩa là : $x_1=\lim_{k\to\infty}\frac{y_{k+1}}{y_k}$

Nếu phương trình vi phân gồm n+1 hệ số,một nghiệm riêng y_k có thể được xác định từ n giá trị $y_{k-1},y_{k-2},...,y_{n-1}$.Điều cho phép tính toán bằng cách truy hồi các nghiệm riêng của phương trình vi phân.

Để tính nghiệm lớn nhất của đa thức,ta xuất phát từ các nghiệm riêng $y_1 = 0,y_1 = 0,..,y_n = 1$ để tính y_{n+1}.Cách tính này được tiếp tục để tính y_{n+2} xuất phát từ $y_1 = 0,y_2 = 0,..,y_{n+1}$ và tiếp tục cho đến khi y_{k+1}/y_k không biến đổi nữa.Trị số của y_{k+n} được tính theo công thức truy hồi :

$$y_{k+n}=-\frac{1}{a_o}(a_1y_{k+n-1}+...+a_ny_k) \qquad (4)$$

Ví dụ:Tính nghiệm của đa thức $P_n(x) = P_3(x) = x^3 - 10x^2 + 31x - 30$.Như vậy $a_o = 1,a_1 = -10,a_2 = 31$ và $a_3 = -30$.Phương trình sai phân tương ứng là :

$y_{k+3} -10y_{k+2} + 31y_{k+1} - 30y_k = 0$

Ta cho trước các giá trị $y_1 = 0$; $y_2 = 0$ và $y_3 = 1$.Theo (4) ta tính được :

$y_4 = - (-10y_3 + 31y_2 - 30y_1) = 10$
$y_5 = - (-10y_4 + 31y_3 - 30y_2) = 69$
$y_6 = - (-10y_5 + 31y_5 - 30y_3) = 410$
$y_7 = - (-10y_6 + 31y_5 - 30y_4) = 2261$
$y_8 = - (-10y_7 + 31y_6 - 30y_5) = 11970$
$y_9 = - (-10y_8 + 31y_7 - 30y_6) = 61909$
$y_{10} = - (-10y_9 + 31y_8 - 30y_8) = 315850$
$y_{11} = - (-10y_{10} + 31y_9 - 30y_8) = 1598421$
$y_{12} = - (-10y_{11} + 31y_{10} - 30y_9) = 8050130$
$y_{13} = - (-10y_{12} + 31y_{11} - 30y_{10}) = 40425749$
$y_{14} = - (-10y_{13} + 31y_{12} - 30y_{11}) = 202656090$
$y_{15} = - (-10y_{14} + 31y_{13} - 30y_{12}) = 1014866581$
$y_{16} = - (-10y_{15} + 31y_{14} - 30y_{13}) = 5079099490$
$y_{17} = - (-10y_{16} + 31y_{15} - 30y_{14}) = 24409813589$
$y_{18} = - (-10y_{17} + 31y_{16} - 30y_{15}) = 127092049130$
$y_{19} = - (-10y_{18} + 31y_{17} - 30y_{16}) = 635589254740$

Tỉ số các số y_{k+1}/y_k lập thành dãy :

10 ; 6.9 ; 5.942 ; 5.5146 ; 5.2941 ; 5.172 ; 5.1018 ; 5.0607 ; 5.0363 ; 5.0218 ; 5.013 ; 5.0078 ; 5.0047 ; 5.0028 ; 5.0017 ; 5.001

nghĩa là chúng sẽ hội tụ tới nghiệm lớn nhất là 5 của đa thức

Chương trình 8-7

```
//phuong phap Bernoulli
#include <conio.h>
#include <stdio.h>
```

```
#include <math.h>
#include <stdlib.h>
#define max  50

void main()
  {
        float a[max],y[max];
        int k,j,i,n,l;
        float s,e1,e2,x0,x1,x;

        clrscr();
        printf("Cho bac cua da thuc can tim nghiem n = ");
        scanf("%d",&n);
        e1=1e-5;
        printf("Cho cac he so cua da thuc can tim nghiem\n");
        for (i=0;i<=n;i++)
          {
                printf("a[%d] = ",i);
                scanf("%f",&a[i]);
          }
        for (k=0;k<=n;k++)
          a[k]=a[k]/a[0];
          tt: x1=0;
        for (k=2;k<=n;k++)
          y[k]=0;
        y[1]=1;
        l=0;
        do
          {
                l=l+1;
                s=0;
                for (k=1;k<=n;k++)
                  s=s+y[k]*a[k];
                y[0]=-s;
                x=y[0]/y[1];
                e2=fabs(x1 - x);
                x1=x;
                for (k=n;k>=1;k--)
                y[k]=y[k-1];
          }
        while((l<=50)||(e2>=e1));
        if(e2>=e1)
          {
                printf("Khong hoi tu");
                getch();
                exit(1);
          }
        else
          printf("Nghiem x = %.4f\n",x);
        n=n-1;
```

```
        if (n!=0)
          {
                a[1]=a[1]+x;
                for (k=2;k<=n;k++)
                  a[k]=a[k]+x*a[k-1];
                goto tt;
          }
        getch();
}
```

Kết quả nghiệm của đa thức $x^3 - 10x^2 + 31x - 30$ là:5 ; 3 và 2

§8.PHƯƠNG PHÁP LẶP BIRGE - VIETTE

Các nghiệm thực,đơn giản của một đa thức $P_n(x)$ được tính toán khi sử dụng phương pháp Newton

$$x_{i+1} = x_i - \frac{P_n(x_i)}{P'_n(x_i)} \qquad (1)$$

Để bắt đầu tính toán cần chọn một giá trị ban đầu x_o.Chúng ta có thể chọn một giá trị x_o nào đó,ví dụ :

$$x_o = -\frac{a_n}{a_{n-1}}$$

và tính tiếp các giá trị sau :

$$x_1 = x_o - \frac{P_n(x_o)}{P'_n(x_o)}$$

$$x_2 = x_1 - \frac{P_n(x_1)}{P'_n(x_1)}$$

Tiếp theo có thể đánh giá $P_n(x_i)$ theo thuật toán Horner :

$$P_0 = a_0$$
$$P_1 = P_0x_i + a_1 \qquad (2)$$
$$P_2 = P_1x_i + a_2$$
$$P_3 = P_2x_i + a_3$$
$$.....$$
$$P(x_i) = P_n = P_{n-1}x_i + a_n$$

Mặt khác khi chia đa thức $P_n(x)$ cho một nhị thức $(x - x_i)$ ta được :

$$P_n(x) = (x - x_i)P_{n-1}(x) + b_n \qquad (3)$$

với $b_n = P_n(x_i)$.Đa thức $P_{n-1}(x)$ có dạng :

$$P_{n-1}(x) = b_ox^{n-1} + b_1x^{n-2} + p_3x^{n-3} + ..+ b_{n-2}x + b_{n-1} \qquad (4)$$

Để xác định các hệ số của đa thức (4) ta thay (4) vào (3) và cân bằng các hệ số với đa thức cần tìm nghiệm $P_n(x)$ mà các hệ số a_i đã cho:

$$(x - x_i)(b_ox^{n-1} + b_1x^{n-2} + b_3x^{n-3} + ..+ b_{n-2}x + b_{n-1}) + b_n$$
$$= a_ox^n + a_1x^{n-1} + a_2x^{n-2} + ...+ a_{n-1}x + a_n \qquad (5)$$

Từ (5) rút ra :

$$b_o = a_o$$
$$b_1 = a_1 + b_ox_i \qquad (6)$$
$$b_2 = a_2 + b_1x_i$$
$$.....$$
$$b_k = a_k + b_{k-1}x_i$$
$$.....$$

$b_n = a_n + b_{n-1}x_i = P_n(x_i)$

Đạo hàm (3) ta được :

$$P'_n(x) = (x - x_i)P'_{n-1}(x) + P_{n-1}(x)$$

và

$$P'_n(x_i) = P_{n-1}(x_i) \qquad (7)$$

Như vậy với một giá trị x_i nào đó theo (2) ta tính được $P_n(x_i)$ và kết hợp (6) với (7) tính được $P'_n(x_i)$.Thay các kết quả này vào (1) ta tính được giá trị x_{i+1}.Quá trình được tiếp tục cho đến khi $| x_{i+1} - x_i | < \varepsilon$ hay $P_n(x_{i+1}) \approx 0$ nên $\alpha_1 \approx x_{i+1}$ là một nghiệm của đa thức.

Phép chia $P_n(x)$ cho $(x - \alpha_1)$ cho ta $P_{n-1}(x)$ và một nghiệm mới khác được tìm theo cách trên khi chọn một giá trị x_o mới hay chọn chính $x_o = \alpha_1$.Khi bậc của đa thức giảm xuống còn bằng 2 ta dùng các công thức tìm nghiệm của tam thức để tìm các nghiệm còn lại.

Ví dụ:tìm nghiệm của đa thức $P_3(x) = x^3 - x^2 - 16x + 24$

$a_o = 1$ $a_1 = -1$ $a_2 = -16$ $a_3 = 24$

Chọn $x_o = 3.5$ ta có :

$P_o = a_o = 1$

$P_1 = a_1 + p_o x_0 = -1 + 3.5*1 = 2.5$

$P_2 = a_2 + p_1 x_0 = -16 + 3.5*2.5 = -7.25$

$P_3 = a_3 + p_2 x_0 = 24 + 3.5*(-7.25) = -1.375$

$b_0 = a_0 = 1;$

$b_1 = a_1 + b_o x_0 = -1 + 3.5*1 = 2.5$

$b_2 = a_2 + b_1 x_0 = -16 + 3.5*2.5 = -7.25$

$P_2(3.5) = b_0 x^2 + b_1 x + b_2 = 13.75$

$$x_1 = x_0 - \frac{P_n(x_0)}{P'_n(x_0)} = 3.5 + \frac{1.375}{13.75} = 3.6$$

Lặp lại bước tính trên cho x_1 ta có:

$P_o = a_o = 1$

$P_1 = a_1 + p_o x_1 = -1 + 3.6*1 = 2.6$

$P_2 = a_2 + p_1 x_1 = -16 + 3.6*2.6 = -6.64$

$P_3 = a_3 + p_2 x_1 = 24 + 3.6*(-6.64) = -0.096$

$b_o = a_o = 1$

$b_1 = a_1 + b_o x_1 = -1 + 3.6*1 = 2.6$

$b_2 = a_2 + p_1 x_1 = -16 + 3.6*2.6 = -6.64$

$P_2(3.6) = b_0 x^2 + b_1 x + b_2 = 15.68$

$$x_2 = x_1 - \frac{P_n(x_1)}{P'_n(x_1)} = 3.6 + \frac{0.096}{15.68} = 3.606$$

Quá trình cứ thế tiếp tục cho đến khi sai số chấp nhận được.Chương trình dưới đây mô tả thuật tính trên.

Chương trình 8-8

```
//phuong phap Birge-Viette
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define  max  20
```

```
void main()
 {
        float a[max],p[max],d[max],x[max];
        int k,j,i,n;
        float e1,e2,x0,x1;

        clrscr();
        printf("Cho bac cua da thuc n = ");
        scanf("%d",&n);
        e1=0.0001;
        printf("Cho cac he so cua da thuc can tim nghiem\n");
        for (i=0;i<=n;i++)
          {
                printf("a[%d] = ",i);
                scanf("%f",&a[i]);
          }
        x0=a[0];
        for (i=0;i<=n;i++)
          a[i]=a[i]/x0;
        printf("Nghiem cua phuong trinh  : \n");
        tt:x0=-a[n]/a[n-1];
        j=0;
        do
          {
                j=j+1;
                p[1]=x0+a[1];
                d[1]=1.0;
                for (k=2;k<=n;k++)
                  {
                        p[k]=p[k-1]*x0+a[k];
                        d[k]=d[k-1]*x0+p[k-1];
                  }
                x1=x0-p[n]/d[n];
                e2=fabs(x1-x0);
                if (e2>e1)
                x0=x1;
          }
        while((j<=50)||(e2>=e1));
        if (e2>=e1)
          printf("Khong hoi tu");
        else
          printf(" x = %.4f\n",x1);
        n=n-1;
        if (n!=0)
          {
                for (k=1;k<=n;k++)
                  a[k]=p[k];
                goto tt;
          }
```

```
    getch();
}
```

Dùng chương trình trên để tìm nghiệm của đa thức $x^4 + 2x^3 - 13x^2 - 14x + 24$ ta được các nghiệm là:-4 ; 3 ; -2 và 1.

§9.PHƯƠNG PHÁP NGOẠI SUY AITKEN

Xét phương pháp lặp :

$$x = f(x) \quad (1)$$

với f(x) thoả mãn điều kiện hội tụ của phép lặp,nghĩa là với mọi $x \in [a,b]$ ta có :

$$| f'(x) | \le q < 1 \quad (2)$$

Như vậy :

$$x_{n+1} = f(x_n) \quad (3)$$

$$x_n = f(x_{n-1}) \quad (4)$$

Trừ (3) cho (4) và áp dụng định lí Lagrange cho vế phải với $c \in [a,b]$ ta có :

$$x_{n+1} - x_n = f(x_n) - f(x_{n-1}) = (x_n - x_{n-1})f'(c) \quad (5)$$

Vì phép lặp (1) nên :

$$| x_{n+1} - x_n | \le q | x_n - x_{n-1} | \quad (6)$$

Do (6) đúng với mọi n nên cho n = 1 , 2 , 3 , . . . ta có :

$$|x_2 - x_1| \le q|x_1 - x_o|$$
$$|x_3 - x_2| \le q|x_2 - x_1|$$
. . . . . .
$$|x_{n+1} - x_n| \le q|x_n - x_{n-1}|$$

Điều này có nghĩa là dãy $x_{i+1} - x_i$, một cách gần đúng,là một cấp số nhân . Ta cũng coi rằng dãy $x_n - y$ với y là nghiệm đúng của (1) , gần đúng như một cấp số nhân có công sai q . Như vậy :

$$\frac{x_{n+1} - y}{x_n - y} = q < 1 \qquad (7)$$

hay : $$x_{n+1} - y = q(x_n - y) \qquad (8)$$

Tương tự ta có : $$x_{n+2} - y = q(x_{n+1} - y) \qquad (9)$$

Từ (8) và (9) ta có :

$$q = \frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n} \qquad (10)$$

Thay giá trị của q vừa tính ở (10) vào biểu thức của q ở trên ta có :

$$y = x_n - \frac{(x_n - x_{n+1})^2}{x_n - 2x_{n+1} + x_{n+1}} \qquad (11)$$

Công thức (11) được gọi là công thức ngoại suy Adam.Như vậy theo (11) trước hết ta dùng phương pháp lặp để tính giá trị gần đúng x_{n+2},x_{n+1},x_n của nghiệm và sau đó theo (11) tìm được nghiệm với sai số nhỏ hơn.

Để làm ví dụ chúng ta xét phương trình :

$$\ln x - x^2 + 3 = 0$$

Ta đưa về dạng lặp :

$$x = \sqrt{\ln(x) + 3}$$

$$f'(x) = \frac{1}{2x\sqrt{\ln x + 3}}$$

Phép lặp hội tụ trong đoạn [0.3,∞].Ta cho $x_1 = 1$ thì tính được :

x_2 = 1,7320508076
x_3 = 1.883960229
x_4 = 1.90614167
y = 1.909934347

Để giảm sai số ta có thể lặp nhiều lần

Chương trình 8-9

```
//phuong phap Aitken
#include <conio.h>
#include <stdio.h>
#include <math.h>

#define  m  5

void main()
 {
      float x[m];
```

```
        float epsi,n,y;
        int i,z;
        float f(float);

        clrscr();
        printf("Cho tri so ban dau x[1] = ");
        scanf("%f",&x[1]);
        printf("Cho tri so sai so epsilon = ");
        scanf("%f",&epsi);
        printf("\n");
        printf( "Ngoai suy Aitken cua ham\n");
        z=0;
        while (z<=20)
            {
                for (i=2;i<=4;i++)
                  x[i]=f(x[i-1]);
                n=x[4]-2*x[3]+x[2];
                if ((fabs(n)<1e-09)||(fabs(x[1]-x[2])<epsi*fabs(x[1])))
                  z=20;
                else
                  {
                        y=x[2]-(x[3]-x[2])*(x[3]-x[2])/n;
                        if (z>20)
                          printf("Khong hoi tu sau hai muoi lan lap\n");
                        x[1]=y;
                  }
                z=z+1;
          }
        printf("Nghiem cua phuong trinh y = %.6f",y);
        getch();
  }

float f(float x)
  {
        float s=sqrt(log(x)+3);
        return(s);
  }
```

Với giá trị ban đầu là 1 và sai số là 1e-8,chương trình cho kết quả y = 1.9096975944

§10.PHƯƠNG PHÁP BAIRSTOW

Nguyên tắc của phương pháp Bairstow là trích từ đa thức $P_n(x)$ một tam thức $Q_2(x) = x^2 - sx + p$ mà ta có thể tính nghiệm thực hay nghiệm phức của nó một cách đơn giản bằng các phương pháp đã biết.

Việc chia đa thức $P_n(x)$ cho tam thức $Q_2(x)$ đưa tới kết quả :

$$P_n(x) = Q_2(x).P_{n-2}(x) + R_1(x)$$

với $P_n(x) = a_o x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_n$

$Q_2(x) = x^2 - sx + p$

$$P_{n-2}(x) = b_o x^{n-2} + b_1 x^{n-3} + b_2 x^{n-4} + ... + b_{n-2}$$
$$R_1(x) = \alpha x + \beta$$

Để có được một thương đúng,cần tìm các giá trị của s và p sao cho $R_1(x) = 0$ (nghĩa là α và β triệt tiêu).Với s và p đã cho,các hệ số b của đa thức $P_{n-2}(x)$ và các hệ số α và β được tính bằng phương pháp truy hồi.Các công thức nhận được khi khai triển biểu thức $P_n(x) = Q_2(x).P_{n-2}(x) + R_1(x)$ và sắp xếp lại các số hạng cùng bậc :

$$a_o x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_n = (x^2 - sx + p)(b_o x^{n-2} + b_1 x^{n-3} + b_2 x^{n-4} + ... + b_{n-2})$$

Số hạng bậc	Hệ số của $P_n(x)$	Hệ số của $Q_2(x).P_{n-2}(x)$
x^n	a_o	b_o
x^{n-1}	a_1	$b_1 - sb_o$
x^{n-2}	a_2	$b_2 - sb_1 + pb_o$
.....	.....	.....
x^{n-k}	a_k	$b_k - sb_{k-1} + pb_{k-2}$
x	a_{n-1}	$\alpha - sb_{n-2} + pb_{n-3}$
x^o	a_n	$\beta + pb_{n-2}$

Như vậy : (1)

$$b_o = a_o$$
$$b_1 = a_1 + sb_o$$
$$b_2 = a_2 + sb_1 - pb_o$$
.....
$$b_k = a_k + sb_{k-1} - pb_{k-2}$$
$$\alpha = a_{n-1} + sb_{n-2} - pb_{n-3}$$
$$\beta = a_n - pb_{n-2}$$

Chúng ta nhận thấy rằng α được tính toán xuất phát từ cùng một công thức truy hồi như các hệ số b_k và tương ứng với hệ số b_{n-1}

$$b_{n-1} = a_{n-1} + sb_{n-2} - pb_{n-3} = \alpha$$

Hệ số b_n là :

$$b_n = a_n + sb_{n-1} - pb_{n-2} = sb_{n-1} + \beta$$

và cuối cùng :

$$R_1(x) = \alpha x + \beta = b_{n-1}(x - s) + b_n$$

Ngoài ra các hệ số b_i phụ thuộc vào s và p và bây giờ chúng ta cần phải tìm các giá trị đặc biệt s^* và p^* để cho b_{n-1} và b_n triệt tiêu.Khi đó $r_1(x) = 0$ và nghiệm của tam thức $x^2 - s^*x + p^*x$ sẽ là nghiệm của đa thức $P_n(x)$.Ta biết rằng b_{n-1} và b_n là hàm của s và p :

$$b_{n-1} = f(s,p)$$
$$b_n = g(s,p)$$

Việc tìm s^* và p^* đưa đến việc giải hệ phương trình phi tuyến:

$$\begin{cases} f(s,p)=0 \\ g(s,p)=0 \end{cases}$$

Phương trình này có thể giải dễ dàng nhờ phương pháp Newton.Thật vậy với một phương trình phi tuyến ta có công thức lặp :

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

hay $$f'(x_i)(x_{i+1} - x_i) = -f(x_i)$$

Với một hệ có hai phương trình,công thức lặp trở thành:

$$J(X_i)(X_{i+1} - X_i) = -F(X_i)$$

với $X_i = \{ s_i,p_i \}^T$ và $X_{i+1} = \{ s_{i+1},p_{i+1} \}^T$

$$F(X_i)=\begin{vmatrix} f(s_i,p_i) \\ g(s_i,p_i) \end{vmatrix}$$

$$J(X_i)=\begin{vmatrix} \frac{\partial f}{\partial s} & \frac{\partial f}{\partial p} \\ \frac{\partial g}{\partial s} & \frac{\partial g}{\partial p} \end{vmatrix}$$

Quan hệ : $J(X_i)\Delta X = -F(X_i)$ với $\Delta X = \{s_{i+1} - s_i, p_{i+1} - p_i\}^T$ tương ứng với một hệ phương trình tuyến tính hai ẩn số $\Delta s = s_{i+1} - s_i$ và $\Delta p = p_{i+1} - p_i$:

$$\begin{cases} \frac{\partial f}{\partial s}\Delta s + \frac{\partial f}{\partial p}\Delta p = -f(s_i, p_i) \\ \frac{\partial g}{\partial s}\Delta s + \frac{\partial g}{\partial p}\Delta p = -g(s_i, p_i) \end{cases}$$

Theo công thức Cramer ta có :

$$\Delta s = \frac{-f\frac{\partial g}{\partial p} + g\frac{\partial f}{\partial p}}{\delta}$$

$$\Delta p = \frac{-g\frac{\partial f}{\partial s} + f\frac{\partial g}{\partial s}}{\delta}$$

$$\delta = \frac{\partial f}{\partial s}\frac{\partial g}{\partial p} - \frac{\partial f}{\partial p}\frac{\partial g}{\partial s}$$

Để dùng được công thức này ta cần tính được các đạo hàm $\frac{\partial f}{\partial s}, \frac{\partial f}{\partial p}, \frac{\partial g}{\partial s}, \frac{\partial g}{\partial p}$.Các đạo hàm này được tính theo công thức truy hồi.

Do $b_o = a_o$ nên

$$\frac{\partial b_o}{\partial s} = 0 \qquad \frac{\partial b_o}{\partial p} = 0$$

$b_1 = a_1 + sb_o$ nên

$$\frac{\partial b_1}{\partial s} = b_o \qquad \frac{\partial b_1}{\partial p} = 0$$

$b_2 = a_2 + sb_1 - pb_o$ nên

$$\frac{\partial b_2}{\partial s} = \frac{\partial a_2}{\partial s} + \frac{\partial (sb_1)}{\partial s} - \frac{\partial (pb_o)}{\partial s}$$

Mặt khác :

$$\frac{\partial a_2}{\partial s} = 0 \qquad \frac{\partial (sb_1)}{\partial s} = s\frac{\partial b_1}{\partial s} + b_1 \qquad \frac{\partial (pb_o)}{\partial s} = 0$$

nên :

$$\frac{\partial b_2}{\partial s} = b_1 + sb_o$$

$b_3 = a_3 + sb_2 - pb_1$ nên

$$\frac{\partial b_3}{\partial s} = b_2 + s\frac{\partial b_2}{\partial s} - p\frac{\partial b_1}{\partial s}$$

Nếu chúng ta đặt :

$$\frac{\partial b_k}{\partial s} = c_{k-1}$$

thì :

$$c_o = b_o \qquad (2)$$
$$c_1 = b_1 + sb_o = b_1 + sc_o$$

$$c_2 = b_2 + sc_1 - pc_o$$
.....
$$c_k = b_k + sc_{k-1} - pc_{k-2}$$
$$c_{n-1} = b_{n-1} + sc_{n-2} - pc_{n-3}$$

Như vậy các hệ số cũng được tính theo cách như các hệ số b_k.Cuối cùng với $f = b_{n-1}$ và $g = b_n$ ta được:

$$\frac{\partial f}{\partial s} = c_{n-2} \quad \frac{\partial f}{\partial s} = c_{n-3} \quad \frac{\partial f}{\partial s} = c_{n-1} \quad \frac{\partial f}{\partial s} = c_{n-2}$$

$$\Delta s = \frac{b_{n-1}c_{n-2} - b_n c_{n-3}}{c_{n-1}c_{n-3} - c_{n-2}^2} \qquad (3)$$

$$\Delta p = \frac{b_{n-1}c_{n-1} - b_n c_{n-2}}{c_{n-1}c_{n-3} - c_{n-2}^2} \qquad (4)$$

Sau khi phân tích xong $P_n(x)$ ta tiếp tục phân tích $P_{n-2}(x)$ theo phương pháp trên Các bước tính toán gồm :

- Chọn các giá trị ban đầu bất kì s_0 và p_0
- Tính các giá trị $b_o,..,b_n$ theo (1)
- Tính các giá trị $c_o,...,c_n$ theo (2)
- Tính Δs_o và Δp_o theo (3) và (4)
- Tính $s_1 = s_0 + \Delta s_o$ và $p_1 = p_o + \Delta p_o$
- Lặp lại bước 1 cho đến khi $p_{i+1} = p_i = p$ và $s_{i+1} = s_i = s$
- Giải phương trình $x_2 - sx + p$ để tìm 2 nghiệm của đa thức
- Bắt đầu quá trình trên cho đa thức $P_{n-2}(x)$

Ví dụ : Tìm nghiệm của đa thức $P_4(x) = x^4 - 1.1x^3 + 2.3x^2 + 0.5x^2 + 3.3$.
Với lần lặp ban đầu ta chọn s = -1 và p =1,nghĩa là tam thức có dạng $x^2 + x + 1$

	a_0	a_1	a_2	a_3	a_4
	1	-1.1	2.3	0.5	3.3
sb_i		-1	2.1	-3.4	0.8
$-pb_{i-1}$			-1	2.1	-3.4
b_i	1	-2.1	3.4	$-0.8 = b_{n-1}$	$0.7 = b_n$
sb_i		-1.0	3.1	-5.5	
$-pb_{i-1}$			-1.0	3.1	
c_i	1	-3.1	5.5	-3.2	

$$\Delta s = \frac{\begin{vmatrix} 0.8 & -3.1 \\ -0.7 & 5.5 \end{vmatrix}}{\begin{vmatrix} 5.5 & -3.1 \\ -3.2 & 5.5 \end{vmatrix}} = 0.11$$

$$\Delta p = \frac{\begin{vmatrix} 5.5 & 0.8 \\ -3.2 & 0.7 \end{vmatrix}}{\begin{vmatrix} 5.5 & -3.1 \\ -3.2 & 5.5 \end{vmatrix}} = 0.06$$

$s^* = -1 + 0.11 = -0.89$
$p^* = 1 + 0.06 = 1.06$

Tiếp tục lặp lần 2 với $s_1 = s^*$ và $p_1 = p^*$ ta có :

	a_0	a_1	a_2	a_3	a_4
	1	-1.1	2.3	0.5	3.3
sb_i		-0.89	1.77	-2.68	0.06
$-pb_{i-1}$			-1.06	2.11	-3.17
b_i	1	-1.99	3.01	$-0.07 = b_{n-1}$	$0.17 = b_n$
sb_i		-0.89	2.56	-4.01	
$-pb_{i-1}$			-1.0	3.1	
c_i	1	-2.88	4.51	-1.03	

$$\Delta s = \frac{\begin{vmatrix} 0.07 & -2.88 \\ -0.7 & 5.5 \end{vmatrix}}{\begin{vmatrix} 4.51 & -2.88 \\ -1.03 & 4.51 \end{vmatrix}} = -0.01$$

$$\Delta p = \frac{\begin{vmatrix} 4.51 & 0.07 \\ -1.03 & -0.17 \end{vmatrix}}{\begin{vmatrix} 4.51 & -2.88 \\ -1.03 & 4.51 \end{vmatrix}} = 0.04$$

$s^* = -0.89 - 0.01 = -0.9$

$p^* = 1.06 + 0.04 = 1.1$

Như vậy $P_4(x) = (x^2+0.9x+1.1)(x^2+2x+3)$

Chương trình sau áp dụng lí thuyết vừa nêu để tìm nghiệm của đa thức.

Chương trình 8-10

```
//phuong phap Bairstow
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define  m   10

void main()
  {
        float a[m],b[m],c[m];
        int i,n,v;
        float s,e1,t,p,q,r,p1,q1;

        clrscr();
        printf("Cho bac cua da thuc n = ");
        scanf("%d",&n);
        printf("Cho cac he so cua da thuc can tim nghiem\n");
        for (i=n;i>=0;i--)
          {
                printf("a[%d] = ",n-i);
                scanf("%f",&a[i]);
          }
        printf("\n");
```

```
e1=0.0001;
if (n<=2)
    if (n==1)
        {
          printf("Nghiem cua he\n");
          printf("%.8f",(a[0]/(-a[1])));
          getch();
          exit(1);
        }
do
  {
        v=0;
        p=1;
        q=-1;
        b[n]=a[n];
        c[n]=a[n];
        do
          {
                b[n-1]=b[n]*p+a[n-1];
                c[n-1]=b[n-1]+b[n]*p;
                for (i=n-2;i>=0;i--)
                  {
                        b[i]=b[i+2]*q+b[i+1]*p+a[i];
                        c[i]=c[i+2]*q+c[i+1]*p+b[i];
                   }
                r=c[2]*c[2]-c[1]*c[3];
                p1=p-(b[1]*c[2]-b[0]*c[3])/r;
                q1=q-(b[0]*c[2]-b[1]*c[1])/r;
                if ((fabs(b[0])<e1)&&(fabs(b[1])<e1))
                  goto tt;
                v=v+1;
                p=p1;
                q=q1;
          }
        while (v<=40);
        if(v>40)
          {
                printf("Khong hoi tu sau 40 lan lap");
                getch();
                exit(1);
          }
        tt:s=p1/2;
        t=p1*p1+4*q1;
        if(t<0)
          {
                printf("Nghiem phuc\n");
                printf("%.8f+%.8fj\n",s,(sqrt(-t)/2));
                printf("%.8f-%.8fj\n",s,(sqrt(-t)/2));
                printf("\n");
          }
```

```
            else
              {
                    printf("Nghiem thuc\n");
                    printf("%.8f\n",(s+sqrt(t)/2));
                    printf("%.8f\n",(s-sqrt(t)/2));
                    printf("\n");
              }
            for (i=2;i<=n;i++)
              a[i-2]=b[i];
              n=n-2;
      }
    while ((n>2)&(r!=0.0));
    s=-a[1]/(2*a[2]);
    t=a[1]*a[1]-4*a[2]*a[0];
    if (t<0)
      {
            printf("Nghiem phuc\n");
            printf("%.8f+%.8fj\n",s,(sqrt(-t)/(2*a[2])));
            printf("%.8f-%.8fj\n",s,(sqrt(-t)/(2*a[2])));
            printf("\n");
      }
    else
      {
            printf("Nghiem thuc\n");
            printf("%.8f\n",(s-sqrt(t)/(2*a[2])));
            printf("%.8f\n",(s-sqrt(t)/(2*a[2])));
            printf("\n");
      }
    getch();
}
```

Dùng chương trình trên để xác định nghiệm của đa thức :

$$x^6 - 2x^5 - 4x^4 + 13x^3 - 24x^2 + 18x - 4 = 0$$

ta nhận được các nghiệm :

$x_1 = 2.61903399$

$x_2 = -2.73205081$

$x_3 = 0.732050755$

$x_4 = 0.381966055$

$x_5 = 0.500011056 + i*1.3228881$

$x_6 = 0.500011056 - i*1.3228881$

§11.HỆ PHƯƠNG TRÌNH PHI TUYẾN

Phương pháp Newton có thể được tổng quát hoá để giải hệ phương trình phi tuyến dạng :

$$\begin{cases} f_1(x_1,x_2,x_3,...,x_2)=0 \\ f_2(x_1,x_2,x_3,...,x_2)=0 \\ f_3(x_1,x_2,x_3,...,x_2)=0 \\ ..... \\ f_n(x_1,x_2,x_3,...,x_2)=0 \end{cases}$$

hay viết gọn hơn dưới dạng :

$$F(X) = 0$$

Trong đó : $X = (x_1,x_2,x_3,.....,x_n)$

Với một phương trình một biến,công thức Newton là :

$$x_{i+1}=x_i-\frac{f(x_i)}{f'(x_i)}$$

hay : $f'(x_i).\Delta x = -f(x_i)$

với $\Delta x = x_{i+1} - x_i$

Đối với hệ,công thức lặp là :

$$J(X_i)\Delta x = -F(X_i)$$

Trong đó $J(X_i)$ là toán tử Jacobi.Nó là một ma trận bậc n (n - tương ứng với số thành phần trong vectơ X) có dạng :

$$J(x_i)=\begin{vmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3}..... & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3}..... & \frac{\partial f_2}{\partial x_n} \\ . & & & \\ . & & & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3}..... & \frac{\partial f_n}{\partial x_n} \end{vmatrix}$$

và $\Delta X = X_{i+1} - X_i$

Phương pháp Newton tuyến tính hoá hệ và như vậy với mỗi bước lặp cần giải một hệ phương trình tuyến tính (mà biến là Δx_i) xác định bởi công thức lặp cho tới khi vectơ $X(x_1,x_2,x_3,.....,x_n)$ gần với nghiệm.

Dưới đây là chương trình giải hệ phương trình phi tuyến

$$\begin{cases} x_1^3-x_2^3-3x_1x_2x_4-8=0 \\ x_1+x_2+x_3+x_4-5=0 \\ \sqrt{25-x_1^2}+8x_3+4=0 \\ 2x_1x_2x_3-x_4+8=0 \end{cases}$$

Ma trận đạo hàm riêng $J(x_i)$là :

$$\begin{pmatrix} 3x_1^2-3x_2x_4 & -3x_2^2-3x_1x_4 & 0 & -3x_1x_2 \\ 1 & 1 & 1 & 1 \\ \frac{-x_1}{\sqrt{25-x_1^2}} & 0 & 8 & 0 \\ 2x_2x_3 & 2x_2x_3 & 2x_2x_3 & -1 \end{pmatrix}$$

Ma trận này được chương trình đọc vào nhờ thủ tục doc.Trong thủ tục này,các hệ số a[i,5] là các hàm $f_i(x)$.Vectơ nghiệm ban đầu được chọn là $\{ 0,-1,-1,1\}^T$.Kết quả tính cho ta : x = $\{0.01328676,-1.94647929,-1.12499779,8.05819031 \}^T$ với độ chính xác 0.000001.Vectơ số dư r = $\{ 0.00000536,-0.00000011,-0.00000001,-0.00000006\}^T$.

Chương trình 8-11

```
//giai he pt phi tuyen
#include <conio.h>
#include <stdio.h>
```

```
#include <math.h>
#include <stdlib.h>
#define n  4

float a[n+1][n+2];
float x[n+1],y[n+1];
int i,j,k,l,z,r;
float e,s,t;

void main()
 {
        void doc();
   clrscr();
        printf("Cho cac gia tri nghiem ban dau\n");
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
          }
        e=1e-6;
        z=30;
        for (r=1;r<=z;r++)
          {
                doc();
                for (k=1;k<=n-1;k++)
                  {
                        s=0 ;
                        for (i=k;i<=n;i++)
                          {
                                t=fabs(a[i][k]);
                                if (s<=t)
                                  {
                                        s=t;
                                        l=i;
                                  }
                          }
                        for (j=k;j<=n+1;j++)
                          {
                                s=a[k][j];
                                a[k][j]=a[l][j];
                                a[l][j]=s;
                          }
                        if (a[1][1]==0)
                          {
                                printf("Cac phan tu duong cheo cua ma tran bang khong");
                                getch();
                                exit(1);
                          }
                        else
                          {
```

```
                        if (fabs(a[k][k]/a[1][1])<(1e-08))
                          {
                                printf("Ma tran suy bien");
                                goto mot;
                          }
                      }
                    for (i=k+1;i<=n;i++)
                      {
                        if (a[k][k]==0)
                          {
                                printf("Cac phan tu duong cheo cua ma tran bang khong\n");
                                goto mot;
                          }
                        s=a[i][k]/a[k][k];
                        a[i][k]=0;
                        for (j=k+1;j<=n+1;j++)
                         a[i][j]=a[i][j]-s*a[k][j];
                      }
                    y[n]=a[n][n+1]/a[n][n];
                    for (i=n-1;i>=1;i--)
                      {
                        s=a[i][n+1];
                        for (j=i+1;j<=n;j++)
                          s=s-a[i][j]*y[j];
                        if (a[i][i]==0)
                          {
                                printf("Cac phan tu duong cheo cua ma tran bang khong\n");
                                goto mot;
                          }
                        y[i]=s/a[i][i];
                      }
                  }
                if (r!=1)
                  for (i=1;i<=n;i++)
                        {
                          if (fabs(y[i])<e*fabs(x[i]))
                                goto ba;
                        }
                for (i=1;i<=n;i++)
                  x[i]=x[i]-y[i];
                printf("\n");
          }
        printf("Khong hoi tu sau %d lan lap\n",z);
        goto mot;
        clrscr();
        ba:printf("Vec to nghiem\n");
        for (i=1;i<=n;i++)
          printf("%.5f\n",(x[i]-y[i]));
```

```
        printf("\n");
        printf("Do chinh xac cua nghiem la %.5f: \n", e);
        printf("\n");
        printf("Vec to tri so du :\n");
        for (i=1;i<=n;i++)
          printf("%.5f\n",(a[i][n+1]));
        mot:printf("\n");
        getch();
  }

void doc()
  {
        a[1][1]=3*x[1]*x[1]-3*x[2]*x[4];
        a[1][2]=-3*x[2]*x[2]-3*x[1]*x[4];
        a[1][3]=0;
        a[1][4]=-3*x[1]*x[2];
        a[1][5]=x[1]*x[1]*x[1]-x[2]*x[2]*x[2]-3*x[1]*x[2]*x[4]-8;

        a[2][1]=1;
        a[2][2]=1;
        a[2][3]=1;
        a[2][4]=1;
        a[2][5]=x[1]+x[2]+x[3]+x[4]-5;

        a[3][1]=-x[1]/sqrt(25-x[1]*x[1]);
        a[3][2]=0;
        a[3][3]=8;
        a[3][4]=0;
        a[3][5]=sqrt(25-x[1]*x[1])+8*x[3]+4;

        a[4][1]=2*x[2]*x[3];
        a[4][2]=2*x[1]*x[3];
        a[4][3]=2*x[1]*x[2];
        a[4][4]=-1;
        a[4][5]=2*x[1]*x[2]*x[3]-x[4]+8;

  }
```

CHƯƠNG 9 : CÁC VẤN ĐỀ VỀ MA TRẬN

§1.ĐỊNH THỨC CỦA MA TRẬN

Cho một ma trận vuông cấp n.Ta cần tìm định thức của nó.Trước hết chúng ta nhắc lại một số tính chất quan trọng của định thức:

- nếu nhân tất cả các phần tử của một hàng (hay cột) với k thì định thức được nhân với k
- định thức không đổi nếu ta cộng thêm vào một hàng tổ hợp tuyến tính của các hàng còn lại.

Ta sẽ áp dụng các tính chất này để tính định thức của một ma trận cấp 4 như sau(phương pháp này có thể mở rộng cho một ma trận cấp n) bằng phương pháp trụ:

$$A=\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

Lấy giá trị trụ là $p_1= a_{11}$.Ta chia các phần tử của hàng thứ nhất cho $p_1= a_{11}$ thì định thức sẽ là D/p_1 (theo tính chất 1) và ma trận còn lại là:

$$\begin{pmatrix} 1 & a'_{12} & a'_{13} & a'_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

Lấy hàng 2 trừ đi hàng 1 đã nhân với a_{21},lấy hàng 3 trừ đi hàng 1 đã nhân với a_{31} và lấy hàng 4 trừ đi hàng 1 đã nhân với a_{41} (thay hàng bằng tổ hợp tuyến tính của các hàng còn lại) thì định thức vẫn là D/p_1 và ma trận là:

$$\begin{pmatrix} 1 & a'_{12} & a'_{13} & a'_{14} \\ 0 & a'_{22} & a'_{23} & a'_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a'_{42} & a'_{43} & a'_{44} \end{pmatrix}$$

Lấy giá trị trụ là $p_2 = a'_{22}$.Ta chia các phần tử của hàng thứ hai cho p_2 thì định thức sẽ là $D/(p_1p_2)$ và ma trận còn lại là:

$$\begin{pmatrix} 1 & a'_{12} & a'_{13} & a'_{14} \\ 0 & 1 & a''_{23} & a''_{24} \\ 0 & a'_{32} & a'_{33} & a'_{34} \\ 0 & a'_{42} & a'_{43} & a'_{44} \end{pmatrix}$$

Lấy hàng 1 trừ đi hàng 2 đã nhân với a'_{12} ,lấy hàng 3 trừ đi hàng 2 đã nhân với a'_{32} và lấy hàng 4 trừ đi hàng 2 đã nhân với a'_{42} thì định thức vẫn là D/p_1 và ma trận là:
thì định thức vẫn là $D/(p_1p_2)$ và ma trận là:

$$\begin{pmatrix} 1 & 0 & a''_{13} & a''_{14} \\ 0 & 1 & a''_{23} & a''_{24} \\ 0 & 0 & a''_{33} & a''_{34} \\ 0 & 0 & a''_{43} & a''_{44} \end{pmatrix}$$

Tiếp tục lấy hàng 3 rồi hàng 4 làm trụ thì ma trận sẽ là:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Định thức của ma trận này là $D/(p_1p_2p_3p_4)= D/(a_{11}a'_{22}a''_{33}a'''_{44}) =1$ nên định thức của ma trận A là $D = p_1p_2p_3p_4$.

Sau đây là chương trình tìm định thức của một ma trận:

Chương trình 9-1

```
//tinh dinh thuc
#include <conio.h>#include <stdio.h>#include <ctype.h>
#include <stdlib.h>

void main()
  {
        int i,j,k,n,ok1,ok2,t;
        float d,c,e,f,g,h;
        float a[50][50];
        char tl;

        clrscr();
        printf("** TINH DINH THUC CAP n **");
        printf("\n");
        printf("\n");
        printf("Cho cap cua dinh thuc n = ");
        scanf("%d",&n);
        printf("Nhap ma tran a\n");
        for (i=1;i<=n;i++)
        {
         printf("Dong %d:\n",i);
                for (j=1;j<=n;j++)
                  {
                        printf("a[%d][%d] = ",i,j);
                        scanf("%f",&a[i][j]);
                  }
                printf("\n");
         }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
         {
                for (j=1;j<=n;j++)
```

```
                printf("%.5f\t",a[i][j]);
              printf("\n");
          }
        printf("\n");
        t=1;
flushall();
        while (t)
          {
              printf("Co sua ma tran a khong(c/k)?");
              scanf("%c",&tl);
              if (toupper(tl)=='C')
                {
                    printf("Cho chi so hang can sua : ");
                    scanf("%d",&i);
                    printf("Cho chi so cot can sua : ");
                    scanf("%d",&j);
                    printf("a[%d][%d] = ",i,j);
                    scanf("%f",&a[i,j]);
                }
              if (toupper(tl)=='K')
                t=0;
          }
        printf("Ma tran a ban dau\n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
              for (j=1;j<=n;j++)
                printf("%.5f\t",a[i][j]);
              printf("\n");
          }
        printf("\n");
        d=1;
        i=1;
        ok2=1;
        while ((ok2)&&(i<=n))
          {
              if (a[i][i]==0)
                {
                    ok1=1;
                    k=k+1;
                    while ((ok1)&&(k<=n))
                      if (a[k,i]!=0)
                          {
                            for (j=i;j<=n;j++)
                                {
                                  c=a[i][j];
                                  a[i][j]=a[k][j];
                                  a[k][j]=c;
                                }
                            d=-d;
```

```
                        ok1=0;
                      }
                else
                      k=k+1;
                      if (k>n)
                        {
                              printf("\n");
                              printf("** MA TRAN SUY BIEN **");
                              ok2=0;
                              d=0;
                        }
            }
          if (a[i][i]!=0)
            {
                  c=a[i][i];
                  for (j=i+1;j<=n;j++)
                    a[i][j]=a[i][j]/c;
                    for (k=i+1;k<=n;k++)
                          {
                           c=a[k][i];
                           for (j=i+1;j<=n;j++)
                                 a[k][j]=a[k][j]-a[i][j]*c;
                          }
            }
          i=i+1;
      }
    if (ok2)
      {
          for (i=1;i<=n;i++)
             d=d*a[i][i];
          printf("\n");
          printf("** GIA TRI DINH THUC D **");
          printf("\n");
          printf("%.3f",d);
      }
    getch();
}
```

§2.NGHỊCH ĐẢO MA TRẬN

Gọi $\mathbf{A}^{-1}$ là ma trận nghịch đảo của một ma trận **A** bậc n ta có $\mathbf{A}\mathbf{A}^{-1} = \mathbf{E}$.(trong biểu thức này **E** là một ma trận vuông có các phần tử trên đường chéo chính bằng 1). Dạng của ma trận E,ví dụ cấp 4,là:

$$E = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Phương pháp loại trừ để nhận được ma trận nghịch đảo $\mathbf{A}^{-1}$ được thực hiện qua nhiều giai đoạn (n),mỗi một giai đoạn gồm hai bước.Đối với giai đoạn thứ k:

- chuẩn hoá phần tử a_{kk} bằng cách nhân hàng với nghịch đảo của nó
- làm cho bằng không các phần tử phía trên và phía dưới đường chéo cho đến cột thứ k.Khi k = n thì $\mathbf{A}^{(k)}$ sẽ trở thành ma trận đơn vị và **E** trở thành $\mathbf{A}^{-1}$

Ví dụ: Tính ma trận nghịch đảo của ma trận

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

Ta viết lại ma trận A và ma trận đơn vị tương ứng với nó

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} \quad E = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

<u>Giai đoạn 1</u>: ***Bước a***: Nhân hàng 1 với $1/a_{11}$,nghĩa là $a'_{1j} = a_{1j}/a_{11}$ đối với dòng thứ nhất,$a'_{ij} = a_{ij}$ đối với các dòng khác

$$A = \begin{pmatrix} 1 & 1/2 & 1/2 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} \quad E = \begin{pmatrix} 1/2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Bước b: Trừ hàng 3 và hàng 2 cho hàng 1,nghĩa là $a^{(1)}_{1j} = a_{ij} - a_{i1}a_{ij}$ đối với $i \neq 1$.

$$A = \begin{pmatrix} 1 & 1/2 & 1/2 \\ 0 & 3/2 & 1/2 \\ 0 & 1/2 & 3/2 \end{pmatrix} \quad E = \begin{pmatrix} 1/2 & 0 & 0 \\ -1/2 & 1 & 0 \\ -1/2 & 0 & 1 \end{pmatrix}$$

<u>Giai đoạn 2</u>: ***Bước a***: Lấy hàng 2 làm chuẩn,nhân hàng 2 với 2/3,để nguyên các hàng khác

$$A = \begin{pmatrix} 1 & 1/2 & 1/2 \\ 0 & 1 & 1/3 \\ 0 & 1/2 & 3/2 \end{pmatrix} \quad E = \begin{pmatrix} 1/2 & 0 & 0 \\ -1/3 & 2/3 & 0 \\ -1/2 & 0 & 1 \end{pmatrix}$$

Bước b: Lấy hàng 1 trừ đi hàng 2 nhân 1/2 và lấy hàng 3 trừ đi hàng 2 nhân 1/2

$$A = \begin{pmatrix} 1 & 0 & 1/3 \\ 0 & 1 & 1/3 \\ 0 & 0 & 4/3 \end{pmatrix} \quad E = \begin{pmatrix} 2/3 & -1/3 & 0 \\ -1/3 & 2/3 & 0 \\ -1/3 & -1/3 & 1 \end{pmatrix}$$

<u>Giai đoạn 3</u>: ***Bước a***: Lấy hàng 3 làm chuẩn,nhân hàng 3 với 3/4,để nguyên các hàng khác

$$A = \begin{pmatrix} 1 & 0 & 1/3 \\ 0 & 1 & 1/3 \\ 0 & 0 & 1 \end{pmatrix} \quad E = \begin{pmatrix} 2/3 & -1/3 & 0 \\ -1/3 & 2/3 & 0 \\ -1/4 & -1/4 & 3/4 \end{pmatrix}$$

Bước b: Lấy hàng 1 trừ đi hàng 3 nhân 1/3 và lấy hàng 2 trừ đi hàng 3 nhân 1/3

$$A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad E = \begin{pmatrix} 3/4 & -1/4 & -1/4 \\ -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & 3/4 \end{pmatrix}$$

Như vậy A^{-1} là:

$$A^{-1} = \begin{pmatrix} 3/4 & -1/4 & -1/4 \\ -1/4 & 3/4 & -1/4 \\ -1/4 & -1/4 & 3/4 \end{pmatrix}$$

Áp dụng phương pháp này chúng ta có chương trình sau:

Chương trình 9-2

```
#include <conio.h>#include <stdio.h>#include <math.h>#include <stdlib.h>#include
<ctype.h>void main() {     int i,j,k,n,t,t1; float c,a[50][50],b[50][50];    char tl;      clrscr();
        printf("       **MA TRAN NGHICH DAO** \n");
        printf("Cho bac cua ma tran n = ");
        scanf("%d",&n);
        printf("Vao ma tran ban dau a\n");
        for (i=1;i<=n;i++)
          {
                printf("Vao hang thu %d :\n",i);
                for (j=1;j<=n;j++)
                  {
                        printf("a[%d][%d] = ",i,j);
                        scanf("%f",&a[i][j]);
                  }
                printf("\n");
          }
        printf("\n");
        printf("Ma tran ban da nhap\n");
    printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%.5f\t",a[i][j]);
                printf("\n");
          }
        t=1;
        flushall();
        while (t)
          {
                printf("\nCo sua ma tran khong(c/k)?");
                scanf("%c",&tl);
                if(toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[%d][%d] = ",i,j);
                        scanf("%f",&a[i][j]);
                  }
                if (toupper(tl)=='K')
                  t=0;
          }
        printf("\nMa tran ban dau\n");
```

```
printf("\n");
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%.5f\t",a[i][j]);
        printf("\n");
  }
printf("\n");
for (i=1;i<=n;i++)
  for (j=n+1;j<=2*n;j++)
        {
         if (j==i+n)
                a[i][j]=1;
         else
                a[i][j]=0;
        }
  i=1;
  t1=1;
  while (t1&&(i<=n))
        {
         if (a[i][i]==0)
                {
                 t=1;
                 k=i+1;
                 while (t&&(k<=n))
                        if (a[k][i]!=0)
                          {
                                for (j=1;j<=2*n;j++)
                                  {
                                        c=a[i][j];
                                        a[i][j]=a[k][j];
                                  a[k][j]=c;
                                  }
                                t=0;
                          }
                        else
                          k=k+1;
                        if (k==n+1)
                          {
                                if (a[i][k-1]==0)
                                  {
                                        printf("MA TRAN SUY BIEN\n ");
                                        t1=0;
                                  }
                          }
                 }
                if (a[i][i]!=0)
                  {
                        c=a[i][i];
                        for (j=i;j<=2*n;j++)
```

```
                        a[i][j]=a[i][j]/c;
                    }
                for (k=1;k<=n;k++)
                  {
                        if (k!=i)
                          {
                              c=a[k][i];
                               for (j=i;j<=2*n;j++)
                                 a[k][j]=a[k][j]-a[i][j]*c;
                          }
                  }
                i=i+1;
            }
          if (t1)
            {
                printf("\n");
                printf("\nMA TRAN KET QUA\n");
                printf("\n");
                for (i=1;i<=n;i++)
                  {
                        for (j=n+1;j<=2*n;j++)
                          printf("%.4f\t\t",a[i][j]);
                        printf("\n");
                  }
                printf("\n");
            }
    getch();
}
```

Dùng chương trình tính nghịch đảo của ma trận:

$$\begin{pmatrix} 9 & 9 & 8 \\ 9 & 8 & 7 \\ 8 & 7 & 6 \end{pmatrix} \text{ cho ta kết quả } \begin{pmatrix} -1 & 2 & -1 \\ 2 & -10 & 9 \\ -1 & 9 & -9 \end{pmatrix}$$

§3.TÍCH HAI MA TRẬN

Giả sử ta có ma trận $\mathbf{A}_{mn}$ và ma trận $\mathbf{B}_{np}$.Tích của $\mathbf{A}_{mn}$ và $\mathbf{B}_{np}$ là ma trận $\mathbf{C}_{mp}$ trong đó mỗi phần tử của $\mathbf{C}_{mp}$ là: $c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$

Chương trình dưới đây thực hiện nhân hai ma trận với nhau.

Chương trình 9-3

#include <conio.h>#include <stdio.h>#include <math.h>#include <stdlib.h>#include <ctype.h>

#define max 50

void main()

```
{
        int n,l,m,i,j,k,t;
        float a[max][max],b[max][max],c[max][max];
        char tl;

        clrscr();
        printf("Cho so hang cua ma tran a : ");
        scanf("%d",&n);
        printf("Cho so cot cua ma tran a : ");
        scanf("%d",&l);
        printf("Cho so cot cua ma tran b : ");
        scanf("%d",&m);
        printf("\nNHAP MA TRAN A\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=l;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=l;j++)
                  printf("%10.5f",a[i][j]);
                printf("\n");
          }
        flushall();
        t=1;
        while (t)
          {
                printf("Co sua ma tran khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[%d][%d] = ",i,j);
                        scanf("%f",&a[i][j]);
                  }
                if (toupper(tl)=='K')
                  t=0;
          }
        printf("Ma tran a ban dau");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=l;j++)
```

```
            printf("%10.5f",a[i][j]);
          printf("\n");
  }
printf("\n");

printf("NHAP MA TRAN B\n");
for (i=1;i<=l;i++)
  for (j=1;j<=m;j++)
          {
            printf("b[%d][%d] = ",i,j);
            scanf("%f",&b[i][j]);
          }
printf("\n");
printf("Ma tran b ban da nhap\n");
for (i=1;i<=l;i++)
  {
          for (j=1;j<=m;j++)
            printf("%10.5f",b[i][j]);
          printf("\n");
  }
flushall();
t=1;
while (t)
  {
          printf("Co sua ma tran khong(c/k)?");
          scanf("%c",&tl);
          if (toupper(tl)=='C')
            {
                  printf("Cho chi so hang can sua : ");
                  scanf("%d",&i);
                  printf("Cho chi so cot can sua : ");
                  scanf("%d",&j);
                  printf("b[%d][%d] = ",i,j);
                  scanf("%f",&b[i][j]);
            }
          if (toupper(tl)=='K')
            t=0;
  }
printf("Ma tran b ban dau");
printf("\n");
for (i=1;i<=l;i++)
  {
          for (j=1;j<=m;j++)
            printf("%10.5f",b[i][j]);
          printf("\n");
  }
printf("\n");
for (i=1;i<=n;i++)
  for (j=1;j<=m;j++)
          {
```

```
                c[i][j]=0;
                for (k=1;k<=l;k++)
                        c[i][j]=c[i][j]+a[i][k]*b[k][j];
              }
        printf("Ma tran tich c :\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=m;j++)
                  printf("%10.5f",c[i][j]);
                printf("\n");
          }
        getch();
}
```

Dùng chương trình tính tính hai ma trận ta nhận được kết quả

$$\begin{pmatrix} 2 & 1 \\ -1 & 3 \\ 1 & 0 \\ 5 & 3 \end{pmatrix} \times \begin{pmatrix} 1 & 2 & -2 \\ 3 & -4 & 3 \end{pmatrix} = \begin{pmatrix} 5 & 0 & -1 \\ 8 & -14 & 11 \\ 1 & 2 & -2 \\ 14 & -2 & -1 \end{pmatrix}$$

§4.GIÁ TRỊ RIÊNG VÀ VEC TƠ RIÊNG CỦA MA TRẬN

1.Khái niệm chung: Trong nghiên lí thuyết và ứng dụng,ta gặp bài toán về ma trận cấp n.Cho một ma trận A cấp n,giá trị λ được gọi là giá trị riêng và vectơ X được gọi là vectơ riêng của ma trận A nếu:

$$AX = \lambda X \qquad (1)$$

Vectơ riêng phải là vectơ khác không.Tương ứng với một giá trị riêng có vô số vectơ riêng.Nếu X là một véc tơ riêng tương ứng với giá trị riêng λ thì cX cũng là vec tư riênh ứng với λ.Có nhiều thuật toán tìm giá trị riêng và vectơ riêng của một ma trận.Giả sử ta có ma trận A,gọi E là ma trận đơn vị thì theo (1) ta có:

$$(A-\lambda E)X = 0 \qquad (2)$$

và $(A - \lambda E)$ là ma trận có dạng:

$$\begin{pmatrix} a_{11}-\lambda & a_{12} \cdots & a_{1n} \\ a_{21} & a_{22}-\lambda \cdots & a_{2n} \\ \cdots\cdots & & \\ a_{n1} & a_{n2} \cdots & a_{nn}-\lambda \end{pmatrix} \qquad (3)$$

Như vậy do (2) là hệ phương trình tuyến tính thuần nhất nên điều kiện cần và đủ để λ là giá trị riêng của ma trận trên là định thức của nó bằng không:

$$\det(A - \lambda E) = 0 \qquad (4)$$

Phương trình (4) được gọi là phương trình đặc trưng của ma trận A.Định thức $\det(A - \lambda E)$ được gọi là định thức đặc trưng của ma trận A.Định thức $P_A(\lambda)$ của ma trận trên được gọi là đa thức đặc trưng của ma trận vuông A.

Ví dụ tìm vec tơ riêng và trị riêng của ma trận:

$$\begin{pmatrix} 3 & 1 & -3 \\ 3 & 1 & -1 \\ 2 & -2 & 0 \end{pmatrix}$$

Trước hết ta tính đa thức đặc trưng của ma trận A:

$$P_A(\lambda)=\begin{pmatrix}3-\lambda & 1 & -3\\ 3 & 1-\lambda & -1\\ 2 & -2 & -\lambda\end{pmatrix}=(4-\lambda)(\lambda^2+4)$$

Nghiệm của $P_A(\lambda)=0$ là $\lambda_1=4, \lambda_2=2j$ và $\lambda_3=-2j$. Vì trường cơ sở là số thực nên ta chỉ lấy $\lambda=4$. Để tìm vec tơ riêng tương ứng với $\lambda=4$ ta giải hệ

$$\begin{pmatrix}3-\lambda & 1 & -3\\ 3 & 1-\lambda & -1\\ 2 & -2 & -\lambda\end{pmatrix}\times\begin{pmatrix}\xi_1\\ \xi_2\\ \xi_3\end{pmatrix}=0$$

ta nhận được các giá trị của ξ, chúng tạo thành vec tơ riêng ứng với λ.
Như vậy khi khai triển định thức ta có một đa thức bậc n có dạng:

$$P_n(\lambda)=\lambda^n - p_1\lambda^{n-1} - p_2\lambda^{n-2} - \ldots - p_n = 0$$

Muốn xác định các hệ số của đa thức đặc tính này ta dùng phương pháp Fadeev-Leverrier. Ta xét ma trận A:

$$A=\begin{pmatrix}a_{11} & a_{12} & \cdots & a_{1n}\\ a_{21} & a_{22} & \cdots & a_{2n}\\ \cdots & \cdots & \cdots & \cdots\\ a_{n1} & a_{n2} & \cdots & a_{nn}\end{pmatrix}$$

Ta gọi vết của ma trận A là số:

$$vet(A)=a_{11}+a_{22}+\ldots+a_{nn}$$

Khi đó tham số p_i của $P_n(\lambda)$ được các định như sau:

$p_1 = vet(B_1)$ với $B_1 = A$
$p_2 = (1/2)vet(B_2)$ với $B_2 = A(B_1 - p_1E)$
$p_3 = (1/3)vet(B_3)$ với $B_3 = A(B_2 - p_2E)$
.....

Chương trình tính các hệ số p_i như sau:

Chương trình 9-4

```
// Faddeev_Leverrier;
#include <stdio.h>
#include <conio.h>
#include <ctype.h>

#define max 50

void main()
  {

        int i,j,k,m,n,k1,t;
        float vet,c1,d;
        char tl;
        float p[max];
        float a[max][max],b[max][max],c[max][max],b1[max][max];

        clrscr();
        printf("Cho bac cua ma tran n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a : \n");
```

```
for (i=1;i<=n;i++)
  for (j=1;j<=n;j++)
          {
            printf("a[%d][%d] = ",i,j );
            scanf("%f",&a[i][j]);
          }
printf("\n");
clrscr();
printf("Ma tran ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%10.5f",a[i][j]);
          printf("\n");
  }
t=1;
flushall();
while (t)
  {
          printf("\n");
          printf("Co sua ma tran khong(c/k)?");
          scanf("%c",&tl);
          if (toupper(tl)=='C')
            {
                    printf("Cho chi so hang can sua : ");
                    scanf("%d",&i);
                    printf("Cho chi so cot can sua : ");
                    scanf("%d",&j);
                    printf("a[%d][%d] = ",i,j);
                    scanf("%f",&a[i][j]);
                    flushall();
            }
          if (toupper(tl)=='K')
            t=0;
  }
printf("Ma tran ban dau");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%10.5f",a[i][j]);
          printf("\n");
  }
for (i=1;i<=n;i++)
  for (j=1;j<=n;j++)
          b[i][j]=a[i][j];
for (k=1;k<=n-1;k++)
  {
          vet=0.0;
```

```
                for (i=1;i<=n;i++)
                  vet+=b[i][i];
                p[k]=vet/k;
                for (i=1;i<=n;i++)
                  for (j=1;j<=n;j++)
                        {
                         if (j!=i)
                                c[i][j]=b[i][j];
                         if (j==i)
                                c[i][j]=b[i][j]-p[k];
                        }
                for (i=1;i<=n;i++)
                  for (j=1;j<=n;j++)
                        {
                         b[i][j]=0.0;
                         for (k1=1;k1<=n;k1++)
                                b[i][j]+=a[i][k1]*c[k1][j];
                        }
          }
        vet=0.0;
        for (i=1;i<=n;i++)
          vet+=b[i][i];
        p[n]=vet/n;
        printf("\n");
        printf("Cac he so cua da thuc dac trung\n");
        printf("\n");
        d=1.0;
        printf("%6.2f",d);
        for (i=1;i<=n;i++)
          {
                c1=-p[i];
                printf("%5c%6.2f",' ',c1);
          }
        getch();
}
```

2.Phương pháp Mises: Thuật toán Mises tìm giá trị riêng lớn nhất của một ma trận A. Nếu ma trận A là thực và và mỗi trị riêng bội k có đủ k vec tơ riêng độc lập tuyến tính thì việc tính toán sẽ cho ta giá trị riêng lớn nhất.

Một vectơ V bất kì có thể được viết dưới dạng:

$$V = v_1X_1 + v_2X_2 + \cdots + v_nX_n = \sum_{i=1}^{n} v_iX_i \qquad (5)$$

Trong đó $X_1,X_2,..,X_n$ là các vec tơ riêng tương ứng với các giá trị riêng $\lambda_1,\lambda_2,\lambda_3,..,\lambda_n$ và $v_1,v_2,v_3,...,v_n$ là các hằng số.

Khi nhân A với V ta có:

$AV = Av_1X_1 + Av_2X_2 + ...+ Av_nX_n$

do: $Av_1X_1 = v_1AX_1 = v_1\lambda_1X_1$; $Av_2X_2 = v_2AX_2 = v_2\lambda_2X_2$ v.v.

Vậy nên: $AV = v_1\lambda_1X_1 + v_2\lambda_2X_2 + ...+ v_n\lambda_nX_n$

$$AV = \sum_{i=1}^{n} v_i A_i X_i = \sum_{i=1}^{n} v_i \lambda_i X_i$$

Lại nhân biểu thức trên với A ta có:

$$A^2V = v_1\lambda_1 AX_1 + v_2\lambda_2 AX_2 + ... + v_n\lambda_n AX_n$$
$$= v_1\lambda^2{}_1X_1 + v_2\lambda_2{}^2X_2 + ... + v_n\lambda_n{}^2X_n$$

và tiếp đến lần thứ p ta có:

$$A^pV = \sum_{i=1}^{n} v_i\lambda_i^p X_i = v_1\lambda_1^p X_1 + v_1\lambda_2^p X_2 + \cdots + v_n\lambda_n^p X_n$$

Lấy $\lambda^p{}_1$ làm thừa số chung ta có:

$$A^pV = \lambda_1^p\left[v_1X_1 + v_2\left(\frac{\lambda_2}{\lambda_1}\right)^p X_2 + v_3\left(\frac{\lambda_3}{\lambda_1}\right)^p X_3 + \cdots + v_n\left(\frac{\lambda_n}{\lambda_1}\right)^p X_n\right]$$

Tương tự ta có:

$$A^{p+1}V = \lambda_1^{p+1}\left[v_1X_1 + v_2\left(\frac{\lambda_2}{\lambda_1}\right)^{p+1} X_2 + v_3\left(\frac{\lambda_3}{\lambda_1}\right)^{p+1} X_3 + \cdots + v_n\left(\frac{\lambda_n}{\lambda_1}\right)^{p+1} X_n\right]$$

Khi p rất lớn,vì $\lambda_1 > \lambda_2 > \lambda_3 >,...,\lambda_n$ nên:

$$\left(\frac{\lambda_i}{\lambda_1}\right) \to 0 \quad \text{khi} \quad p \to \infty$$

Do đó:

$$\lim_{p\to\infty} A^pV = \lambda_1^p v_1X_1 \qquad (6)$$

$$\lim_{p\to\infty} A^{p+1}V = \lambda_1^{p+1} v_1X_1$$

nghĩa là khi p đủ lớn thì:

$$A^pV = \lambda_1^p v_1X_1$$

$$A^{p+1}V = \lambda_1^{p+1} v_1X_1$$

do đó: $A^{p+1}V = \lambda_1 A^pV$

hay: $A\left(A^pV\right) = \lambda_1 A^pV$

Như vậy A^pV là véc tơ riêng của A ứng với λ_1 còn giá trị riêng λ_1 sẽ là:

$$\lim_{p\to\infty} \frac{A^{p+1}V}{A^pV} = \lambda_1$$

Trong thực tế để tránh vượt quá dung lượng bộ nhớ khi λ_1 khá lớn,các vectơ V_k được chuẩn hoá sau mỗi bước bằng cách chia các phần tử của nó cho phần tử lớn nhất m_k và nhận được vectơ V'_k

Như vậy các bước tính sẽ là:

- cho một vec tơ V bất kì (có thể là $V = \{1,1,1,...,1\}^T$)
- tính $V_1 = AV$ và nhận được phần tử lớn nhất là m_{1j} từ đó tính tiếp $V'_1 = V_1/m_{1j}$

Một cách tổng quát,tại lần lặp thứ p ta nhận được vectơ V_p và phần tử lớn nhất m_{pj} thì $V'_p = V_p/ m_{pj}$.

- tính $V_{p+1} = AV'_p$ với $v_{p+1,j}$ là phần tử thứ j của V_{p+1}.Ta có:

$$\begin{cases} \lim_{p\to\infty} V'_p = X_1 \\ \lim_{p\to\infty} v_{p+1,j} = \lambda_1 \end{cases}$$

Ví dụ: Tìm giá trị riêng lớn nhất và vec tơ riêng tương ứng của ma trận:

$$A = \begin{pmatrix} 17 & 24 & 30 & 17 \\ 8 & 13 & 20 & 7 \\ 2 & 10 & 8 & 6 \\ -23 & -43 & -54 & -26 \end{pmatrix}$$

Chọn V= $\{1,1,1,1\}^T$ ta tính được

	V	$V_1 = AV$	V'_1	$V_2 = AV'_1$	V'_2
	1	88	-0.6027	-6.4801	-0.5578
	1	48	-0.3288	-5.6580	-0.4870
	1	26	-0.1781	0.0818	0.0070
	1	-146	1	11.6179	1
λ				11.6179	
	$V_3 = AV'_2$	V'_3	$V_4 = AV'_3$	V'_4	$V_5 = AV'_4$
	-3.9594	-0.5358	-3.6823	-0.5218	-3.5718
	-3.6526	-0.4942	-3.5196	-0.4987	-3.4791
	0.0707	0.0096	0.0630	0.0089	0.0408
	7.3902	1	7.0573	1	6.9638
λ	7.3902		7.0573		6.9638

	V'_5	$V_6 = AV'_5$	V'_6	$V_7 = AV'_6$	V'_7
	-0.5129	-3.5341	-0.5075	-3.5173	-0.5043
	-0.4996	-3.4809	-0.4999	-3.4868	-0.5000
	0.0059	0.0250	0.0036	0.0147	0.0021
	1	6.9634	1	6.9742	1
λ		6.9634		6.9742	

Dùng thuật toán trên ta có chương trình sau:

Chương trình 9-5

```
#include <conio.h>#include <stdio.h>#include <math.h>#include <stdlib.h>#include
<ctype.h>#define max  50void main() { int i,j,k,n,t;  char tl;  float  t0,t1,epsi,s;
    float a[max][max];
    float x0[max],x1[max];

    clrscr();
    printf("Phuong phap lap luy thua tim tri rieng lon nhat\n");
    printf("Cho so hang va cot cua ma tran n = ");
    scanf("%d",&n);
    printf("Cho cac phan tu cua ma tran a : \n");
    for (i=1;i<=n;i++)
      for (j=1;j<=n;j++)
```

```
            {
              printf("a[%d][%d] = ",i,j);
              scanf("%f",&a[i][j]);
            }
    printf("\n");
    printf("Ma tran ban da nhap\n");
printf("\n");
    for (i=1;i<=n;i++)
      {
            for (j=1;j<=n;j++)
              printf("%15.5f",a[i][j]);
       printf("\n");
      }
    flushall();
    t=1;
    while (t)
      {
            printf("\nCo sua ma tran khong(c/k)?");
            scanf("%c",&tl);
            if (toupper(tl)=='C')
              {
                    printf("Cho chi so hang can sua : ");
                    scanf("%d",&i);
                    printf("Cho chi so cot can sua : ");
                    scanf("%d",&j);
                    printf("a[%d][%d] = ",i,j);
                    scanf("%f",&a[i][j]);
              }
            if (toupper(tl)=='K')
              t=0;
      }
    epsi=1e-5;
    printf("\nMa tran ban dau\n");
    printf("\n");
    for (i=1;i<=n;i++)
      {
            for (j=1;j<=n;j++)
              printf("%15.5f",a[i][j]);
            printf("\n");
      }
    printf("\n");
    for (i=1;i<=n;i++)
       x0[i]=1;
    k=1;
    t=0;
    t1=0;
    do
       {
            t0=t1;
            for (i=1;i<=n;i++)
```

```
                {
                        x1[i]=0;
                        for (j=1;j<=n;j++)
                          x1[i]=x1[i]+a[i][j]*x0[j];
                }
              s=0;
              j=0;
              for (i=1;i<=n;i++)
                if (s<fabs(x1[i]))
                        {
                         j=i;
                         s=fabs(x1[i]);
                        }
              t1=x1[j];
              for (i=1;i<=n;i++)
                x1[i]=x1[i]/t1;
              if (fabs(t1-t0)<epsi)
                {
                        printf("Da thuc hien %d buoc lap\n",k);
                        printf("Gia tri rieng lon nhat Vmax = %15.5f\n",t1);
                        printf("Vec to rieng tuong ung\n");
                        for (i=1;i<=n;i++)
                          printf("%.5f\n",x1[i]);
                        t=1;
                }
              if (fabs(t1-t0)>epsi)
                {
                        for (i=1;i<=n;i++)
                          x0[i]=x1[i];
                        k=k+1;
                }
              if (k>max)
                t=1;
        }
      while(t==0);
      getch();
}
```

Dùng chương trình này tính gía trị riêng và vec tơ riêng của ma trận:

$$\begin{pmatrix} 2 & -1 & 0 \\ 9 & 4 & 6 \\ -8 & 0 & -3 \end{pmatrix}$$

ta nhận được giá trị riêng là 3.0000 và vec tơ riêng là x = { -0.75 ; 0.75 ; 1 }^{T}

Như chúng ta đã nói trước đây,phương pháp Mises (hay còn gọi là phương pháp lặp lũy thừa) chỉ cho phép tìm giá trị riêng lớn nhất và vec tơ riêng tương ứng của ma trận.Để xác định các giá trị riêng khác,ma trận A được biến đổi thành một ma trận khác A_1 mà các giá trị riêng là $\lambda_2 > \lambda_3 > ... > \lambda_n$.Phương pháp này gọi là phương pháp xuống thang.Sau đây là phương pháp biến đổi ma trận:

Giả sử X_1 là vec tơ riêng của ma trận A tương ứng với giá trị riêng λ_1 và W_1 là vec tơ riêng của ma trận A^T tương ứng với giá trị riêng λ_1.Từ định nghĩa $AX_1 = \lambda_1 X_1$ ta viết:

$$(A - \lambda E)X_1 = 0$$

Ta tạo ma trận A_1 dạng:

$$A_1 = A - \frac{\lambda_1}{W_1^T X_1} X_1 W_1^T \qquad (7)$$

Ta chú ý là $X_1 W_1^T$ là một ma trận còn $W_1^T X_1$ là một con số.Khi nhân hai vế của biểu thức (7) với X_1 và chý ý đến tính kết hợp của tích các ma trận ta có:

$$\begin{aligned} A_1 X_1 &= AX_1 - \frac{\lambda_1}{W_1^T X_1} X_1 W_1^T X_1 \qquad (8)\\ &= AX_1 - \lambda_1 X_1 \frac{W_1^T X_1}{W_1^T X_1} \\ &= AX_1 - \lambda_1 X_1 \\ &= 0 \end{aligned}$$

A_1 chấp nhận giá trị riêng bằng không.

Nếu X_2 là vec tơ riêng tương ứng với giá trị riêng λ_2,thì khi nhân A_1 với X_2 ta có:

$$\begin{aligned} A_1 X_2 &= AX_2 - \frac{\lambda_1}{W_1^T X_1} X_1 W_1^T X_2 \qquad (9)\\ &= AX_2 - \lambda_1 X_1 \frac{W_1^T X_2}{W_1^T X_1} \end{aligned}$$

Theo định nghĩa vì W_1 là vectơ riêng của A^T nên:

$$\lambda_1 W_1 = A^T W_1 \qquad (10)$$

Mặt khác do:

$$(AX)^T = X^T A^T \text{ và } (A^T)^T = A$$

Nên khi chuyển vị (10) ta nhận được:

$$(A^T W_1)^T = \lambda_1 W^T_1$$

Hay:

$$W_1^T A = \lambda_1 W_1^T \qquad (11)$$

Khi nhân (11) với X_2 ta có:

$$\lambda_1 W_1^T X_2 = W_1^T A X_2$$

và do định nghĩa:

$$AX_2 = \lambda_2 X_2$$

nên:

$$\lambda_1 W_1^T X_2 = W_1^T \lambda_2 X_2$$

vậy thì:

$$(\lambda_1 - \lambda_2)\, W_1^T X_2 = 0$$

khi $\lambda_1 \neq \lambda_2$ thì:

$$W_1^T X_2 = 0 \qquad (12)$$

Cuối cùng thay (12) vào (9) ta có:

$$A_1 X_2 = AX_2 = \lambda_2 X_2$$

Như vậy λ_2 là giá trị riêng lớn nhất của ma trận A_1 và như vậy có thể áp dụng thuật toán này để tìm các giá trị riêng còn lại của ma trận.Các bước tính toán như sau

- khi đã có λ_1 và X_1 ta tìm W_1 là vec tơ riêng của A^T ứng với giá trị riêng λ_1 (ví dụ tìm W_1 bằng cách giải phương trình $(A^T - \lambda_1 E)W_1 = 0$).Từ đó tính ma trận A_{12} theo (7).

- tìm giá trị riêng và vec tơ riêng của A_1 bằng cách lặp công suất và cứ thế tiếp tục và xuống thang (n-1) lần ta tìm đủ n giá trị riêng của ma trận A.

Ví dụ: Tìm giá trị riêng và vectơ riêng của ma trận sau:

$$A = \begin{pmatrix} 17 & 24 & 30 & 17 \\ 8 & 13 & 20 & 7 \\ 2 & 10 & 8 & 6 \\ -23 & -43 & -54 & -26 \end{pmatrix}$$

Ta đã tìm được giá trị riêng lớn nhất $\lambda_1 = 7$ và một vectơ riêng tương ứng:

$X_1 = \{ 1,1,0,-2\}^T$.

Ma trận A^T có dạng:

$$A^T = \begin{pmatrix} 17 & 8 & 2 & -23 \\ 24 & 13 & 10 & -43 \\ 30 & 20 & 8 & -54 \\ 17 & 7 & 6 & -26 \end{pmatrix}$$

và theo phương trình $A^T - \lambda_1 E)W_1 = 0$ ta tìm được vectơ $W_1 = \{293,695,746,434\}^T$

Ta lập ma trận mới A_1 theo (7):

$$\lambda_1 \frac{X_1 W_1^T}{W_1^T X_1} = \frac{7}{120} \begin{pmatrix} 293 & 695 & 746 & 434 \\ 293 & 695 & 746 & 434 \\ 0 & 0 & 0 & 0 \\ -586 & -1390 & -1492 & -868 \end{pmatrix}$$

và:

$$A_1 = \begin{pmatrix} -0.0917 & -16.5417 & -13.5167 & -8.3167 \\ -9.0917 & -27.5417 & -23.5167 & -18.3167 \\ 2 & 10 & 8 & 6 \\ 11.1833 & 38.0833 & 33.0333 & 24.6333 \end{pmatrix}$$

Từ ma trận A_1 ta tìm tiếp được λ_2 theo phép lặp luỹ thừa và sau đó lại tìm ma trận A_3 và tìm giá trị riêng tương ứng.

Chương trình lặp tìm các giá trị riêng và vec tơ riêng của ma trận như sau:

Chương trình 9-6

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  50

void main()
 {
        float a[max][max],vv[max][max],at[max][max];
        float x[max],y[max],vd[max];
        int i,j,k,n,l,t;
        float vp,v1,z,epsi,va,ps;
        char tl;

        clrscr();
    epsi=0.000001;
        printf("Cho bac cua ma tran n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a : \n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
```

```
        {
          printf("a[%d][%d] = ",i,j);
          scanf("%f",&a[i][j]);
        }
printf("\n");
clrscr();
printf("Ma tran ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%15.5f",a[i][j]);
        printf("\n");
  }
t=1;
flushall();
while (t)
  {
        printf("\n");
        printf("Co sua ma tran khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
            {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("Cho chi so cot can sua : ");
                scanf("%d",&j);
                printf("a[%d][%d] = ",i,j);
                scanf("%f",&a[i][j]);
          }
        if (toupper(tl)=='K')
          t=0;
  }
for (l=1;l<=n;l++)
  {
        for (i=1;i<=n;i++)
          x[i]=1;
        vp=1.23456789;
        k=0;
        for (k=1;k<=40;k++)
          {
                for (i=1;i<=n;i++)
                  {
                        y[i]=0;
                        for (j=1;j<=n;j++)
                          y[i]=y[i]+a[i][j]*x[j];
                  }
                v1=y[1]/x[1];
                z=0;
                for (i=1;i<=n;i++)
```

```
            if (fabs(y[i])>z)
                    z=y[i];
          for (i=1;i<=n;i++)
            x[i]=y[i]/z;
          if (fabs(vp-v1)<epsi)
            break;
          vp=v1;
  }
  {
          printf("Gia tri rieng : %9.6f\n",v1);
          printf("Vec to rieng : \n");
            for (i=1;i<=n;i++)
                    printf("%.5f\n",x[i]);
          printf("\n");
          getch();
  }
vd[l]=v1;
va=v1;
for (i=1;i<=n;i++)
  vv[l][i]=x[i];
for (i=1;i<=n;i++)
  for (j=1;j<=n;j++)
          at[i][j]=a[j][i];
for (i=1;i<=n;i++)
  x[i]=1;
vp=1.23456;
k=0;
for (k=1;k<=40;k++)
  {
          for (i=1;i<=n;i++)
            {
                    y[i]=0;
                    for (j=1;j<=n;j++)
                      y[i]=y[i]+at[i][j]*x[j];
            }
          v1=y[1]/x[1];
          z=0;
          for (i=1;i<=n;i++)
            if (fabs(y[i])>z)
                    z=y[i];
          for (i=1;i<=n;i++)
            x[i]=y[i]/z;
          if (fabs(vp-v1)<epsi)
            break;
          vp=v1;
  }
if (fabs(vp-v1)>epsi)
  {
          printf("Khong hoi tu sau 40 lan lap\n");
          getch();
```

```
                exit(1);
              }
            if (fabs(va-v1)>3*epsi)
                {
                printf("Co loi\n");
                getch();
                exit(1);
              }
            ps=0;
            for (i=1;i<=n;i++)
              ps=ps+x[i]*vv[l][i];
            ps=v1/ps;
            for (i=1;i<=n;i++)
              for (j=1;j<=n;j++)
                a[i][j]=a[i][j]-ps*vv[l][i]*x[j];
        }
}
```

Do (6) ®óng víi mäi n nªn cho n = 1 , 2 , 3 , . . . ta cã :

$$|x_2 - x_1| \le q|x_1 - x_o|$$
$$|x_3 - x_2| \le q|x_2 - x_1|$$
. . . . . .
$$|x_{n+1} - x_n| \le q|x_n - x_{n-1}|$$

§iÒu nµy cã nghÜa lµ d·y $x_{i+1} - x_i$, mét c¸ch gÇn ®óng,lµ mét cÊp sè nh©n . Ta còng coi r»ng d·y $x_n - y$ víi y lµ nghiÖm ®óng cña (1) , gÇn ®óng nh- mét cÊp sè nh©n cã c«ng sai q . Nh- vËy :

$$\frac{x_{n+1} - y}{x_n - y} = q < 1 \qquad (7)$$

hay : $$x_{n+1} - y = q(x_n - y) \qquad (8)$$

T-¬ng tù ta cã : $$x_{n+2} - y = q(x_{n+1} - y) \qquad (9)$$

Tõ (8) vµ (9) ta cã :

$$q = \frac{x_{n+2} - x_{n+1}}{x_{n+1} - x_n} \qquad (10)$$

Thay gi¸ trÞ cña q võa tÝnh ë (10) vµo biÓu thøc cña q ë trªn ta cã :

$$y = x_n - \frac{(x_n - x_{n+1})^2}{x_n - 2x_{n+1} + x_{n+1}} \qquad (11)$$

C«ng thøc (11) ®-îc gäi lµ c«ng thøc ngo¹i suy Adam.Nh- vËy theo (11) tr-íc hÕt ta dïng ph-¬ng ph¸p lÆp ®Ó tÝnh gi¸ trÞ gÇn ®óng x_{n+2}, x_{n+1}, x_n cña nghiÖm vµ sau ®ã theo (11) t×m ®-îc nghiÖm víi sai sè nhá h¬n.

§Ó lµm vÝ dô chóng ta xÐt ph-¬ng tr×nh :

$$\ln x - x^2 + 3 = 0$$

Ta ®-a vÒ d¹ng lÆp :

$$x = \sqrt{\ln(x) + 3}$$

$$f'(x) = \frac{1}{2x\sqrt{\ln x + 3}}$$

PhÐp lÆp héi tô trong ®o¹n [0.3,∞].Ta cho $x_1 = 1$ th× tÝnh ®-îc :

$x_2 = 1{,}7320508076$
$x_3 = 1.883960229$
$x_4 = 1.90614167$
$y = 1.909934347$

§Ó gi¶m sai sè ta cã thÓ lÆp nhiÒu lÇn

Chương trình 8-9

```
//phuong phap Aitken
#include <conio.h>
#include <stdio.h>
#include <math.h>

#define  m  5

void main()
  {
```

```
        float x[m];
        float epsi,n,y;
        int i,z;
        float f(float);

        clrscr();
        printf("Cho tri so ban dau x[1] = ");
        scanf("%f",&x[1]);
        printf("Cho tri so sai so epsilon = ");
        scanf("%f",&epsi);
        printf("\n");
        printf( "Ngoai suy Aitken cua ham\n");
        z=0;
        while (z<=20)
            {
                for (i=2;i<=4;i++)
                  x[i]=f(x[i-1]);
                n=x[4]-2*x[3]+x[2];
                if ((fabs(n)<1e-09)||(fabs(x[1]-x[2])<epsi*fabs(x[1])))
                  z=20;
                else
                  {
                        y=x[2]-(x[3]-x[2])*(x[3]-x[2])/n;
                        if (z>20)
                          printf("Khong hoi tu sau hai muoi lan lap\n");
                        x[1]=y;
                  }
                z=z+1;
          }
        printf("Nghiem cua phuong trinh y = %.6f",y);
        getch();
  }

float f(float x)
  {
        float s=sqrt(log(x)+3);
        return(s);
  }
```

Víi gi¸ trÞ ban ®Çu lµ 1 vµ sai sè lµ 1e-8,ch-¬ng tr×nh cho kÕt qu¶ y = 1.9096975944

§10.PHƯƠNG PHÁP BAIRSTOW

Nguyªn t¾c cña ph-¬ng ph¸p Bairstow lµ trÝch tõ ®a thøc $P_n(x)$ mét tam thøc $Q_2(x) = x^2 - sx + p$ mµ ta cã thÓ tÝnh nghiÖm thùc hay nghiÖm phøc cña nã mét c¸ch ®¬n gi¶n b»ng c¸c ph-¬ng ph¸p ®· biÕt.

ViÖc chia ®a thøc $P_n(x)$ cho tam thøc $Q_2(x)$ ®-a tíi kÕt qu¶ :

$$P_n(x) = Q_2(x).P_{n-2}(x) + R_1(x)$$

víi $$P_n(x) = a_o x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_n$$

$$Q_2(x) = x^2 - sx + p$$
$$P_{n-2}(x) = b_o x^{n-2} + b_1 x^{n-3} + b_2 x^{n-4} + ... + b_{n-2}$$
$$R_1(x) = \alpha x + \beta$$

§Ó cã ®-îc mét th-¬ng ®óng,cÇn t×m c¸c gi¸ trÞ cña s vµ p sao cho $R_1(x) = 0$ (nghÜa lµ α vµ β triÖt tiªu).Víi s vµ p ®· cho,c¸c hÖ sè b cña ®a thøc $P_{n-2}(x)$ vµ c¸c hÖ sè α vµ β ®-îc tÝnh b»ng ph-¬ng ph¸p truy håi.C¸c c«ng thøc nhËn ®-îc khi khai triÓn biÓu thøc $P_n(x) = Q_2(x).P_{n-2}(x) + R_1(x)$ vµ s¸p xÕp l¹i c¸c sè h¹ng cïng bËc :

$$a_o x^n + a_1 x^{n-1} + a_2 x^{n-2} + ... + a_n = (x^2 - sx + p)(b_o x^{n-2} + b_1 x^{n-3} + b_2 x^{n-4} + ... + b_{n-2})$$

Sè h¹ng bËc	HÖ sè cña $P_n(x)$	HÖ sè cña $Q_2(x).P_{n-2}(x)$
x^n	a_o	b_o
x^{n-1}	a_1	$b_1 - sb_o$
x^{n-2}	a_2	$b_2 - sb_1 + pb_o$
.....	.....	.....
x^{n-k}	a_k	$b_k - sb_{k-1} + pb_{k-2}$
x	a_{n-1}	$\alpha - sb_{n-2} + pb_{n-3}$
x^o	a_n	$\beta + pb_{n-2}$

Nh- vËy :
$$b_o = a_o \qquad (1)$$
$$b_1 = a_1 + sb_o$$
$$b_2 = a_2 + sb_1 - pb_o$$
$$.....$$
$$b_k = a_k + sb_{k-1} - pb_{k-2}$$
$$\alpha = a_{n-1} + sb_{n-2} - pb_{n-3}$$
$$\beta = a_n - pb_{n-2}$$

Chóng ta nhËn thÊy r»ng α ®-îc tÝnh to¸n xuÊt ph¸t tõ cïng mét c«ng thøc truy håi nh- c¸c hÖ sè b_k vµ t-¬ng øng víi hÖ sè b_{n-1}

$$b_{n-1} = a_{n-1} + sb_{n-2} - pb_{n-3} = \alpha$$

HÖ sè b_n lµ :

$$b_n = a_n + sb_{n-1} - pb_{n-2} = sb_{n-1} + \beta$$

vµ cuèi cïng :

$$R_1(x) = \alpha x + \beta = b_{n-1}(x - s) + b_n$$

Ngoµi ra c¸c hÖ sè b_i phô thuéc vµo s vµ p vµ b©y giê chóng ta cÇn ph¶i t×m c¸c gi¸ trÞ ®Æc biÖt s^* vµ p^* ®Ó cho b_{n-1} vµ b_n triÖt tiªu.Khi ®ã $r_1(x) = 0$ vµ nghiÖm cña tam thøc $x^2 - s^*x + p^*$ x sÏ lµ nghiÖm cña ®a thøc $P_n(x)$.Ta biÕt r»ng b_{n-1} vµ b_n lµ hµm cña s vµ p :

$$b_{n-1} = f(s,p)$$
$$b_n = g(s,p)$$

ViÖc t×m s^* vµ p^* ®-a ®Õn viÖc gi¶i hÖ ph-¬ng tr×nh phi tuyÕn:

$$\begin{cases} f(s,p)=0 \\ g(s,p)=0 \end{cases}$$

Ph-¬ng tr×nh nµy cã thÓ gi¶i dÔ dµng nhê ph-¬ng ph¸p Newton.ThËt vËy víi mét ph-¬ng tr×nh phi tuyÕn ta cã c«ng thøc lÆp :

$$x_{i+1} = x_i - f(x_i)/f'(x_i)$$

hay
$$f'(x_i)(x_{i+1} - x_i) = -f(x_i)$$

Víi mét hÖ cã hai ph-¬ng tr×nh,c«ng thøc lÆp trë thµnh:

$$J(X_i)(X_{i+1} - X_i) = -F(X_i)$$

víi $X_i = \{ s_i, p_i \}^T$ vµ $X_{i+1} = \{ s_{i+1}, p_{i+1} \}^T$

$$F(X_i) = \begin{vmatrix} f(s_i, p_i) \\ g(s_i, p_i) \end{vmatrix}$$

$$J(X_i) = \begin{vmatrix} \frac{\partial f}{\partial s} & \frac{\partial f}{\partial p} \\ \frac{\partial g}{\partial s} & \frac{\partial g}{\partial p} \end{vmatrix}$$

Quan hÖ : $J(X_i)\Delta X = -F(X_i)$ víi $\Delta X = \{s_{i+1} - s_i, p_{i+1} - p_i\}^T$ t-¬ng øng víi mét hÖ ph-¬ng tr×nh tuyÕn tÝnh hai Èn sè $\Delta s = s_{i+1} - s_i$ vµ $\Delta p = p_{i+1} - p_i$:

$$\begin{cases} \frac{\partial f}{\partial s}\Delta s + \frac{\partial f}{\partial p}\Delta p = -f(s_i, p_i) \\ \frac{\partial g}{\partial s}\Delta s + \frac{\partial g}{\partial p}\Delta p = -g(s_i, p_i) \end{cases}$$

Theo c«ng thøc Cramer ta cã :

$$\Delta s = \frac{-f\frac{\partial g}{\partial p} + g\frac{\partial f}{\partial p}}{\delta}$$

$$\Delta p = \frac{-g\frac{\partial f}{\partial s} + f\frac{\partial g}{\partial s}}{\delta}$$

$$\delta = \frac{\partial f}{\partial s}\frac{\partial g}{\partial p} - \frac{\partial f}{\partial p}\frac{\partial g}{\partial s}$$

§Ó dïng ®-îc c«ng thøc nµy ta cÇn tÝnh ®-îc c¸c ®¹o hµm $\frac{\partial f}{\partial s}, \frac{\partial f}{\partial p}, \frac{\partial g}{\partial s}, \frac{\partial g}{\partial p}$.C¸c ®¹o hµm nµy ®-îc tÝnh theo c«ng thøc truy håi.

Do $b_o = a_o$ n¹n

$$\frac{\partial b_o}{\partial s} = 0 \qquad \frac{\partial b_o}{\partial p} = 0$$

$b_1 = a_1 + sb_o$ n¹n

$$\frac{\partial b_1}{\partial s} = b_o \qquad \frac{\partial b_1}{\partial p} = 0$$

$b_2 = a_2 + sb_1 - pb_o$ n¹n

$$\frac{\partial b_2}{\partial s} = \frac{\partial a_2}{\partial s} + \frac{\partial (sb_1)}{\partial s} - \frac{\partial (pb_o)}{\partial s}$$

MÆt kh¸c :

$$\frac{\partial a_2}{\partial s} = 0 \qquad \frac{\partial (sb_1)}{\partial s} = s\frac{\partial b_1}{\partial s} + b_1 \qquad \frac{\partial (pb_o)}{\partial s} = 0$$

n¹n :

$$\frac{\partial b_2}{\partial s} = b_1 + sb_o$$

$b_3 = a_3 + sb_2 - pb_1$ n¹n

$$\frac{\partial b_3}{\partial s} = b_2 + s\frac{\partial b_2}{\partial s} - p\frac{\partial b_1}{\partial s}$$

NÕu chóng ta ®Æt :

$$\frac{\partial b_k}{\partial s} = c_{k-1}$$

th× :

$$c_o = b_o \qquad (2)$$
$$c_1 = b_1 + sb_o = b_1 + sc_o$$
$$c_2 = b_2 + sc_1 - pc_o$$
$$\text{.....}$$
$$c_k = b_k + sc_{k-1} - pc_{k-2}$$
$$c_{n-1} = b_{n-1} + sc_{n-2} - pc_{n-3}$$

Nh- vËy c¸c hÖ sè còng ®-îc tÝnh theo c¸ch nh- c¸c hÖ sè b_k.Cuèi cïng víi $f = b_{n-1}$ vµ $g = b_n$ ta ®-îc:

$$\frac{\partial f}{\partial s} = c_{n-2} \quad \frac{\partial f}{\partial s} = c_{n-3} \quad \frac{\partial f}{\partial s} = c_{n-1} \quad \frac{\partial f}{\partial s} = c_{n-2}$$

$$\Delta s = \frac{b_{n-1}c_{n-2} - b_n c_{n-3}}{c_{n-1}c_{n-3} - c_{n-2}^2} \qquad (3)$$

$$\Delta p = \frac{b_{n-1}c_{n-1} - b_n c_{n-2}}{c_{n-1}c_{n-3} - c_{n-2}^2} \qquad (4)$$

Sau khi ph©n tÝch xong $P_n(x)$ ta tiÕp tôc ph©n tÝch $P_{n-2}(x)$ theo ph-¬ng ph¸p trªn C¸c b-íc tÝnh to¸n gåm :

- Chän c¸c gi¸ trÞ ban ®Çu bÊt k× s_0 vµ p_0
- TÝnh c¸c gi¸ trÞ $b_o,..,b_n$ theo (1)
- TÝnh c¸c gi¸ trÞ $c_o,...,c_n$ theo (2)
- TÝnh Δs_o vµ Δp_o theo (3) vµ (4)
- TÝnh $s_1 = s_0 + \Delta s_o$ vµ $p_1 = p_o + \Delta p_o$
- LÆp l¹i b-íc 1 cho ®Õn khi $p_{i+1} = p_i = p$ vµ $s_{i+1} = s_i = s$
- Gi¶i ph-¬ng tr×nh $x_2 - sx + p$ ®Ó t×m 2 nghiÖm cña ®a thøc
- B¾t ®Çu qu¸ tr×nh trªn cho ®a thøc $P_{n-2}(x)$

VÝ dô : T×m nghiÖm cña ®a thøc $P_4(x) = x^4 - 1.1x^3 + 2.3x^2 + 0.5x^2 + 3.3$.
Víi lÇn lÆp ban ®Çu ta chän s = -1 vµ p =1,nghÜa lµ tam thøc cã d¹ng $x^2 + x + 1$

	a_0	a_1	a_2	a_3	a_4
	1	-1.1	2.3	0.5	3.3
sb_i		-1	2.1	-3.4	0.8
$-pb_{i-1}$			-1	2.1	-3.4
b_i	1	-2.1	3.4	$-0.8 = b_{n-1}$	$0.7 = b_n$
sb_i		-1.0	3.1	-5.5	
$-pb_{i-1}$			-1.0	3.1	
c_i	1	-3.1	5.5	-3.2	

$$\Delta s = \frac{\begin{vmatrix} 0.8 & -3.1 \\ -0.7 & 5.5 \end{vmatrix}}{\begin{vmatrix} 5.5 & -3.1 \\ -3.2 & 5.5 \end{vmatrix}} = 0.11$$

$$\Delta p = \frac{\begin{vmatrix} 5.5 & 0.8 \\ -3.2 & 0.7 \end{vmatrix}}{\begin{vmatrix} 5.5 & -3.1 \\ -3.2 & 5.5 \end{vmatrix}} = 0.06$$

$s^* = -1 + 0.11 = -0.89$

$p^* = 1 + 0.06 = 1.06$

TiÕp tôc lÆp lÇn 2 víi $s_1 = s^*$ vµ $p_1 = p^*$ ta cã :

	a_0	a_1	a_2	a_3	a_4
	1	-1.1	2.3	0.5	3.3
sb_i		-0.89	1.77	-2.68	0.06
$-pb_{i-1}$			-1.06	2.11	-3.17
b_i	1	-1.99	3.01	$-0.07 = b_{n-1}$	$0.17 = b_n$
sb_i		-0.89	2.56	-4.01	
$-pb_{i-1}$			-1.0	3.1	
c_i	1	-2.88	4.51	-1.03	

$$\Delta s = \frac{\begin{vmatrix} 0.07 & -2.88 \\ -0.7 & 5.5 \end{vmatrix}}{\begin{vmatrix} 4.51 & -2.88 \\ -1.03 & 4.51 \end{vmatrix}} = -0.01$$

$$\Delta p = \frac{\begin{vmatrix} 4.51 & 0.07 \\ -1.03 & -0.17 \end{vmatrix}}{\begin{vmatrix} 4.51 & -2.88 \\ -1.03 & 4.51 \end{vmatrix}} = 0.04$$

$s^* = -0.89 - 0.01 = -0.9$

$p^* = 1.06 + 0.04 = 1.1$

Nh- vËy $P_4(x) = (x^2+0.9x+1.1)(x^2 + 2x+3)$

Ch-¬ng tr×nh sau ¸p dông lÝ thuyÕt võa nªu ®Ó t×m nghiÖm cña ®a thøc.

Chương trình 8-10

```
//phuong phap Bairstow
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define  m   10

void main()
  {
        float a[m],b[m],c[m];
        int i,n,v;
        float s,e1,t,p,q,r,p1,q1;

        clrscr();
        printf("Cho bac cua da thuc n = ");
        scanf("%d",&n);
        printf("Cho cac he so cua da thuc can tim nghiem\n");
        for (i=n;i>=0;i--)
          {
```

```
        printf("a[%d] = ",n-i);
        scanf("%f",&a[i]);
  }
printf("\n");
e1=0.0001;
if (n<=2)
    if (n==1)
        {
         printf("Nghiem cua he\n");
         printf("%.8f",(a[0]/(-a[1])));
         getch();
         exit(1);
        }
do
  {
        v=0;
        p=1;
        q=-1;
        b[n]=a[n];
        c[n]=a[n];
        do
          {
                b[n-1]=b[n]*p+a[n-1];
                c[n-1]=b[n-1]+b[n]*p;
                for (i=n-2;i>=0;i--)
                  {
                        b[i]=b[i+2]*q+b[i+1]*p+a[i];
                        c[i]=c[i+2]*q+c[i+1]*p+b[i];
                   }
                r=c[2]*c[2]-c[1]*c[3];
                p1=p-(b[1]*c[2]-b[0]*c[3])/r;
                q1=q-(b[0]*c[2]-b[1]*c[1])/r;
                if ((fabs(b[0])<e1)&&(fabs(b[1])<e1))
                  goto tt;
                v=v+1;
                p=p1;
                q=q1;
          }
        while (v<=40);
        if(v>40)
          {
                printf("Khong hoi tu sau 40 lan lap");
                getch();
                exit(1);
          }
        tt:s=p1/2;
        t=p1*p1+4*q1;
        if(t<0)
          {
                printf("Nghiem phuc\n");
```

```
                printf("%.8f+%.8fj\n",s,(sqrt(-t)/2));
                printf("%.8f-%.8fj\n",s,(sqrt(-t)/2));
                printf("\n");
              }
            else
              {
                printf("Nghiem thuc\n");
                printf("%.8f\n",(s+sqrt(t)/2));
                printf("%.8f\n",(s-sqrt(t)/2));
                printf("\n");
              }
            for (i=2;i<=n;i++)
              a[i-2]=b[i];
              n=n-2;
          }
        while ((n>2)&(r!=0.0));
        s=-a[1]/(2*a[2]);
        t=a[1]*a[1]-4*a[2]*a[0];
        if (t<0)
          {
            printf("Nghiem phuc\n");
            printf("%.8f+%.8fj\n",s,(sqrt(-t)/(2*a[2])));
            printf("%.8f-%.8fj\n",s,(sqrt(-t)/(2*a[2])));
            printf("\n");
          }
        else
          {
            printf("Nghiem thuc\n");
            printf("%.8f\n",(s-sqrt(t)/(2*a[2])));
            printf("%.8f\n",(s-sqrt(t)/(2*a[2])));
            printf("\n");
          }
        getch();
  }
```

Dïng ch-¬ng tr×nh tr^{a}n ®Ó x¸c ®Þnh nghiÖm cña ®a thøc :

$$x^6 - 2x^5 - 4x^4 + 13x^3 - 24x^2 + 18x - 4 = 0$$

ta nhËn ®-îc c¸c nghiÖm :

$x_1 = 2.61903399$

$x_2 = -2.73205081$

$x_3 = 0.732050755$

$x_4 = 0.381966055$

$x_5 = 0.500011056 + i*1.3228881$

$x_6 = 0.500011056 - i*1.3228881$

§11.HỆ PHƯƠNG TRÌNH PHI TUYẾN

Ph-¬ng ph¸p Newton cã thÓ ®-îc tæng qu¸t ho¸ ®Ó gi¶i hÖ ph-¬ng tr×nh phi tuyÕn d^{1}ng :

$$\begin{cases} f_1(x_1,x_2,x_3,...,x_2)=0 \\ f_2(x_1,x_2,x_3,...,x_2)=0 \\ f_3(x_1,x_2,x_3,...,x_2)=0 \\ ..... \\ f_n(x_1,x_2,x_3,...,x_2)=0 \end{cases}$$

hay viÕt gän h¬n d-íi d¹ng :

$$F(X) = 0$$

Trong ®ã : $X = (x_1,x_2,x_3,.....,x_n)$

Víi mét ph-¬ng tr×nh mét biÕn,c«ng thøc Newton lµ :

$$x_{i+1}=x_i-\frac{f(x_i)}{f'(x_i)}$$

hay : $f'(x_i).\Delta x = -f(x_i)$

víi $\Delta x = x_{i+1} - x_i$

§èi víi hÖ,c«ng thøc lÆp lµ :

$$J(X_i)\Delta x = -F(X_i)$$

Trong ®ã $J(X_i)$ lµ to¸n tö Jacobi.Nã lµ mét ma trËn bËc n (n - t-¬ng øng víi sè thµnh phÇn trong vect¬ X) cã d¹ng :

$$J(x_i)=\begin{vmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3}..... & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3}..... & \frac{\partial f_2}{\partial x_n} \\ . & & & \\ . & & & \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3}..... & \frac{\partial f_n}{\partial x_n} \end{vmatrix}$$

vµ $\Delta X = X_{i+1} - X_i$

Ph-¬ng ph¸p Newton tuyÕn tÝnh ho¸ hÖ vµ nh- vËy víi mçi b-íc lÆp cÇn gi¶i mét hÖ ph-¬ng tr×nh tuyÕn tÝnh (mµ biÕn lµ Δx_i) x¸c ®Þnh bëi c«ng thøc lÆp cho tíi khi vect¬ $X(x_1,x_2,x_3,.....,x_n)$ gÇn víi nghiÖm.

D-íi ®©y lµ ch-¬ng tr×nh gi¶i hÖ ph-¬ng tr×nh phi tuyÕn

$$\begin{cases} x_1^3-x_2^3-3x_1x_2x_4-8=0 \\ x_1+x_2+x_3+x_4-5=0 \\ \sqrt{25-x_1^2}+8x_3+4=0 \\ 2x_1x_2x_3-x_4+8=0 \end{cases}$$

Ma trËn ®¹o hµm ri¸ng $J(x_i)$lµ :

$$\begin{pmatrix} 3x_1^2-3x_2x_4 & -3x_2^2-3x_1x_4 & 0 & -3x_1x_2 \\ 1 & 1 & 1 & 1 \\ \frac{-x_1}{\sqrt{25-x_1^2}} & 0 & 8 & 0 \\ 2x_2x_3 & 2x_2x_3 & 2x_2x_3 & -1 \end{pmatrix}$$

Ma trËn nµy ®-îc ch-¬ng tr×nh ®äc vµo nhê thñ tôc doc.Trong thñ tôc nµy,c¸c hÖ sè a[i,5] lµ c¸c hµm $f_i(x)$.Vect¬ nghiÖm ban ®Çu ®-îc chän lµ $\{ 0,-1,-1,1\}^T$.KÕt qu¶ tÝnh cho ta : x = $\{0.01328676,-1.94647929,-1.12499779,8.05819031 \}^T$ víi ®é chÝnh x¸c 0.000001.Vect¬ sè d- r = $\{ 0.00000536,-0.00000011,-0.00000001,-0.00000006\}^T$.

Chương trình 8-11

```
//giai he pt phi tuyen
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#define n  4

float a[n+1][n+2];
float x[n+1],y[n+1];
int i,j,k,l,z,r;
float e,s,t;

void main()
  {
        void doc();
    clrscr();
        printf("Cho cac gia tri nghiem ban dau\n");
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
          }
        e=1e-6;
        z=30;
        for (r=1;r<=z;r++)
          {
                doc();
                for (k=1;k<=n-1;k++)
                  {
                        s=0 ;
                        for (i=k;i<=n;i++)
                          {
                                t=fabs(a[i][k]);
                                if (s<=t)
                                  {
                                        s=t;
                                        l=i;
                                  }
                          }
                        for (j=k;j<=n+1;j++)
                          {
                                s=a[k][j];
                                a[k][j]=a[l][j];
                                a[l][j]=s;
                          }
                        if (a[1][1]==0)
                          {
```

```
                        printf("Cac phan tu duong cheo cua ma tran bang khong");
                        getch();
                        exit(1);
                    }
                else
                    {
                        if (fabs(a[k][k]/a[1][1])<(1e-08))
                          {
                            printf("Ma tran suy bien");
                            goto mot;
                          }
                    }
                for (i=k+1;i<=n;i++)
                    {
                        if (a[k][k]==0)
                          {
                            printf("Cac phan tu duong cheo cua ma tran bang khong\n");
                            goto mot;
                          }
                        s=a[i][k]/a[k][k];
                        a[i][k]=0;
                        for (j=k+1;j<=n+1;j++)
                         a[i][j]=a[i][j]-s*a[k][j];
                    }
                y[n]=a[n][n+1]/a[n][n];
                for (i=n-1;i>=1;i--)
                    {
                        s=a[i][n+1];
                        for (j=i+1;j<=n;j++)
                          s=s-a[i][j]*y[j];
                        if (a[i][i]==0)
                          {
                            printf("Cac phan tu duong cheo cua ma tran bang khong\n");
                            goto mot;
                          }
                        y[i]=s/a[i][i];
                    }
              }
            if (r!=1)
              for (i=1;i<=n;i++)
                    {
                      if (fabs(y[i])<e*fabs(x[i]))
                        goto ba;
                    }
            for (i=1;i<=n;i++)
              x[i]=x[i]-y[i];
            printf("\n");
        }
```

```
        printf("Khong hoi tu sau %d lan lap\n",z);
        goto mot;
        clrscr();
        ba:printf("Vec to nghiem\n");
        for (i=1;i<=n;i++)
          printf("%.5f\n",(x[i]-y[i]));
        printf("\n");
        printf("Do chinh xac cua nghiem la %.5f: \n", e);
        printf("\n");
        printf("Vec to tri so du :\n");
        for (i=1;i<=n;i++)
          printf("%.5f\n",(a[i][n+1]));
        mot:printf("\n");
        getch();
  }

void doc()
  {
        a[1][1]=3*x[1]*x[1]-3*x[2]*x[4];
        a[1][2]=-3*x[2]*x[2]-3*x[1]*x[4];
        a[1][3]=0;
        a[1][4]=-3*x[1]*x[2];
        a[1][5]=x[1]*x[1]*x[1]-x[2]*x[2]*x[2]-3*x[1]*x[2]*x[4]-8;

        a[2][1]=1;
        a[2][2]=1;
        a[2][3]=1;
        a[2][4]=1;
        a[2][5]=x[1]+x[2]+x[3]+x[4]-5;

        a[3][1]=-x[1]/sqrt(25-x[1]*x[1]);
        a[3][2]=0;
        a[3][3]=8;
        a[3][4]=0;
        a[3][5]=sqrt(25-x[1]*x[1])+8*x[3]+4;

        a[4][1]=2*x[2]*x[3];
        a[4][2]=2*x[1]*x[3];
        a[4][3]=2*x[1]*x[2];
        a[4][4]=-1;
        a[4][5]=2*x[1]*x[2]*x[3]-x[4]+8;

  }
```

CHƯƠNG 10 : GIẢI HỆ PHƯƠNG TRÌNH ĐẠI SỐ TUYẾN TÍNH

§1.PHƯƠNG PHÁP GAUSS

Có nhiều phương pháp để giải một hệ phương trình tuyến tính dạng **AX = B**. Phương pháp giải sẽ đơn giản hơn nếu ma trận **A** có dạng tam giác nghĩa là có dạng :

$$\begin{pmatrix} a_{11} & 0 & 0 \\ a_{21} & a_{22} & 0 \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad \text{hay} \quad \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix}$$

Trong trường hợp đầu tiên,ma trận được gọi là ma trận tam giác dưới và trường hợp thứ hai ma trận được gọi là ma trận tam giác trên.Phương trình tương ứng với ma trận tam giác dưới có dạng tường minh là :

$$\begin{cases} a_{11}x_1 + 0x_2 + 0x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + 0x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{cases}$$

Với phương trình dạng này chúng ta sẽ giải phương trình từ trên xuống.
Chương trình giải phương trình ma trận tam giác dưới là :

Chương trình 10-1

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  10

void main()
 {
        float a[max][max];
        float b[max],x[max];
        int i,j,k,n,t;
        float s,c;
        char tl;

        clrscr();
        printf("Cho so phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                 printf("a[%d][%d] = ",i,j);
                 scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
```

```
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%15.5f",a[i][j]);
        printf("\n");
  }
printf("\n");
t=1;
flushall();
while (t)
  {
        printf("Co sua ma tran a khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("Cho chi so cot can sua : ");
                scanf("%d",&j);
                printf("a[%d][%d] = ",i,j);
                scanf("%f",&a[i][j]);
          }
        if (toupper(tl)=='K')
        t=0;
  }
printf("Ma tran a ban dau\n");
printf("\n");
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%15.5f",a[i][j]);
        printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran b\n");
for (i=1;i<=n;i++)
  {
        printf("b[%d] = ",i);
        scanf("%f",&b[i]);
  }
printf("\n");
printf("Ma tran b ma ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
flushall();
t=1;
while (t)
  {
```

```
            printf("Co sua ma tran b khong(c/k)?");
            scanf("%c",&tl);
            if (toupper(tl)=='C')
              {
                    printf("Cho chi so hang can sua : ");
                    scanf("%d",&i);
                    printf("b[%d] = ",i);
                    scanf("%f",&b[i]);
              }
            if (toupper(tl)=='K')
              t=0;
          }
        printf("\n");
        printf("Ma tran b ban dau");
        printf("\n");
        for (i=1;i<=n;i++)
          printf("%15.5f\n",b[i]);
        {
          if (a[1][1]==0)
            if (b[1]!=0)
              printf("He da cho vo nghiem\n");
            else
              {
                    printf("He da cho co vo so nghiem");
                    x[n]=c;
              }
          else
            x[1]=b[1]/a[1][1];
          for (i=2;i<=n;i++)
            {
              s=0;
              for (k=1;k<=i-1;k++)
                    s=s+a[i][k]*x[k];
              x[i]=(b[i]-s)/a[i][i];
            }
          printf("\n");
          printf("Nghiem cua he da cho la");
          printf("\n");
          for (i=1;i<=n;i++)
            printf("x[%d] = %10.5f\n",i,x[i]);
          getch();
        }
}
```

Phương trình tương ứng với ma trận tam giác trên có dạng tường minh là :

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ 0\ x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ 0\ x_1 + 0\ x_2 + a_{33}x_3 = b_3 \end{cases}$$

Với phương trình này chúng ta giải từ dưới lên.

Chương trình giải phương trình ma trận tam giác trên là :

Chương trình 10-2

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  10

void main()
  {
        float a[max][max];
        float b[max],x[max];
        int i,j,k,n,t;
        float s,c;
        char tl;


        clrscr();
        printf("Cho so phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a :\n");
        for (i=1;i<=n;i++)
            for (j=1;j<=n;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%15.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t=1;
        flushall();
        while (t)
          {
                printf("Co sua ma tran a khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
```

```
                printf("Cho chi so cot can sua : ");
                scanf("%d",&j);
                printf("a[%d][%d] = ",i,j);
                scanf("%f",&a[i][j]);
              }
          if (toupper(tl)=='K')
            t=0;
  }
printf("Ma tran a ban dau");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%15.5f",a[i][j]);
          printf("\n");
    }
printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
  {
          printf("b[%d] = ",i);
          scanf("%f",&b[i]);
  }
printf("\n");
printf("Ma tran b ma ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
flushall();
t=1;
while (t)
  {
          printf("Co sua ma tran b khong(c/k)?");
          scanf("%c",&tl);
          if (toupper(tl)=='C')
            {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
              }
          if (toupper(tl)=='K')
            t=0;
  }
printf("\n");
printf("Ma tran b ban dau\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
```

```
        printf("\n");
        {
         if (a[n][n]==0)
                if (b[n]!=0)
                  printf("He da cho vo nghiem");
                else
                  {
                        printf("He da cho co vo so nghiem");
                        x[n]=c;
                  }
         else
                 x[n]=b[n]/a[n][n];
                 for (i=n-1;i>=1;i--)
                        {
                         s=0;
                         for (k=i+1;k<=n;k++)
                                s=s+a[i][k]*x[k];
                         x[i]=(b[i]-s)/a[i][i];
                        }
                 printf("\n");
                 printf("Nghiem cua he da cho la\n");
                 printf("\n");
                 for (i=1;i<=n;i++)
                   printf("x[%d] = %10.5f\n",i,x[i]);
                 getch();
        }
}
```

Tuy nhiên, các hệ phương trình đơn giản hiếm khi gặp trong thực tế. Các hệ phương trình tuyến tính có thể biểu diễn dưới dạng tam giác nếu định thức của nó khác không, nghĩa là phương trình có nghiệm.Chúng ta biết rằng các nghiệm của hệ không đổi nếu ta thay một hàng bằng tổ hợp tuyến tính của các hàng khác.Như vậy bằng một loạt các biến đổi ta có thể đưahệ ban đầu về dạng tam giác. Đó chính là nội dung của phương pháp loại trừ Gauss. Chúng ta hãy xét hệ phương trình :

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \end{cases}$$

Nhân hàng thứ nhất với a_{21}/a_{11} ta có :

$$a_{21}x_1 + \frac{a_{21}}{a_{11}}a_{12}x_2 + \frac{a_{21}}{a_{11}}a_{13}x_3 = \frac{a_{21}}{a_{11}}b_1$$

Số hạng đầu của phương trình bằng số hạng đầu của hàng thứ hai trong hệ phương trình ban đầu.Khi trừ hàng một đã được biến đổi cho hàng 2 ta nhận được hàng 2 mới

$$0\ x_1 + (a_{22} - \frac{a_{21}}{a_{11}}a_{12})x_2 + (a_{23} - \frac{a_{21}}{a_{11}}a_{13})x_3 = b_2 - \frac{a_{21}}{a_{11}}b_1$$

Ta tiếp tục cách này để loại trừ x_1 ra khỏi hàng thứ 3.Phương trình trở thành :

$$\begin{pmatrix} a'_{11} & a'_{12} & a'_{13} \\ 0 & a'_{22} & a'_{23} \\ 0 & a'_{32} & a'_{33} \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b'_1 \\ b'_2 \\ b'_3 \end{pmatrix}$$

với $a'_{11} = a_{11}$; $a'_{12} = a_{12}$; $a'_{13} = a_{13}$; $a'_{13} = a_{13}$; $b'_1 = b_1$

$a'_{22} = a_{22} - \frac{a_{21}}{a_{11}} a_{12}$ $a'_{23} = a_{23} - \frac{a_{21}}{a_{11}} a_{13}$ $a'_{32} = a_{32} - \frac{a_{31}}{a_{11}} a_{12}$ $a'_{33} = a_{33} - \frac{a_{31}}{a_{11}} a_{13}$ $b'_2 = b_2 - \frac{a_{21}}{a_{11}} b_1$

$b'_3 = b_3 - \frac{a_{31}}{a_{11}} b_1$

Ta loại trừ số hạng chứa x_3 trong dòng thứ 3 bằng cách tương tự.Ta nhân hàng thứ 2 trong hệ $\mathbf{A'X = B'}$ với a'_{32}/a'_{22} và đem trừ đi hàng thứ 3 trong hệ mới.Như vậy số hạng chứa x_3 biến mất và ta nhận được ma trận tam giác trên.

$$\begin{pmatrix} a''_{11} & a''_{12} & a''_{13} \\ 0 & a''_{22} & a''_{23} \\ 0 & 0 & a''_{33} \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b''_1 \\ b''_2 \\ b''_3 \end{pmatrix}$$

với $a''_{11} = a'_{11}$ $a''_{12} = a'_{12}$ $a''_{13} = a'_{13}$ $b''_1 = b'_1$ $a''_{22} = a'_{22}$ $a''_{23} = a'_{23}$ $b''_2 = b'_2$

$a''_{33} = a'_{33} - \frac{a'_{32}}{a'_{22}} a'_{23}$ $b''_3 = b'_3 - \frac{a'_{33}}{a'_{22}} b'_2$

Các phép tính này chỉ thực hiện được khi $a_{11} \neq 0$ và $a'_{11} \neq 0$.

Với một hệ có n phương trình,thuật tính hoàn toàn tương tự.Sau đây là chương trình giải hệ phương trình n ẩn số bằng phương pháp loại trừ Gauss.

Chương trình 10-3

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  10

void main()
 {
        float b[max],x[max];
        float a[max][max];
        int i,j,k,n,t;
        float c,s,d;
        char tl;

        clrscr();
        printf("Cho so phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a :\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                 printf("a[%d][%d] = ",i,j);
                 scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
```

```
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%15.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t=1;
flushall();
        while (t)
          {
                printf("Co sua ma tran a khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                    {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[%d][%d] = ",i,j);
                        scanf("%f",&a[i][j]);
                    }
                if (toupper(tl)=='K')
                  t=0;
          }
        printf("Ma tran a ban dau\n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                    printf("%15.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        printf("Cho cac phan tu cua ma tran b : \n");
        for (i=1;i<=n;i++)
          {
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
          }
        printf("\n");
        printf("Ma tran b ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
            printf("b[%d] = %15.5f\n",i,b[i]);
        printf("\n");
        flushall();
        t=1;
        while (t)
          {
```

```
            printf("Co sua ma tran b khong(c/k)?");
            scanf("%c",&tl);
            if (toupper(tl)=='C')
              {
                    printf("Cho chi so hang can sua : ");
                    scanf("%f",&i);
                    printf("b[%d] = ",i);
                    scanf("%f",&b[i]);
              }
            if (toupper(tl)=='K')
              t=0;
      }
    printf("\n");
    printf("Ma tran b\n");
    for (i=1;i<=n;i++)
      printf("b[%d] = %15.5f\n",i,b[i]);
    printf("\n");
    for (k=1;k<=n-1;k++)
      {
            for (i=k+1;i<=n;i++)
              {
                    b[i]=b[i]-b[k]*a[i][k]/a[k][k];
                    for (j=k+1;j<=n;j++)
                      a[i][j]=a[i][j]-a[k][j]*a[i][k]/a[k][k];
              }
      }

    {
      if (a[n][n]==0)
            if (b[n]==0)
              printf("He da cho vo nghiem");
  else
              {
                    printf("He da cho co vo so nghiem");
                    x[n]=c;
              }
else
              x[n]=b[n]/a[n][n];
              for (i=n-1;i>=1;i--)
                    {
                      s=0;
                      for (k=i+1;k<=n;k++)
                            s=s+a[i][k]*x[k];
                      x[i]=(b[i]-s)/a[i][i];
                    }
              printf("\n");
              printf("Nghiem cua he da cho la\n");
              printf("\n");
              for (i=1;i<=n;i++)
                printf("x[%d] = %15.5f\n",i,x[i]);
```

```
        getch();
    }
}
```

§2.PHƯƠNG PHÁP GAUSS-JORDAN

Xét hệ phương trình AX=B. Khi giải hệ bằng phương pháp Gauss ta đưa nó về dạng ma trận tam giác sau một loạt biến đổi. Phương pháp khử Gauss-Jordan cải tiến khử Gauss bằng cách đưa hệ về dạng :

$$EX = B^*$$

và khi đó nghiệm của hệ chính là B^*.Trong phương pháp Gauss-Jordan mỗi bước tính phải tính nhiều hơn phương pháp Gauss nhưng lại không phải tính nghiệm.Để đưa ma trận A về dạng ma trận E tại bước thứ i ta phải có $a_{ii} = 1$ và $a_{ij}=0$.Như vậy tại lần khử thứ i ta biến đổi :

1.$a_{ij} = a_{ij}/a_{ii}$ (j=i+1,i+2,...,n)

2.k=1,2,...,n

$a_{kj} = a_{kj} - a_{ij}a_{ki}$ (j=i+1,i+2,...,n)

$b_k = b_k - b_i a_{ki}$

Ví dụ : Cho hệ $\begin{pmatrix} 8 & 4 & 2 & 0 \\ 4 & 10 & 5 & 4 \\ 2 & 5 & 6.5 & 4 \\ 0 & 4 & 4 & 9 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 24 \\ 32 \\ 26 \\ 21 \end{pmatrix}$

Biến đổi lần 1 : ta chia hàng 1 cho $a_{11} = 8$;nhân hàng 1 vừa nhận được với 4 và lấy hàng 2 trừ đi; nhân hàng 1 vừa nhận được với 2 và lấy hàng 3 trừ đi;giữ nguyên hàng 4 vì phần tử đầu tiên đã bằng 0 ta có

$$\begin{pmatrix} 1 & 0.5 & 0.25 & 0 \\ 0 & 8 & 4 & 4 \\ 0 & 4 & 6 & 4 \\ 0 & 4 & 4 & 9 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 3 \\ 20 \\ 20 \\ 21 \end{pmatrix}$$

Biến đổi lần 2 : ta chia hàng 2 cho $a_{22} = 8$;nhân hàng 2 vừa nhận được với 0.5 và lấy hàng 1 trừ đi; nhân hàng 2 vừa nhận được với 4 và lấy hàng 3 trừ đi; nhân hàng 2 vừa nhận được với 4 và lấy hàng 4 trừ đi ta có :

$$\begin{pmatrix} 1 & 0 & 0 & -0.25 \\ 0 & 1 & 0.5 & 0.5 \\ 0 & 0 & 4 & 2 \\ 0 & 0 & 2 & 7 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1.75 \\ 2.5 \\ 10 \\ 11 \end{pmatrix}$$

Biến đổi lần 3 : ta chia hàng 3 cho $a_{33} = 4$;giữ nguyên hàng 1;nhân hàng 3 vừa nhận được với 0.5 và lấy hàng 2 trừ đi; nhân hàng 3 vừa nhận được với 2 và lấy hàng 4 trừ đi ta có :

$$\begin{pmatrix} 1 & 0 & 0 & -0.25 \\ 0 & 1 & 0 & 0.25 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 6 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1.75 \\ 1.25 \\ 2.5 \\ 6 \end{pmatrix}$$

Biến đổi lần 4 : ta chia hàng 4 cho $a_{44} = 6$;nhân hàng 4 vừa nhận được với -0.25 và lấy hàng 1 trừ đi; nhân hàng 4 vừa nhận được với 0.25 và lấy hàng 2 trừ đi; nhân hàng 4 vừa nhận được với 0.5 và lấy hàng 3 trừ đi ta có :

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 2 \\ 1 \\ 2 \\ 1 \end{pmatrix}$$

và ta có ngay vec tơ nghiệm.

Chương trình10-4

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define  spt  10

void main()
 {
        float a[spt][2*spt];
        float b[spt];
        int i,j,k,n,m,t;
        float max,c;
        char tl;

        clrscr();
        printf("Cho so phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a :\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%15.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t=1;
        flushall();
```

```
while (t)
  {
        printf("Co sua ma tran a khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("Cho chi so cot can sua : ");
                scanf("%d",&j);
                printf("a[%d][%d] = ",i,j);
                scanf("%f",&a[i][j]);
          }
        if (toupper(tl)=='K')
          t=0;
  }
printf("Ma tran a\n");
printf("\n");
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%15.5f",a[i][j]);
        printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
  {
        printf("b[%d] = ",i);
        scanf("%f",&b[i]);
  }
printf("\n");
printf("Ma tran b ma ban da nhap\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %15.5f\n",i,b[i]);
printf("\n");
t=1;
flushall();
while (t)
  {
        printf("Co sua ma tran b khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
          }
```

```
        if (toupper(tl)=='K')
          t=0;
  }
printf("\n");
printf("Ma tran b\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("%15.5f\n",b[i]);
printf("\n");
t=1;
flushall();
i=1;
while (t)
  {
        if (a[i][i]==0)
          {
                max=0;
                m=i;
                for (k=i+1;k<=n;k++)
                  if (max<fabs(a[k][i]))
                        {
                          m=k;
                          max=fabs(a[i][i]);
                        }
                  if (m!=i)
                        {
                          for (j=i;j<=n;j++)
                                {
                                  c=a[i][j];
                                  a[i][j]=a[m][j];
                                  a[m][j]=c;
                                }
                          c=b[i];
                          b[i]=b[m];
                          b[m]=c;
                        }
                  if (m==i)
                        {
                          t=0;
                          printf("MA TRAN SUY BIEN");
                        }
          }
        if (a[i][i]!=0)
          {
                c=1/a[i][i];
                for (j=i;j<=n;j++)
                  a[i][j]=a[i][j]*c;
                b[i]=b[i]*c;
                for (k=1;k<=n;k++)
                  if (k!=i)
```

```
                        {
                         c=a[k][i];
                         for (j=i;j<=n;j++)
                                a[k][j]=a[k][j]-a[i][j]*c;
                         b[k]=b[k]-b[i]*c;
                        }
                  }
                i=i+1;
                if (i==(n+1))
                  t=0;
          }
        if (i==(n+1))
          {
                printf("NGHIEM CUA HE");
                printf("\n");
                for (i=1;i<=n;i++)
                  printf("x[%d] = %15.5f\n",i,b[i]);
          }
        getch();
  }
```

§3.PHƯƠNG PHÁP CHOLESKY

Trong phương pháp Cholesky một ma trận đối xứng A được phân tích thành dạng $A = R^TR$ trong đó R là một ma trận tam giác trên.Hệ phương trình lúc đó chuyển thành $AX = R^TRX = B$.Như vậy trước hết ta phân tích ma trận A thành tích hai ma trận.Sau đó giải hệ phương trình $R^TY = B$ và cuối cùng là hệ RX = Y**.**Chương trình mô tả thuật toán này được cho dưới đây :

Chương trình 10-5

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  6

void main()
  {
        float a[max][max],r[max][max];
        float b[max],x[max],y[max];
        int i,j,k,l,n,t;
        float s;
        char tl;

        clrscr();
        printf("Cho so phuong trinh n = ");
```

```
scanf("%d",&n);
printf("Cho cac phan tu cua ma tran a : \n");
for (i=1;i<=n;i++)
  for (j=1;j<=n;j++)
          {
            printf("a[%d][%d] = ",i,j);
            scanf("%f",&a[i][j]);
          }
printf("\n");
printf("Ma tran a ma ban da nhap\n");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%15.5f",a[i][j]);
          printf("\n");
  }
printf("\n");
flushall();
t=1;
while (t)
  {
          printf("Co sua ma tran a khong(c/k)?");
          scanf("%c",&tl);
          if (toupper(tl)=='C')
            {
                    printf("Cho chi so hang can sua : ");
                    scanf("%d",&i);
                    printf("Cho chi so cot can sua : ");
                    scanf("%d",&j);
                    printf("a[",i,",",j,"] = ");
                    scanf("%f",&a[i][j]);
            }
          if (toupper(tl)=='K')
            t=0;
  }
printf("Ma tran a\n");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%15.5f",a[i][j]);
          printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
  {
          printf("b[%d] = ",i);
          scanf("%f",&b[i]);
```

```
  }
printf("\n");
printf("Ma tran b ma ban da nhap\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %15.5f\n",i,b[i]);
printf("\n");
flushall();
t=1;
while (t)
  {
        printf("Co sua ma tran b khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
          }
        if (toupper(tl)=='K')
          t=0;
  }
printf("\n");
printf("Ma tran b\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %15.5f\n",i,b[i]);
printf("\n");
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
        r[i][j]=0.0;
for (i=1;i<=n;i++)
  {
        if (a[i][i]>=0)
          {
                r[i][i]=sqrt(a[i][i]);
                for (j=1+i;j<=n;j++)
                  r[i][j]=a[i][j]/r[i][i];
                for (k=i+1;k<=n;k++)
                  for (l=k;l<=n;l++)
                        a[k][l]=a[k][l]-r[i][k]*r[i][l];
          }
  }
for (k=1;k<=n;k++)
  {
        s=b[k];
        if (k!=1)
          for (i=1;i<=k-1;i++)
                s=s+r[i][k]*y[i];
```

```
            y[k]=-s/r[k][k];
      }
    for (i=n;i>=1;i--)
      {
            s=-y[i];
            if (i!=n)
              for (k=i+1;k<=n;k++)
                    s=s-r[i][k]*x[k];
              x[i]=s/r[i][i];
      }
    printf("Nghiem cua he phuong trinh la\n ");
    for (i=1;i<=n;i++)
      printf("x[%d] = %10.5f\n",i,x[i]);
    getch();
}
```

§4.PHƯƠNG PHÁP CROUT

Phương pháp Crout là một dạng của phương pháp Gauss.Với phương pháp Gauss,chúng ta biến đổi ma trận A thành một ma trận tam giác thì ở phương pháp Crout chúng ta phân tích ma trận này thành tích của ma trận tam giác trên R và ma trận tam giác dưới L.Trong ma trận L,các hệ số trên đường chéo chính bằng 1.Như vậy phương trình AX = B được viết thành :

A.X = L.R.X = B

Chúng ta đặt RX = Y

nên : LY = B

Như vậy trước hết chúng ta phân tích ma trận thành tích của L.R.Tiếp theo ta giải phương trình LY = B và sau đó giải phương trình RX = A để tìm nghiệm X.

Chương trình 10-6

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  6

void main()
  {
    float b[max],x[max],y[max];
    float a[max][max],r[max][max],l[max][max];
    int i,j,k,n,t;
    float c,tr,tl,s;
    char tloi;

    clrscr();
    printf("Cho so phuong trinh n = ");
```

```
scanf("%d",&n);
printf("Cho cac phan tu cua ma tran a : \n");
for (i=1;i<=n;i++)
  for (j=1;j<=n;j++)
          {
           printf("a[%d][%d] = ",i,j);
           scanf("%f",&a[i][j]);
          }
printf("\n");
printf("Ma tran a ma ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
                  printf("%10.5f",a[i][j]);
          printf("\n");
  }
printf("\n");
t=1;
flushall();
while (t)
  {
          printf("Co sua ma tran a khong(c/k)?");
          scanf("%c",&tloi);
          if (toupper(tloi)=='C')
            {
                  printf("Cho chi so hang can sua : ");
                  scanf("%d",&i);
                  printf("Cho chi so cot can sua : ");
                  scanf("%d",&j);
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                  flushall();
            }
          if (toupper(tloi)=='K')
            t=0;
  }
printf("Ma tran a\n");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%10.5f",a[i][j]);
          printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
  {
          printf("b[%d] = ",i);
```

```
        scanf("%f",&b[i]);
  }
printf("\n");
printf("Ma tran b ma ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
t=1;
flushall();
while (t)
  {
        printf("Co sua ma tran b khong(c/k)?");
        scanf("%c",&tloi);
        if (toupper(tloi)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
                flushall();
          }
        if (toupper(tloi)=='K')
          t=0;
  }
printf("\n");
printf("Ma tran b\n");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
        {
          r[i][j]=0.0;
          l[i][j]=0.0;
        }
for (i=1;i<=n;i++)
  {
        r[1][i]=a[1][i];
        l[i][i]=1.0;
        l[i][1]=a[i][1]/a[1][1];
  }
for (k=2;k<=n;k++)
  {
        for (j=k;j<=n;j++)
          {
                tr=0.0;
                for (i=1;i<=k;i++)
                  tr=tr+l[k][i]*r[i][j];
```

```
                r[k][j]=a[k][j]-tr;
            }
          if (k!=n)
            {
                for (i=1;i<=n;i++)
                  {
                        tl=0.0;
                        for (j=1;j<=k-1;j++)
                          tl=tl+l[i][j]*r[j][k];
                        l[i][k]=(a[i][k]-tl)/r[k][k];
                  }
            }
          else
            printf("\n");
    }
if (l[1][1]==0.0)
  if (b[1]==0.0)
        printf("He da cho vo nghiem\n");
  else
        {
          printf("He da cho co vo so nghiem\n");
          y[n]=c;
        }
else
  y[1]=b[1]/l[1][1];
for (i=2;i<=n;i++)
  {
        s=0.0;
        for (k=1;k<=i-1;k++)
          s=s+l[i][k]*y[k];
        y[i]=(b[i]-s)/l[i][i];
  }
if (r[n][n]==0.0)
  if (y[n]==0.0)
        printf("He da cho vo nghiem\n");
  else
        {
          printf("He da cho co vo so nghiem\n");
          x[n]=c;
        }
else
  x[n]=y[n]/r[n][n];
for (i=n-1;i>=1;i--)
  {
        s=0.0;
        for (k=i+1;k<=n;k++)
          s+=r[i][k]*x[k];
        x[i]=(y[i]-s)/r[i][i];
  }
printf("\n");
```

```
        printf("Nghiem cua he da cho la\n");
        printf("\n");
        for (i=1;i<=n;i++)
          printf("x[%d] = %15.5f\n",i,x[i]);
        getch();
}
```

§5.PHƯƠNG PHÁP LẶP ĐƠN

Xét hệ phương trình AX = F.Bằng cách nào đó ta đưa hệ phương trình về dạng

$$X = BX + G$$

trong đó $B = (b_{ij})_{n,n}$

$G = (g_1,g_2,...,g_n)^T$

Chọn vectơ $X = (x_1^{(o)},x_2^{(o)},....,x_n^{(o)})^T$

làm xấp xỉ thứ 0 của nghiệm đúng và xây dựng xấp xỉ

$$X^{(m+1)} = BX^{(m)} + G \quad (m = 0,1,....)$$

Người ta chứng minh rằng nếu phương trình ban đầu có nghiệm duy nhất và một trong ba chuẩn của ma trận B nhỏ hơn 1 thì dãy xấp xỉ hội tụ về nghiệm duy nhất đó.(Cho một ma trận B,chuẩn của ma trận B,kí hiệu là || B || là một trong 3 số :

$$\|B\|_1 = \max_i \sum_{j=1}^{n} b_{ij}$$

$$\|B\|_2 = \max_{ji} \sum_{i=1}^{n} b_{ij}$$

$$\|B\|_3 = (\sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}^2)^{1/2}$$

(Chuẩn của ma trận quan hệ tới sự hội tụ của phương pháp lặp)

Ví dụ chúng ta có phương trình

$$\begin{cases} 10x_1 + 2x_2 + x_3 = 10 \\ x_1 + 10x_2 + 2x_3 = 12 \\ x_1 + x_2 + 10x_3 = 8 \end{cases}$$

Chúng ta đưa phương trình về dạng :

$$\begin{cases} x_1 = -\frac{1}{5}x_2 - \frac{1}{10}x_3 + 1 \\ x_2 = -\frac{1}{10}x_1 - \frac{1}{5}x_3 + \frac{6}{5} \\ x_3 = -\frac{1}{10}x_1 - \frac{1}{10}x_2 + \frac{4}{5} \end{cases}$$

Như vậy :

$$B = \begin{bmatrix} 0 & -\frac{1}{5} & -\frac{1}{10} \\ -\frac{1}{10} & 0 & -\frac{1}{5} \\ -\frac{1}{10} & -\frac{1}{10} & 0 \end{bmatrix} \quad \text{và} \quad G = \begin{bmatrix} 1 \\ \frac{5}{6} \\ \frac{4}{5} \end{bmatrix}$$

Dễ thấy $\| B \|_1 = 3/10$; $\| B \|_2 = 3/10$ và $\| B \|_3 = 12/100$ nên phép lặp hội tụ .Chương trình lặp đơn là :

Chương trình 10-7

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  10

void main()
 {
        float a[max][max];
        float f[max],x0[max],x1[max];
        int i,j,k,n,l,t;
        float s,c,epsi;
        char tl;

        clrscr();
        printf("Cho so phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a : \n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%15.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t=1;
        flushall();
        while (t)
          {
                printf("Co sua ma tran a khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[%d][%d] = ",i,j);
```

```
                scanf("%f",&a[i][j]);
                flushall();
              }
          if (toupper(tl)=='K')
            t=0;
  }
printf("Ma tran a\n");
printf("\n");
for (i=1;i<=n;i++)
  {
          for (j=1;j<=n;j++)
            printf("%10.5f",a[i][j]);
          printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran f : \n");
for (i=1;i<=n;i++)
  {
          printf("f[%d] = ",i);
          scanf("%f",&f[i]);
  }
printf("\n");
printf("Ma tran f ma ban da nhap");
printf("\n");
for (i=1;i<=n;i++)
  printf("f[%d] = %10.5f\n",i,f[i]);
printf("\n");
t=1;
flushall();
while (t)
  {
          printf("Co sua ma tran f khong(c/k)?");
          scanf("%c",&tl);
          if (toupper(tl)=='C')
            {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("f[%d] = ",i);
                scanf("%f",&f[i]);
                flushall();
            }
          if (toupper(tl)=='K')
            t=0;
  }
printf("\n");
printf("Ma tran f");
printf("\n");
for (i=1;i<=n;i++)
  printf("f[%d] = %10.5f\n",i,f[i]);
printf("\n");
```

```
for (i=1;i<=n;i++)
  x0[i]=0.0;
x0[1]=1.0;
printf("Cho so lan lap l = ");
scanf("%d",&l);
epsi=1e-5;
for (i=1;i<=n;i++)
  {
        c=1.0/a[i][i];
        for (j=1;j<=n;j++)
          if (j!=i)
                a[i][j]*=c;
        f[i]*=c;
        a[i][i]=0.0;
  }
k=1;
t=0;
do
  {
        for (i=1;i<=n;i++)
          {
                x1[i]=f[i];
                for (j=1;j<=n;j++)
                  x1[i]=x1[i]-a[i][j]*x0[j];
          }
        s=0.0;
        for (i=1;i<=n;i++)
          s=s+fabs(x1[i]-x0[i]);
        if (s>=epsi)
          for (i=1;i<=n;i++)
                x0[i]=x1[i];
        if (s<epsi)
          {
                t=1;
                printf("\n");
                printf("Phep lap hoi tu sau %d buoc tinh",k);
                printf("\n");
                printf("NGHIEM CUA HE");
                printf("\n");
                for (i=1;i<=n;i++)
                  printf("x[%d] = %10.5f\n",i,x1[i]);
          }
        k=k+1;
        if (k>l)
          {
                t=1;
                printf("Phep lap khong hoi tu sau %d buoc tinh",k-1);
          }
  }
while (t==0);
```

```
    getch();
}
```

§6. PHƯƠNG PHÁP LẶP GAUSS-SEIDEL

Phương pháp lặp Gauss-Seidel được cải tiến từ phương pháp lặp đơn . Nội dung cơ bản của phương pháp là ở chỗ khi tính nghiệm xấp xỉ thứ (k+1) của ẩn x_i ta sở dụng các xấp xỉ thứ (k+1) đã tính của các ẩn $x_1,...,x_{i-1}$.Giả sử đã cho hệ : **AX = B** và ta có nghiệm :

$$x_i = \beta_i + \sum_{j=1}^{n} \alpha_{ij} x_j \quad (i=1,....,n)$$

Lấy xấp xỉ ban đầu tuỳ ý $x_1^{(o)}$, $x_2^{(o)}$,...., $x_n^{(o)}$ và tất nhiên ta cố gắng lấy chúng tương ứng với x_1,x_2 ,...,x_n (càng gần càng tốt). Tiếp theo ta giả sử rằng đã biết xấp xỉ thứ k $x_i^{(k)}$ của nghiệm . Theo Seidel ta sẽ tìm xấp xỉ thứ (k+1) của nghiệm theo các công thức sau :

$$x_1^{(k+1)} = \beta_1 + \sum_{j=1}^{n} \alpha_{1j} x_j^{(k)}$$

$$x_2^{(k+1)} = \beta_2 + \alpha_{21} x_1^{(k+1)} + \sum_{j=2}^{n} \alpha_{2j} x_j^{(k)}$$

.....

$$x_i^{(k+1)} = \beta_i + \sum_{j=1}^{i-1} \alpha_{ij} x_j^{(k+1)} + \sum_{j=1}^{n} \alpha_{ij} x_j^{(k)}$$

.....

$$x_{ni}^{(k+1)} = \beta_n + \sum_{j=1}^{n-1} \alpha_{ij} x_j^{(k+1)} + \alpha_{nn} x_n^{(k)}$$

Thông thường phương pháp Gauss - Seidel hội tụ nhanh hơn phương pháp lặp đon nhưng tính toán phức tạp hơn.Để dễ hiểu phương pháp này chúng ta xét một ví dụ cụ thể :

Cho hệ phương trình :

$$\begin{cases} 10x_1 + x_2 + x_3 = 12 \\ 2x_1 + 10x_2 + x_3 = 13 \\ 2x_1 + 2x_2 + 10x_3 = 14 \end{cases}$$

nghiệm đúng của hệ là (1 , 1, 1)

Ta đưa về dạng thuận tiện cho phép lặp :

$$\begin{cases} x_1 = 1,2 - 0.1x_2 - 0.1x_3 \\ x_2 = 1,3 - 0.2x_1 - 0.1x_3 \\ x_3 = 1,4 - 0.2x_2 - 0.2x_2 \end{cases}$$

Lấy $x_1^{(o)} = 1.2$; $x_2^{(o)} = 0$; $x_3^{(o)} = 0$;

Sử dụng phương pháp lặp Seidel ta có :

$$\begin{cases} x_1^1 = 1.2 - 0.1 \times 0 - 0.1 \times 0 = 1.2 \\ x_2^1 = 1.3 - 0.2 \times 1.2 - 0.1 \times 0 = 1.06 \\ x_3^1 = 1.4 - 0.2 \times 1.2 - 0.2 \times 1.06 = 0.948 \end{cases}$$

$$\begin{cases} x_1^2 = 1,2 - 0.1 \times 1.06 - 0.1 \times 0.948 = 0.9992 \\ x_2^2 = 1,3 - 0.2 \times 0.9992 - 0.1 \times 0.948 = 1.00536 \\ x_3^2 = 1,4 - 0.2 \times 0.9992 - 0.2 \times 1.00536 = 0.999098 \end{cases}$$

và cứ thế tiếp tục . Chương trình mô tả thuật toán Gauss - Seidel như sau :

Chương trình 10-8

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  6

void main()
 {
        float b[max],x[max];
        float a[max][max];
        int i,j,k,n,dem,t1;
        float t,s,d,w,epsi;
        char tl;

   clrscr();
        printf("Cho so an so n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a : \n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%10.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t1=1;
        flushall();
        while (t1)
          {
                printf("Co sua ma tran a khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[%d][%d] = ",i,j);
```

```
                scanf("%f",&a[i][j]);
                flushall();
            }
        if (toupper(tl)=='K')
          t1=0;
  }
printf("Ma tran a\n");
printf("\n");
for (i=1;i<=n;i++)
  {
        for (j=1;j<=n;j++)
          printf("%15.5f",a[i][j]);
        printf("\n");
  }
printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
  {
        printf("b[%d] = ",i);
        scanf("%f",&b[i]);
  }
printf("\n");
printf("Ma tran b");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
t1=1;
flushall();
while (t1)
  {
        printf("Co sua ma tran b khong(c/k)?");
        scanf("%c",&tl);
        if (toupper(tl)=='C')
          {
                printf("Cho chi so hang can sua : ");
                scanf("%d",&i);
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
                flushall();
          }
        if (toupper(tl)=='K')
          t1=0;
  }
printf("\n");
printf("Ma tran b");
printf("\n");
for (i=1;i<=n;i++)
  printf("b[%d] = %10.5f\n",i,b[i]);
printf("\n");
```

```
        printf("Cho so lan lap k : ");
        scanf("%d",&k);
        printf("\n");
        w=1;
        epsi=1e-8;

        for (i=1;i<=n;i++)
          x[i]=0.0;
        dem = 0;
        do
          {
                dem=dem+1;
                for (i=1;i<=n;i++)
                  {
                        s=0.0;
                        for (j=1;j<=n;j++)
                          s=s+a[i][j]*x[j];
                        d=x[i];
                        x[i]=(1-w)*d+w*(-s+a[i][i]*d+b[i])/a[i][i];
                        t=fabs(d-x[i]);
                  }
    }
        while ((dem<=k)&&(t>epsi*fabs(x[n])));
        if (t<epsi*fabs(x[n]))
          {
                printf("Nghiem sau %d lan lap la :\n",dem);
                for (i=1;i<=n;i++)
                  printf("x[%d] = %12.8f\n",i,x[i]);
          }
        else
          {
                printf("Khong dat do chinh xac sau %d lan lap\n",k);
                printf("Nghiem cua lan lap cuoi cung la : \n");
                for (i=1;i<=n;i++)
                  printf("x[%d] = %12.8f\n",i,x[i]);
          }
        getch();
  }
```

§7. PHƯƠNG PHÁP CRAMER

Một trường hợp riêng của hệ phương trình , trong đó số phương trình bằng số ẩn , nghĩa là hệ có dạng :

$$\begin{cases} a_{11}x_1+a_{12}x_2+a_{13}x_3+\cdots+a_{1n}x_n=b_1 \\ a_{21}x_1+a_{22}x_2+a_{23}x_3+\cdots+a_{2n}x_n=b_2 \\ \cdots\cdots \\ a_{n1}x_1+a_{n2}x_2+a_{n3}x_3+\cdots+a_{nn}x_n=b_n \end{cases}$$

trong đó A= (a_{ij}) là một ma trận không suy biến . một hệ phương trình như vậy có tên là *hệ thống Cramer*

Định lí Crame : Hệ thống Crame có nghiệm duy nhất được cho bởi công thức :

$$x_i = \frac{|A^{(i)}|}{|A|} \qquad (i = 1,...,n)$$

trong đó $A^{(i)}$ là ma trận nhận được từ A bằng cách thay cột thứ i bởi cột $B=[\ b_1\ ,...,b_n]^T$

Như vậy để giải hệ bằng phương pháp Cramer chúng ta lần lượt tính các định thức của ma trận và ma trận thay thế rồi tìm nghiệm theo công thức Cramer. Chương trình sau mô tả thuật toán này,

Chương trình 10-.9

```
// Cramer;
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#define max  50

void main()
 {
        float r[max][max],a[max][max];
        float b[max],x[max];
        float delta[max];
        int i,j,k,l,t,n,ok1,ok2,t1;
        float c,d;
        char tl;

        clrscr();
        printf("Cho so an cua phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho cac phan tu cua ma tran a : \n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
        printf("\n");
        printf("Ma tran a ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%10.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
        t1=1;
        flushall();
        while (t1)
          {
```

```
                printf("Co sua ma tran a khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("Cho chi so cot can sua : ");
                        scanf("%d",&j);
                        printf("a[",i,",",j,"] = ");
                        scanf("%f",&a[i][j]);
                        flushall();
                  }
                if (toupper(tl)=='K')
                  t1=0;
          }
        printf("Ma tran a \n");
        printf("\n");
        for (i=1;i<=n;i++)
          {
                for (j=1;j<=n;j++)
                  printf("%10.5f",a[i][j]);
                printf("\n");
          }
        printf("\n");
printf("Cho cac phan tu cua ma tran b : \n");
for (i=1;i<=n;i++)
         {
                printf("b[%d] = ",i);
                scanf("%f",&b[i]);
          }
        printf("\n");
        printf("Ma tran b ma ban da nhap\n");
        printf("\n");
        for (i=1;i<=n;i++)
          printf("b[%d] = %10.5f\n",i,b[i]);
        printf("\n");
        t1=1;
        flushall();
        while (t1)
          {
                printf("Co sua ma tran b khong(c/k)?");
                scanf("%c",&tl);
                if (toupper(tl)=='C')
                  {
                        printf("Cho chi so hang can sua : ");
                        scanf("%d",&i);
                        printf("b[%d] = ",i);
                        scanf("%f",&b[i]);
                        flushall();
                  }
```

```
                if (toupper(tl)=='K')
                  t1=0;
            }
          printf("\n");
          printf("Ma tran b\n");
          printf("\n");
          for (i=1;i<=n;i++)
            printf("%10.5f",b[i]);
          printf("\n");

//thay b vao tung cot cua a de tinh cac dinh thuc
          for (k=0;k<=n;k++)
            {
                for (i=1;i<=n;i++)
                  for (j=1;j<=n;j++)//thay cot b vao a
                        if (i==k)
                          r[j][i]=b[j];
                        else
                          r[j][i]=a[j][i];
//tinh dinh thuc
          d=1.0;
          i=1;
          ok2=1;
          while (ok2&&(i<=n))
                {
                 if (r[i][i]==0.0)
                        {
                         ok1=1;
                         t=t+1;
                         while (ok1&&(t<=n))
                                if (r[t][i]!=0)
                                  {
                                        for (j=i;j<=n;j++)
                                          {
                                                c=r[i][j];
                                                r[i][j]=r[t][j];
                                                r[k][j]=c;
                                          }
                                        d=-d;
                                        ok1=0;
                                  }
                                else
                                  t=t+1;
                                if (k>n)
                                  {
                                        printf("\n");
                                        printf("** MA TRAN SUY BIEN **");
                                        ok2=0;
                                        d=0.0;
                                  }
```

```
                    }
                if (r[i][i]!=0)
                  {
                        c=r[i][i];
                        for (j=i+1;j<=n;j++)
                          r[i][j]=r[i][j]/c;
                        for (t=i+1;t<=n;t++)
                          {
                                c=r[t][i];
                                for (j=i+1;j<=n;j++)
                                  r[t][j]=r[t][j]-r[i][j]*c;
                          }
                  }
                i=i+1;
              }
            if (ok2)
              for (i=1;i<=n;i++)
                    d=d*r[i][i];
            delta[k]=d;
          }
        if (delta[0]==0.0)
          printf("He da cho vo nghiem\n");
        else
          {
            printf("\nNGHIEM CUA HE :\n");
            printf("\n");
            for (i=1;i<=n;i++)
              {
                x[i]=delta[i]/delta[0];
                printf("x[%d] = %10.5f\n",i,x[i]);
              }
          }
        getch();
}
```

§8. HỆ PHƯƠNG TRÌNH SỐ PHỨC

Giả sử ta có một hệ phương trình dạng số phức dạng AX = B trong đó A = C + jD , B = E +jF và X = Y + jZ . Ta viết lại phương trình dưới dạng :

(C + jD)(Y + jZ) = (E +jF)

Nhân biểu thức về trái và cân bằng phần thực với phần thực và phần ảo với phần ảo ta nhận được hệ mới :

$$\begin{cases} CY - DZ = E \\ DY + CZ = F \end{cases}$$

Như vậy chúng ta nhận được một hệ gồm 2n phương trình số thực . Giải hệ này và kết hợp các phần thực à phần ảo ta nhận được nghiệm của hệ phương trình ban đầu . Chương trình giải hệ phương trình như vậy cho ở dưới đây :

Chương trình 10-10

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <ctype.h>
#define max  20

void main()
 {
        int i,j,k,l,n,m;
        float s,t,a[max][max],b[max][max],x[max];


        clrscr();
        printf("Cho so an so cua phuong trinh n = ");
        scanf("%d",&n);
        printf("Cho phan thuc cua cac he so,ke ca ve phai\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n+1;j++)
                {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
                }
   printf("\n");
        printf("Cho phan ao cua cac he so,ke ca ve phai\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n+1;j++)
                {
                  printf("b[%d][%d] = ",i,j);
                  scanf("%f",&b[i][j]);
                }
        for (i=1;i<=n;i++)
          a[i][2*n+1]=a[i][n+1];
        for (i=n+1;i<=2*n;i++)
          a[i][2*n+1]=b[i-n][n+1];
        for (i=n+1;i<=2*n;i++)
          for (j=1;j<=n;j++)
                a[i][j]=b[i-n][j];
        for (i=1;i<=n;i++)
          for (j=n+1;j<=2*n;j++)
                a[i][j]=-b[i][j-n];
        for (i=n+1;i<=2*n;i++)
          for (j=n+1;j<=2*n;j++)
                a[i][j]=a[i-n][j-n];
        m=2*n;
        for (k=1;k<=m-1;k++)
          {
                s=0.0;
```

```
            for (i=k;i<=m;i++)
              {
                    t=fabs(a[i][k]);
                    if (s<=t)
                      {
                            s=t;
                            l=i;
                      }
              }
            for (j=k;j<=m+1;j++)
              {
                    s=a[k][j];
                    a[k][j]=a[l][j];
                    a[l][j]=s;
              }
            if (fabs(a[k][k]/a[1][1])<=1e-08)
              {
                    printf("Ma tran suy bien\n");
                    getch();
                    exit(1);
              }
            for (i=k+1;i<=m;i++)
              {
                    s=a[i][k]/a[k][k];
                    a[i][k]=0.0;
                    for (j=k+1;j<=m+1;j++)
                      a[i][j]-=s*a[k][j];
              }
          }
        x[m]=a[m][m+1]/a[m][m];
        for (i=m-1;i>=1;i--)
          {
            s=a[i][m+1];
            for (j=i+1;j<=m;j++)
              s-=a[i][j]*x[j];
            x[i]=s/a[i][i];
          }
        printf("\n");
        printf("Nghiem phuc cua he\n");
        for (i=1;i<=n;i++)
          if (x[i+n]<0)
            printf("%10.5f-%10.5fj\n",x[i],fabs(x[i+n]));
          else
            printf("%10.5f+%10.5fj\n",x[i],x[i+n]);
        getch();
}
```

Dùng chương trình này giải hệ phương trình :

$$\begin{cases} (3+7j)x + (-2+4j)y + (1-3j)z + (4+2j)r = 8+36j \\ (5-6j)x + (2+5j)z + (-3+j)r = 4+10j \\ (4+5j)x + (1+2j)y + (-5-j)z + 6r = 13-3j \\ (2+4j)x + (1-j)y + (2-3j)r = -10+6j \end{cases}$$

Ta nhận được các nghiệm x = 2 + 3j ; y = 1 - 2j ; z = -1 + 4j và r = 1- j

Ngoài các phương pháp nêu trên ta thấy rằng từ hệ phương trình AX = B ta có thể tìm nghiệm X của hệ bằng cách viết lại phương trình dưới dạng X = B/A =A^{-1}B với A^{-1} là ma trận nghịch đảo của A . Do vậy trước hết ta cần tìm A^{-1} và sau đó tính tích A^{-1}B.

CHƯƠNG 11 : NỘI SUY VÀ XẤP XỈ HÀM

§1.NỘI SUY LAGRANGE

Trong thực tế nhiều khi phải phục hồi một hàm y = f(x) tại mọi giá trị x trong một đoạn [a,b] nào đó mà chỉ biết một số nhất định các giá trị của hàm tại một số điểm cho trước.Các giá trị này được cung cấp qua thực nghiệm hay tính toán.Vì vậy nảy sinh vấn đề toán học là trên đoạn $a \le x \le b$ cho một loạt các điểm x_i (i= 0,1,2..) và tại các điểm x_i này giá trị của hàm là $y_i = f(x_i)$ đã biết.Bây giờ ta cần tìm đa thức :

$$P_n(x) = a_o x^n + a_1 x^{n-1} + \ldots + a_{n-1} x + a_n$$

sao cho $P_n(x_i) = f(x_i) = y_i$.Đa thức $P_n(x)$ được gọi là đa thức nội suy của hàm y = f(x).Ta chọn đa thức để nội suy hàm y = f(x) vì đa thức là loại hàm đơn giản,luôn có đạo hàm và nguyên hàm.Việc tính giá trị của nó theo thuật toán Horner cũng đơn giản.

Bây giờ ta xây dựng đa thức nội suy kiểu Lagrange.Gọi L_i là đa thức :

$$L_i = \frac{(x - x_0)...(x - x_{i-1})(x - x_{i+1})...(x - x_n)}{(x_i - x_0)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_n)}$$

Rõ ràng là $L_i(x)$ là một đa thức bậc n và :

$$L_i(x_j) = \begin{cases} 1 & j=i \\ 0 & j \neq i \end{cases}$$

Ta gọi đa thức này là đa thức Lagrange cơ bản.

Bây giờ ta xét biểu thức :

$$P_n(x) = \sum_{i=0}^{n} f(x_i) L_i(x)$$

Ta thấy $P_n(x)$ là một đa thức bậc n vì các $L_i(x)$ là các đa thức bậc n và thoả mãn điều kiện $P_n(x_i) = f(x_i) = y_i$.Ta gọi nó là đa thức nội suy Lagrange.

Với n = 1 ta có bảng

x	x_0	x_1
y	y_0	y_1

Đa thức nội suy sẽ là :

$$P_1(x) = y_o L_0(x) + y_1 L_1(x_1)$$

$$L_0 = \frac{x - x_1}{x_0 - x_1} \qquad L_1 = \frac{x - x_0}{x_1 - x_0}$$

nên $$P_1(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0}$$

Như vậy $P_1(x)$ là một đa thức bậc nhất đối với x

Với n = 2 ta có bảng

x	x_0	x_1	x_2
y	y_0	y_1	y_2

Đa thức nội suy sẽ là :

$$P_2(x) = y_o L_0(x) + y_1 L_1(x_1) + y_2 L_2(x_2)$$

$$L_0 = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)}$$

$$L_1 = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)}$$

$$L_2 = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

Như vậy $P_1(x)$ là một đa thức bậc hai đối với x

Trên cơ sở thuật toán trên ta có chương trình tìm đa thức nội suy của một hàm khi cho trước các điểm và sau đó tính trị số của nó tại một giá trị nào đó như sau :

Chương trình 11-1

```
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#define max   21

int maxkq,n;
float x[max],y[max],a[max],xx[max],yy[max];
float x0,p0;

void main()
 {
        int i,k;
        char ok ;

        void vaosolieu(void);
        float lagrange(int,float [],float [],float);
        void inkq(void);

        clrscr();
        printf("%24cNOI SUY DA THUC LAGRANGE\n",' ');
        vaosolieu();
        k=0;
        ok='c';
        while (ok=='c')
          {
                printf("Tinh gia tri cua y voi x la x0 = ");
                scanf("%f",&x0);
                p0=lagrange(n,x,y,x0);
                printf("Gia tri cua y = %15.5f\n",p0);
                printf("\n");
                k=k+1;
                maxkq=k;
                xx[k]=x0;
                yy[k]=p0;
                flushall();
                printf("Tinh tiep khong(c/k)?");
                scanf("%c",&ok);
          }
        inkq();
```

```
 }

void vaosolieu()
 {
        int i,t;
        char ok;

        printf("\n");
        printf("Ham y = f(x)\n");
        printf("So cap (x,y) nhieu nhat la max = 20\n");
        printf("So diem da cho truoc n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
          }
        printf("\n");
        printf("     SO LIEU BAN VUA NHAP\n");
        printf("         x               y\n");
        for (i=1;i<=n;i++)
          printf("%8.4f          %8.4f\n",x[i],y[i]);
        ok=' ';
        t=1;
        flushall();
        while (t)
          {
                printf("\nCo sua so lieu khong(c/k):?");
                scanf("%c",&ok);
                if (toupper(ok)=='C')
                  {
                        printf("Chi so cua phan tu can sua i = ");
                        scanf("%d",&i);
                        printf("Gia tri moi : ");
                        printf("x[%d] = ",i);
                        scanf("%f",&x[i]);
                        printf("y[%d] = ",i);
                        scanf("%f",&y[i]);
                        flushall();
                  }
                if (toupper(ok)!='C')
                  t=0;
          }
 }

float lagrange(int n,float x[max],float y[max],float x0)
 {
        int i,k;
```

```
        float g0;

        p0=0.0;
        for (k=1;k<=n;k++)
          {
                g0=1.0;
                for (i=1;i<=n;i++)
                  if (i!=k)
                        g0=g0*(x0-x[i])/(x[k]-x[i]);
                  p0=p0+y[k]*g0;
          }
        return(p0);
  }

void inkq()
  {
        int i,j,k;
        printf("\n");
        printf("%24cBANG SO LIEU\n",' ');
        printf("%18cx %24cy\n",' ',' ');
        for (i=1;i<=n;i++)
          printf("%20.4f %25.4f\n",x[i],y[i]);
        printf("\n");
        printf("%24cKET QUA TINH TOAN\n",' ');
        printf("%14cx %10cy\n",' ',' ');
        for (k=1;k<=maxkq;k++)
          printf("%15.5f %15.5f\n",xx[k],yy[k]);
        getch();
  }
```

Giả sử ta có bảng các giá trị x,y :

x	0	3	-2	2	4
y	0	-3.75	10	-2	4

vậy theo chương trình tại x = 2.5 y = -3.3549.

§2.NỘI SUY NEWTON

Bây giờ ta xét một cách khác để xây dựng đa thức nội suy gọi là phương pháp Newton.Trước hết ts đưa vào một khái niệm mới là tỉ hiệu

Giả sử hàm y = y(x) có giá trị cho trong bảng sau :

x	x_0	x_1	x_2	…	x_{n-1}	x_n
y	y_0	y_1	y_2	…	y_{n-1}	y_n

Tỉ hiệu cấp 1 của y tại x_i,x_j là :

$$y[x_i, x_j] = \frac{y_i - y_j}{x_i - x_j}$$

Tỉ hiệu cấp hai của y tại x_i,x_j,x_k là :

$$y[x_i, x_j, x_k] = \frac{y[x_i, x_j] - y[x_j, x_k]}{x_i - x_k}$$

v.v.

Với $y(x) = P_n(x)$ là một đa thức bậc n thì tỉ hiệu cấp 1 tại x,x_0 :

$$P_n[x, x_0] = \frac{P_n(x) - P_n(x_0)}{x - x_0}$$

là một đa thức bậc (n-1).Tỉ hiệu cấp 2 tại x,x_0,x_1 :

$$P_n[x, x_0, x_1] = \frac{P_n[x, x_0] - P_n[x_0, x_1]}{x - x_1}$$

là một đa thức bậc (n-2) v.v và tới tỉ hiệu cấp (n+1) thì :

$$P_n[x, x_o, .., x_n] = 0$$

Từ các định nghĩa tỉ hiệu ta suy ra :

$$P_n(x) = P_n(x_0) + (x - x_0)P_n[x, x_o]$$
$$P_n[x, x_0] = P_n[x_0, x_1] + (x - x_1) P_n[x, x_o, x_1]$$
$$P_n[x, x_o, x_1] = P_n[x_0, x_1, x_2] + (x - x_2) P_n[x, x_o, x_1, x_2]$$
.....
$$P_n[x, x_o, .., x_{n-1}] = P_n[x_0, x_1, .., x_n] + (x - x_n) P_n[x, x_o, .., x_n]$$

Do $P_n[x, x_o, .., x_n] = 0$ nên từ đó ta có :

$$P_n(x) = P_n(x_0) + (x - x_0)P_n[x_o, x_1] + (x - x_0)(x - x_1)P_n[x_0, x_1, x_2] + \ldots$$
$$+(x - x_0)\ldots(x - x_{n-1})P_n[x_0, \ldots, x_n]$$

Nếu $P_n(x)$ là đa thức nội suy của hàm y=f(x) thì :

$P_n(x_i) = f(x_i) = y_i$ với $i = 0 \div n$

Do đó các tỉ hiệu từ cấp 1 đến cấp n của P_n và của y là trùng nhau và như vậy ta có :

$$P_n(x) = y_0 + (x - x_0)y[x_0, x_1] + (x - x_0)(x - x_1)y[x_0, x_1, x_2] + .. +$$
$$(x - x_0)(x - x_1)...(x - x_{n-1})y[x_0, .., x_n]$$

Đa thức này gọi là đa thức nội suy Newton tiến xuất phát từ nút x_0 của hàm y = f(x).Ngoài đa thức tiến còn có đa thức nội suy Newton lùi xuất phát từ điểm x_n có dạng như sau :

$$P_n(x) = y_n + (x - x_n)y[x_n, x_{n-1}] + (x - x_n)(x - x_{n-1})y[x_n, x_{n-1}, x_{n-2}] + .. +$$
$$(x - x_n)(x - x_{n-1})...(x - x_1)y[x_n, .., x_0]$$

Trường hợp các nút cách đều thì $x_i = x_0 + ih$ với i = 0,1,..,n.Ta gọi sai phân tiến cấp 1 tại i là :

$$\Delta y_i = y_{i+1} - y_i$$

và sai phân tiến cấp hai tại i :

$$\Delta^2 y_i = \Delta(\Delta y_i) = y_{i+2} - 2y_{i+1} + y_i$$
.....

và sai phân tiến cấp n là :

$$\Delta^n y_i = \Delta(\Delta^{n-1} y_i)$$

Khi đó ta có :

$$y[x_0, x_1] = \frac{\Delta y_0}{h}$$

$$y[x_0, x_1, x_2] = \frac{\Delta^2 y_0}{2h^2}$$

.....

$$y[x_0,\ldots,x_n] = \frac{\Delta^n y_0}{(n!h^n)}$$

Bây giờ đặt $x = x_0 + ht$ trong đa thức Newton tiến ta được :

$$P_n(x_0+ht) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \ldots + \frac{t(t-1)\ldots(t-n+1)}{n!}\Delta^n y_0$$

thì ta nhận được đa thức Newton tiến xuất phát từ x_0 trong trường hợp nút cách đều.Với n =1 ta có :

$$P_1(x_0+ht) = y_0 + \Delta y_0$$

Với n =2 ta có :

$$P_2(x_0+ht) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2}\Delta^2 y_0$$

Một cách tương tự ta có khái niệm các sai phân lùi tại i :

$\nabla y_i = y_i - y_{i-1}$

$\nabla^2 y_i = \nabla(\nabla y_i) = y_i - 2y_{i-1} + y_{i-2}$

.....

$\nabla^n y_i = \nabla(\nabla^{n-1} y_i)$

và đa thức nội suy Newton lùi khi các điểm nội suy cách đều :

$$P_n(x_0+ht) = y_n + t\nabla y_n + \frac{t(t+1)}{2!}\nabla^2 y_n + \ldots + \frac{t(t+1)\ldots(t+n-1)}{n!}\nabla^n y_n$$

Ví dụ : Cho hàm như bảng sau :

x	0.1	0.2	0.3	0.4
y	0.09983	0.19867	0.29552	0.38942

Ta tính giá trị của hàm tại 0.14 bằng đa thức nội suy Newton vì các mốc cách đều h = 0.1.Ta có bảng sai phân sau :

i	x	y	Δy	$\Delta^2 y$	$\Delta^3 y$
0	0.1	0.09983			
			0.09884		
1	0.2	0.19867		-	
				0.00199	
			0.09685		-0.00096
2	0.3	0.29552		-	
				0.00295	
			0.09390		
3	0.4	0.38942			

Ta dùng công thức Newton tiến với điểm gốc là $x_0 = 0.1$.h = 0.1.Với x = 0.14 ta có 0.14 = 0.1 + 0.1t nên t = 0.4 và kết quả là :

$$P(0.1+0.1t) = 0.09983 + t.0.099884 + \frac{t(t-1)}{2!}0.00199 - \frac{t(t-1)(t-2)}{3!}0.00096 = 0.13954336$$Chương trình nội suy Newton như sau :

Chương trình 11-2

```
//Noi suy Newton
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#define max   11

void main()
  {
        int i,j,k,n,t;
        float a[max],b[max],x[max],y[max];
        char ok;
        float x0,p;

        clrscr();
        printf("So diem da cho n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
          }
        printf("%10cBANG SO LIEU\n",' ');
        printf("%8cx%30cy\n",' ',' ');
        for (i=1;i<=n;i++)
          printf("%4c%8.4f%23c%8.4f\n",' ',x[i],' ',y[i]);
        ok=' ';
        t=0;
        flushall();
        while (t)
          {
                printf("Co sua so lieu khong(c/k): ");
                scanf("%c",&ok);
                if (toupper(ok)=='C')
                  {
                        printf("Chi so cua phan tu can sua i = ");
                        scanf("%d",&i);
                        printf("Gia tri moi : ");
                        printf("x[%d] = ",i);
                        scanf("%f",&x[i]);
                        printf("y[%d] = ",i);
                        scanf("%f",&y[i]);
                        flushall();
                  }
                if (toupper(ok)!='C')
                  t=0;
          }
        a[1]=y[1];
        for (j=1;j<=n-1;j++)
```

```
        {
                for (i=1;i<=n-j;i++)
                  y[i]=(y[i+1]-y[i])/(x[i+j]-x[i]);
                a[j+1]=y[1];
        }
      b[n]=a[n];
      for (k=n-1;k>=1;k--)
        {
                for (j=n-1;j>=1;j--)
                  b[j]=a[j] ;
                for (i=n-1;i>=k;i--)
                  a[i]=a[i]-b[i+1]*x[k];
        }
      for (i=n;i>=1;i--)
        printf("He so bac %d la :%8.4f\n",i-1,a[i]);
      printf("\n");
      k=0;
      ok='c';
      flushall();
      while (ok=='c')
        {
                printf("Tinh gia tri cua y tai x = ");
                scanf("%f",&x0);
                p=0;
                for (k=n;k>=1;k--)
                  p=p*x0+a[k];
                printf("Tri so noi suy tai x0 = %4.2f la : %10.5f\n",x0,p);
                getch();
                printf("Ban co muon tinh tiep cac diem khac khong(c/k)");
                do
                  scanf("%c",&ok);
                while ((ok!='c')&&(ok!='k'));
        }
}
```

Dùng chương trình này nội suy các giá trị cho trong bảng sau

0	0.2	0.4	0.6	0.8	1.0
1	1.22140276	1.4918247	1.8221188	2.22554093	2.71828183

ta có các hệ số của đa thức nội suy : 0.0139(bậc 5),0.0349(bậc 4),0.1704(bậc3),0.4991(bậc 2),1.0001(bậc 1) và 1.0000(bậc 0).

§3.NỘI SUY AITKEN

Một dạng khác của đa thức nội suy được xác định bằng thuật toán Aitken.Giả sử ta có n điểm đã cho của hàm f(x).Như vậy qua hai điểm x_0 và x_1 ta có đa thức nội suy Lagrange của hàm f(x) được viết dưới dạng :

$$P_{01}(x) = \frac{\begin{vmatrix} y_0 & x_0 - x \\ y_1 & x_1 - x \end{vmatrix}}{x_1 - x_0}$$

là một đa thức bậc 1 :

$$P_{01}(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0}$$

.Khi $x = x_0$ thì :

$$P_{01}(x_0) = \frac{\begin{vmatrix} y_0 & x_0 - x_0 \\ y_1 & x_1 - x_0 \end{vmatrix}}{x_1 - x_0} = y_0$$

Khi $x = x_1$ thì :

$$P_{01}(x_1) = \frac{\begin{vmatrix} y_0 & x_0 - x_1 \\ y_1 & x_1 - x_1 \end{vmatrix}}{x_1 - x_0} = y_1$$

Đa thức nội suy Lagrange của f(x) qua 3 điểm x_0,x_1,x_2 có dạng :

$$P_{012}(x) = \frac{\begin{vmatrix} P_{01}(x) & x_0 - x \\ P_{12}(x) & x_2 - x \end{vmatrix}}{x_2 - x_0}$$

và là một đa thức bậc 2:

$$P_{012}(x) = y_0 \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} + y_1 \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} + y_2 \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)}$$

Khi $x = x_0$ thì :

$$P_{012}(x_0) = \frac{\begin{vmatrix} y_0 & x_0 - x_0 \\ P_{12}(x) & x_2 - x_0 \end{vmatrix}}{x_2 - x_0} = y_0$$

Khi $x = x_1$ thì :

$$P_{012}(x_1) = \frac{\begin{vmatrix} y_1 & x_0 - x_1 \\ y_1 & x_2 - x_1 \end{vmatrix}}{x_2 - x_0} = y_1$$

Khi $x = x_2$ thì :

$$P_{012}(x_2) = \frac{\begin{vmatrix} P_{01}(x_2) & x_0 - x_2 \\ y_2 & x_2 - x_2 \end{vmatrix}}{x_2 - x_0} = y_2$$

Tổng quát đa thức nội suy Lagrange qua n điểm là :

$$P_{012..n}(x) = \frac{\begin{vmatrix} P_{01..(n-1)}(x) & x_0 - x \\ P_{12..n}(x) & x_n - x \end{vmatrix}}{x_2 - x_0}$$

Như vậy ta có thể dùng phép lặp để xác định lần lượt các đa thức Lagrange.Sơ đồ tính toán như vậy gọi là sơ đồ Neville-Aitken.

Ví dụ : Cho các cặp điểm (0,0.4),(1.4,1.5),(2.6,1.8),(3.9,2.6),tính y tại x=2

$$P_{01}(x)=\frac{\begin{vmatrix} y_0 & x_0-x \\ y_1 & x_1-x \end{vmatrix}}{x_1-x_0}=\frac{\begin{vmatrix} 0.4 & -2 \\ 1.5 & -0.6 \end{vmatrix}}{1.4-0}=1.97143$$

$$P_{12}(x)=\frac{\begin{vmatrix} y_1 & x_1-x \\ y_2 & x_2-x \end{vmatrix}}{x_2-x_1}=\frac{\begin{vmatrix} 1.5 & -0.6 \\ 1.8 & 0.6 \end{vmatrix}}{2.6-1.4}=1.65$$

$$P_{012}(x)=\frac{\begin{vmatrix} P_{01}(x) & x_0-x \\ P_{12}(x) & x_2-x \end{vmatrix}}{x_2-x_0}=\frac{\begin{vmatrix} 1.97143 & -2 \\ 1.65 & 0.6 \end{vmatrix}}{2.6-0}=1.7242$$

$$P_{23}(x)=\frac{\begin{vmatrix} y_2 & x_2-x \\ y_3 & x_3-x \end{vmatrix}}{x_3-x_2}=\frac{\begin{vmatrix} 1.8 & 0.6 \\ 2.6 & 1.9 \end{vmatrix}}{3.9-2.6}=1.4308$$

$$P_{123}(x)=\frac{\begin{vmatrix} P_{12}(x) & x_1-x \\ P_{23}(x) & x_3-x \end{vmatrix}}{x_3-x_1}=\frac{\begin{vmatrix} 1.65 & -0.6 \\ 1.4308 & 1.9 \end{vmatrix}}{3.9-1.4}=1.5974$$

$$P_{0123}(x)=\frac{\begin{vmatrix} P_{012}(x) & x_0-x \\ P_{123}(x) & x_3-x \end{vmatrix}}{x_3-x_0}=\frac{\begin{vmatrix} 1.7242 & -2 \\ 1.5974 & 1.9 \end{vmatrix}}{3.9-0}=1.6592$$

Chương trình được viết như sau

Chương trình 11-3

```
//Noi suy Aitken
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#define max   11

void main()
 {
        float x[max],y[max],yd[max];
        float x1;
        int j,k,n,n1;

        clrscr();
        printf("Cho so diem da co n = ");
        scanf("%d",&n1);
        n=n1-1 ;
        for (k=0;k<=n;k++)
          {
                printf("x[%d] = ",k+1);
                scanf("%f",&x[k]);
                printf("y[%d] = ",k+1);
                scanf("%f",&y[k]);
          }
        printf("Cho diem can tinh gia tri cua ham x1 = ");
```

```
        scanf("%f",&x1);
        for (k=0;k<=n-1;k++)
          {
                yd[k]=(y[k]*(x1-x[k+1])-y[k+1]*(x1-x[k]))/(x[k]-x[k+1]);
                if (k!=0)
                for (j=k-1;j>=0;j--)
                  yd[j]=(yd[j]*(x1-x[k+1])-yd[j+1]*(x1-x[j]))/(x[j]-x[k+1]);
          }
        printf("Gia tri ham tai x = %6.3f la y = %8.4f\n",x1,yd[0]);
        getch();
}
```

Dùng chương trình này để nội suy các cặp số (1,3),(2,5),(3,7),(4,9) và (5,11) tại x = 2.5 ta có y = 6.

§4.XẤP XỈ HÀM BẰNG PHƯƠNG PHÁP BÌNH PHƯƠNG BÉ NHẤT

Trong các mục trước ta đã nội suy giá trị của hàm.Bài toán đó là cho một hàm dưới dạng bảng số và phải tìm giá trị của hàm tại một giá trị của đối số không nằm trong bảng.

Trong thực tế,bên cạnh bài toán nội suy ta còn gặp một dạng bài toán khác.Đó là tìm công thức thực nghiệm của một hàm.Nội dung bài toán là từ một loạt các điểm cho trước (có thể là các giá trị của một phép đo nào đó) ta phải tìm một hàm xấp xỉ các giá trị đã cho.Ta sẽ dùng phương pháp bình phương tối thiểu để giải bài toán.Giả sử có mẫu quan sát (x_i,y_i) của hàm y = f(x).Ta chọn hàm f(x) có dạng :

$$f(x) = a_0f_0(x) + a_1f_1(x) + a_2f_2(x)... \qquad (1)$$

Trong đó các hàm $f_0(x),f_1(x),f_2(x)$ v.v.là (m+1) hàm độc lập tuyến tính mà ta có thể chọn tuỳ ý và các hệ số a_i là tham số chưa biết mà ta phải xác định dựa vào hệ hàm đã chọn và các điểm quan sát.Sai số giữa trị đo được và trị tính theo (1) là :

$$e_i = y_i - f(x_i) \qquad (2)$$

Sai số này có thể âm hay dương tuỳ từng giá trị của y_i.Khi dùng phương pháp bình phương bé nhất ta xét bình phương của sai số tại một điểm :

$$e_i^2 = \left[y_i - f(x_i)\right]^2 \qquad (3)$$

Với n điểm tổng bình phương của sai số sẽ là :

$$S = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left\{y_i - \left[a_0f_0(x_i) + a_1f_1(x_i) + \cdots + a_nf_n(x_i)\right]\right\}^2$$

Rõ ràng S là hàm của các giá trị cần tìm a_i.và chúng ta sẽ chọn các a_i sao cho S đạt giá trị min,nghĩa là các đạo hàm $\frac{\partial S}{\partial a_i}$ phải bằng không.Ta sẽ xét các trường hợp cụ thể.

1.Hàm xấp xỉ có dạng đa thức : Trong trường hợp tổng quát ta chọn hệ hàm xấp xỉ là một đa thức,nghĩa là :

$$f(x) = a_0 + a_1x + a_2x^2 +...+ a_mx^m$$

Vậy hàm S là :

$$S = \left(y_i - a_0 + a_1x + a_2x + \cdots + a_mx\right)^2$$

Theo điều kiện đạo hàm $\frac{\partial S}{\partial a_i} = 0$ ta nhận được hệ phương trình:

$$\begin{cases} a_m\sum_{i=1}^{n}x_i^m + a_{m-1}\sum_{i=1}^{n}x_i^{m-1} + \cdots + na_0 = \sum_{i=1}^{n}y_i \\ a_m\sum_{i=1}^{n}x_i^{m+1} + a_{m-1}\sum_{i=1}^{n}x_i^m + \cdots + a_0\sum_{i=1}^{n}x_i = \sum_{i=1}^{n}x_iy_i \\ a_m\sum_{i=1}^{n}x_i^{m+2} + a_{m-1}\sum_{i=1}^{n}x_i^{m+1} + \cdots + a_0\sum_{i=1}^{n}x_i^2 = \sum_{i=1}^{n}x_i^2y_i \\ a_m\sum_{i=1}^{n}x_i^{m+3} + a_{m-1}\sum_{i=1}^{n}x_i^{m+2} + \cdots + a_0\sum_{i=1}^{n}x_i^3 = \sum_{i=1}^{n}x_i^3y_i \\ \cdots \\ a_m\sum_{i=1}^{n}x_i^{2m} + a_{m-1}\sum_{i=1}^{n}x_i^{2m-1} + \cdots + a_0\sum_{i=1}^{n}x_i^m = \sum_{i=1}^{n}x_i^my_i \end{cases}$$

Đây là một hệ phương trình tuyến tính.Giải nó ta nhận được các gía trị a_i.Sau đây là chương trình viết theo thuật toán trên.

Chương trình 11-4

```
//Xap xi da thuc
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#define max   11

void main()
 {
        int i,j,k,m,n,p,kp,t;
        float a[max],x[max],y[max],y1[max];
        float b[max][max];
        char ok;
        float s,sx,s1,c,d;

        clrscr();
        printf("PHUONG PHAP BINH PHUONG TOI THIEU");
        printf("\n");
        printf("Cho bac cua da thuc xap xi m = ");
        scanf("%d",&m);
        printf("So diem da cho n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
          }
        x[0]=1;
        printf("\n");
        printf("%4cBANG SO LIEU\n",' ');
```

```
printf("%8cx%30cy\n",' ',' ');
for (i=1;i<=n;i++)
  printf("%4c%8.4f%20c%8.4f\n",' ',x[i],' ',y[i]);
ok=' ';
t=1;
flushall();
while (t)
  {
        printf("Co sua so lieu khong(c/k): ");
        scanf("%c",&ok);
        if (toupper(ok)=='C')
          {
                printf("Chi so cua phan tu can sua i = ");
                scanf("%d",&i);
                printf("Gia tri moi : ");
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
                flushall();
          }
        if (toupper(ok)!='C')
          t=0;
  }
//for (i=0;i<=n;i++)
  //a[i]=0.0;
printf("\n");
printf("CAC GIA TRI DA CHO");
printf("\n");
printf("X = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ',x[i]);
printf("\n");
printf("Y = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ',y[i]);
printf("\n");
for (p=0;p<=m;p++)
  {
        y1[p]=0.0;
        for (i=1;i<=n;i++)
          {
                sx=1.0;
                for (j=1;j<=p;j++)
                  sx*=x[i];
                y1[p]+=y[i]*sx;
          }
  }
for (p=0;p<=m;p++)
  for (k=0;k<=m;k++)
```

```
                {
                 kp=k+p;
                 b[p][k]=0.0;
                 for (i=1;i<=n;i++)
                        {
                         sx=1.0;
                         for (j=1;j<=kp;j++)
                                sx*=x[i];
                         b[p][k]+=sx;
                        }
                }
        for (i=0;i<=m-1;i++)
          {
                c=1.0/b[i][i];
                for (k=i+1;k<=m;k++)
                 {
                        d=b[i][k];
                        for (j=i+1;j<=m;j++)
                         b[k][j]-=b[i][j]*c*d;
                        y1[k]-=y1[i]*c*d;
                        b[i][k]*=c;
                 }
                y1[i]*=c;
          }
        y1[m]/=b[m][m];
        for (i=m-1;i>=0;i--)
          for (j=i+1;j<=m;j++)
                y1[i]-=b[i][j]*y1[j];
        printf("\n");
        printf("CAC HE SO CUA DA THUC CAN TIM");
        printf("\n");
        for (i=0;i<=m;i++)
          printf("a[%d] = %10.5f\n",i,y1[i]);
        getch();
}
```

Với các giá trị x,y đo được theo bảng

x	7	8	9	10	11	12	13
y	7,4	8,4	9,1	9,4	9,5	9,5	9,4

ta có n = 7 và chọn m = 2 và tính được theo chương trình các hệ số :

$a_0 = -0.111905$; $a_1 = 2.545238$; $a_2 = -4.857143$

và hàm xấp xỉ sẽ là : $f(x) = -0.111905 + 2.545238x - 4.857143x^2$

2.Hàm dạng Ae^{cx} : Khi các số liệu thể hiện một sự biến đổi đơn điệu ta dùng hàm xấp xỉ là $y = Ae^{cx}$.Lấy logarit hai vế ta có :

$\ln y = \ln A + cx\ln e$

Theo điều kiện đạo hàm $\frac{\partial S}{\partial a_i} = 0$ ta có hệ phương trình :

$$\begin{cases} c\sum_{i=1}^{n} x_i + n \ln A = \sum_{i=1}^{n} \ln y_i \\ c\sum_{i=1}^{n} x_i^2 + \ln A \sum_{i=1}^{n} x_i = \sum_{i=1}^{n} x_i \ln y_i \end{cases}$$

Giải hệ phương trình này ta có các hệ số A và c :

Chương trình 11-5

```
//xap_xi_e_mu;
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <math.h>
#define max   11
void main()
 {
        int i,n,t;
        float x[max],y[max];
        char ok;
        float a,b,c,d,e,f,d1,d2,d3;

        clrscr();
        printf("PHUONG PHAP BINH PHUONG TOI THIEU");
        printf("\n");
        printf("So diem da cho n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
          }
        x[0]=1.0;
        printf("%4cBANG SO LIEU\n",' ');
        printf("%8cx%30cy\n",' ',' ');
        for (i=1;i<=n;i++)
          printf("%4c%8.4f%23c%8.4f\n",' ',x[i],' ',y[i]);
        ok=' ';
        t=1;
        while (t)
          {
                printf("Co sua so lieu khong(c/k): ");
                scanf("%c",&ok);
                if (toupper(ok)=='C')
                  {
```

```
                printf("Chi so cua phan tu can sua i = ");
                scanf("%d",&i);
                printf("Gia tri moi : ");
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
              }
            if (toupper(ok)!='C')
              t=0;
          }
        printf("CAC GIA TRI DA CHO");
        printf("\n");
        printf("X = ");
        for (i=1;i<=n;i++)
          printf("%c%8.3f",' ',x[i]);
        printf("\n");
        printf("Y = ");
        for (i=1;i<=n;i++)
          printf("%c%8.3f",' ',y[i]);
        printf("\n");
        a=0.0;
        for (i=1;i<=n;i++)
          a+=x[i];
        b=n;
        c=0.0;
        for (i=1;i<=n;i++)
          c+=log(y[i]);
        d=0.0;
        for (i=1;i<=n;i++)
          d+=x[i]*x[i];
        e=0.0;
        for (i=1;i<=n;i++)
          e+=x[i]*log(y[i]);
        d1=a*a-d*b;
        d2=c*a-e*b;
        d3=a*e-c*d;
        c=d2/d1;
        a=d3/d1;
        printf("\n");
        printf("He so A = %8.4f",exp(a));
        printf(" va so mu c = %8.4",c);
        printf("\n");
        printf("\nBANG CAC GIA TRI TINH TOAN");
printf("\n");
        printf("%5cx%28cy\n",' ',' ');
        for (i=1;i<=n;i++)
          {
                printf("%8.4f%21c%8.4f\n",x[i],' ',exp(a)*exp(c*x[i]));
          }
```

```
        getch();
 }
```

Với các giá trị x,y đo được theo bảng

x	0	2	4	6	8	10	12
y	1280	635	324	162	76	43	19

ta có n = 7 và tính được theo chương trình các hệ số : A = 1285.44 va c = -0.3476 và hàm xấp xỉ sẽ là : f(x) = 1285.44

3.Hàm dạng Ax^q : Khi các số liệu thể hiện một sự biến đổi đơn điệu ta cũng có thể dùng hàm xấp xỉ là $y = Ax^q$.Lấy logarit hai vế ta có :

$$\ln y = \ln A + q \ln x$$

Theo điều kiện đạo hàm triệt tiêu ta có hệ phương trình :

$$\begin{cases} q\sum_{i=1}^{n} \ln x_i + n \ln A = \sum_{i=1}^{n} \ln y_i \\ q\sum_{i=1}^{n} \ln^2 x_i + \ln A \sum_{i=1}^{n} \ln x_i = \sum_{i=1}^{n} \ln x_i \ln y_i \end{cases}$$

Giải hệ phương trình này ta có các hệ số A và q :

Chương trình 11-6

```
//xap_xi_x_mu;
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <math.h>
#define max   11

void main()
 {
        int i,n,t;
        float x[max],y[max];
        char ok;
        float a,b,c,d,e,f,d1,d2,d3;

        clrscr();
        printf("PHUONG PHAP BINH PHUONG TOI THIEU");
        printf("\n");
        printf("So diem da cho n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
```

```
  }
x[0]=1.0;
printf("%4cBANG SO LIEU\n",' ');
printf("%8cx%30cy\n",' ',' ');
for (i=1;i<=n;i++)
  printf("%4c%8.4f%23c%8.4f\n",' ',x[i],' ',y[i]);
ok=' ';
flushall();
t=1;
while (t)
  {
          printf("Co sua so lieu khong(c/k): ");
          scanf("%c",&ok);
          if (toupper(ok)=='C')
            {
                  printf("Chi so cua phan tu can sua i = ");
                  scanf("%d",&i);
                  printf("Gia tri moi : ");
                  printf("x[",i,"] = ");
                  scanf("%f",&x[i]);
                  printf("y[%d] = ",i);
                  scanf("%f",&y[i]);
            }
          if (toupper(ok)!='C')
            t=0;
  }
printf("\n");
printf("\nCAC GIA TRI DA CHO");
printf("\n");
printf("X = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ',x[i]);
printf("\n");
printf("Y = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ',y[i]);
printf("\n");
a=0.0;
for (i=1;i<=n;i++)
a+=log(x[i]);
b=n;
c=0.0;
for (i=1;i<=n;i++)
c+=log(y[i]);
d=0.0;
for (i=1;i<=n;i++)
  d+=log(x[i])*log(x[i]);
e=0.;
for (i=1;i<=n;i++)
  e+=log(x[i])*log(y[i]);
```

```
        d1=a*a-d*b;
        d2=c*a-e*b;
        d3=a*e-c*d;
        c=d2/d1;
        a=d3/d1;
        printf("\n");
        printf("He so A = %8.4f",exp(a));
        printf("  va so mu q = %8.4f\n",c);
        printf("\n");
        printf("\nBANG CAC GIA TRI TINH TOAN\n");
        printf("%5cx%27cy\n",' ',' ');
        for (i=1;i<=n;i++)
          {
                printf("%8.4f%20c%8.4f\n",x[i],' ',exp(a)*exp(c*log(x[i])));
          }
        getch();
}
```

Với các giá trị x,y đo được theo bảng

x	1	2	4	5	6
y	7.1	27.8	62.1	110	161

ta có n = 5 và tính được theo chương trình các hệ số : A = 7.1641 và q = 1.9531 và hàm xấp xỉ sẽ là : $f(x) = 1285.44x^{1.9531}$

4.Hàm lượng giác : Khi quan hệ y=f(x) có dạng tuần hoàn ta dùng hàm xấp xỉ là tổ hợp tuyến tính của các hàm sin và cosin dạng :

$$f(x) = a_0 + \sum_{i=1}^{n} a_i \cos(i\omega x) + \sum_{i=1}^{n} b_i \sin(i\omega x)$$

Để đơn giản trước hết ta xét hàm chỉ có một số hạng sin-cos,nghĩa là :

$$f(x) = a_0 + a_1 \cos\omega x + b_1 \sin\omega x$$

Hàm S sẽ có dạng :

$$S = \sum_{i=1}^{n} \left[y_i - (a_0 + a_1 \cos\omega x + b_1 \sin\omega x) \right]^2$$

Theo điều kiện đạo hàm triệt tiêu ta có hệ phương trình đối với các hệ số dạng :

$$\begin{bmatrix} n & \sum\cos\omega x & \sum\sin\omega x \\ \sum\cos\omega x & \sum\cos^2\omega x & \sum\cos\omega x\sin\omega x \\ \sum\sin\omega x & \sum\cos\omega x\sin\omega x & \sum\sin^2\omega x \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ b_1 \end{bmatrix} = \begin{bmatrix} \sum y \\ \sum y\cos\omega x \\ \sum y\sin\omega x \end{bmatrix}$$

Do :

$$\frac{\sum\sin\omega x}{n} = 0 \qquad \frac{\sum\cos\omega x}{n} = 0$$

$$\frac{\sum\sin^2\omega x}{n} = \frac{1}{2} \qquad \frac{\sum\cos^2\omega x}{n} = \frac{1}{2}$$

$$\frac{\sum\cos\omega x\sin\omega x}{n} = 0$$

nên hệ phương trình có dạng đơn giản :

$$\begin{bmatrix} n & 0 & 0 \\ 0 & n/2 & 0 \\ 0 & 0 & n/2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ b_1 \end{bmatrix} = \begin{bmatrix} \sum y \\ \sum y\cos\omega x \\ \sum y\sin\omega x \end{bmatrix}$$

Giải hệ ta có :

$$a_0 = \frac{\sum y}{n} \qquad a_1 = \frac{2}{n}\sum y\cos\omega x \qquad b_1 = \frac{2}{n}\sum y\sin\omega x$$

Trong trường hợp tổng quát,một cách tương tự ta có :

$$a_0 = \frac{\sum y}{n} \qquad a_i = \frac{2}{n}\sum y\cos i\omega x \qquad b_i = \frac{2}{n}\sum y\sin i\omega x$$

Chương trình tìm các hệ số a_i và b_i được thể hiện như sau :

Chương trình 11-7

```
//xap_xi_sin_cos;
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <math.h>
#define max   11
#define pi  3.15159

void main()
 {
        int i,j,m,n,t;
        float x[max],y[max],a[max],b[max];
        char ok;
        float omg,t1;

        clrscr();
        printf("PHUONG PHAP BINH PHUONG TOI THIEU");
        printf("\n");
        printf("Cho so so hang sin-cos m = ");
        scanf("%d",&m);
        printf("Cho chu ki T = ");
        scanf("%f",&t1);
        printf("So diem da cho n = ");
        scanf("%d",&n);
        for (i=1;i<=n;i++)
          {
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
          }
        x[0]=1.0;
        printf("%4cBANG SO LIEU\n",' ');
        printf("%8cx%30cy\n",' ',' ');
        for (i=1;i<=n;i++)
```

```
  printf("%4c%8.4f%23c%8.4f\n",' ',x[i],' ',y[i]);
ok=' ';
t=1;
flushall();
while (t)
  {
        printf("Co sua so lieu khong(c/k): ");
        scanf("%c",&ok);
        if (toupper(ok)=='C')
          {
                printf("Chi so cua phan tu can sua i = ");
                scanf("%d",&i);
                printf("Gia tri moi : ");
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
                flushall();
          }
        if (toupper(ok)!='C')
          t=0;
  }
printf("\nCAC GIA TRI DA CHO\n");
printf("\n");
printf("      X           Y\n");
for (i=1;i<=n;i++)
  printf("%c%8.3f%c%8.3f\n",' ',x[i],' ',y[i]);

printf("\n");
a[0]=0.0;
omg=2*pi/t1;
for (i=1;i<=n;i++)
  a[0]+=y[i];
a[0]/=n;
for (j=1;j<=m;j++)
  {
        a[j]=0.0;
        for (i=1;i<=n;i++)
          a[j]+=y[i]*cos(j*omg*x[i]);
        a[j]=2*a[j]/n;
  }
for (j=1;j<=m;j++)
  {
        b[j]=0.0;
        for (i=1;i<=n;i++)
          b[j]+=y[i]*sin(j*omg*x[i]);
        b[j]=2*b[j]/n;
  }
printf("\n");
printf("TAN SO GOC OMEGA = %10.5f\n",omg);
```

```
        printf("HE SO HANG\n");
        printf("a[0] = %8.4f\n",a[0]);
        printf("CAC HE SO BAC CAO\n");
        printf("%5ccos%25csin\n",' ',' ');
        for (i=1;i<=m;i++)
          printf("%8.4f%21c%6.4f\n",a[i],' ',b[i]);
        getch();
 }
```

Với hàm cho bằng bảng số :

x	0	0.15	0.3	0.45	0.6	0.75	0.9	1.05	1.2	1.3
y	2.200	1.595	1.031	0.722	0.786	1.200	1.805	2.369	2.678	2.614

Chọn số hệ số sin-cos m = 1,số điểm cho trước n = 10,chu kì T = 15 ta nhận được kết quả tính $a_0 = 1.7$; $a_1 = 0.5$; $b_1 = -0.8661$ và $\omega = 4.18879$.Như vậy hàm xấp xỉ có dạng :

$$f(x) = 1.7 + 0.5\cos(4.18879x) - 0.8661\sin(4.18879x)$$

5.Hàm hữu tỉ : Khi quan hệ y = f(x) có dạng đường cong bão hoà hay dạng arctan,tan v.v ta dùng hàm xấp xỉ là hàm hữu tỉ dạng đơn giản :

$$y = \frac{ax}{b + x}$$

Lấy nghịch đảo của nó ta có :

$$\frac{1}{y} = \frac{b}{a}\frac{1}{x} + \frac{1}{a}$$

Đặt 1/y = Y,1/x = X,b/a = B và 1/a = A phương trình trên sẽ có dạng :

$$Y = A + BX$$

và là một đa thức bậc một.Do vậy ta có hệ phương trình đối với các hệ số A và B là :

$$\begin{cases} nA + B\sum_{i=1}^{n}\frac{1}{x_i} = \sum_{i=1}^{n}\frac{1}{y_i} \\ A\sum_{i=1}^{n}\frac{1}{x_i} + B\sum_{i=1}^{n}\frac{1}{x_i^2} = \sum_{i=1}^{n}\frac{1}{x_i y_i} \end{cases}$$

và từ đó tính được a và b.Chương trình sau mô tả thuật toán trên

Chương trình 11-.8

```
//xap xi huu_ty;
#include <conio.h>
#include <stdio.h>
#include <ctype.h>
#include <math.h>
#define k 11
void main()
 {
        float x[k],y[k];
        float a,b,a1,b1,c,d,e;
        int i,n,t;
```

```
char ok;

clrscr();
printf("PHUONG PHAP BINH PHUONG TOI THIEU");
printf("\n");
printf("So diem da cho n = ");
scanf("%d",&n);
for (i=1;i<=n;i++)
  {
        printf("x[%d] = ",i);
        scanf("%f",&x[i]);
        printf("y[%d] = ",i);
        scanf("%f",&y[i]);
  }
printf("%4cBANG SO LIEU\n",' ');
printf("%8cx%30cy\n",' ',' ');
for (i=1;i<=n;i++)
  printf("%4c%8.4f%23c%8.4f\n",' ',x[i],' ',y[i]);
ok=' ';
t=1;
flushall();
while (t)
  {
        printf("Co sua so lieu khong(c/k): ");
        scanf("%c",&ok);
        if (toupper(ok)=='C')
          {
                printf("Chi so cua phan tu can sua i = ");
                scanf("%d",&i);
                printf("Gia tri moi : ");
                printf("x[%d] = ",i);
                scanf("%f",&x[i]);
                printf("y[%d] = ",i);
                scanf("%f",&y[i]);
                flushall();
          }
        if (toupper(ok)!='C')
          t=0;
  }
printf("CAC GIA TRI DA CHO\n");
printf("\n");
printf("X = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ',x[i]);
printf("\n");
printf("Y = ");
for (i=1;i<=n;i++)
  printf("%c%8.3f",' ' ,y[i]);
printf("\n");
a=n;
```

```
        b=0.0;
        c=0.0;
        d=0.0;
        e=0.0;
        for (i=1;i<=n;i++)
          {
                b+=1/x[i];
                c+=1/y[i];
                d+=1/(x[i]*x[i]);
                e+=1/(x[i]*y[i]);
          }
        a1=(c*d-b*e)/(a*d-b*b);
        b1=(a*e-b*c)/(a*d-b*b);
        a=1/a1;
        b=b1*a;
        printf("\n");
        printf("Cac he so cua ham huu ty\n");
        printf("a = %10.5f b = %10.5f",a,b);
        getch();
}
```

Với dãy số liệu đã cho :

x	1	2	3	4	5
y	0.3333333	0.5	0.6	0.66666	0.7142857

ta nhận được từ chương trình trị số a = 1 và b = 2

CHƯƠNG 12 : TÍNH GẦN ĐÚNG ĐẠO HÀM VÀ TÍCH PHÂN XÁC ĐỊNH

§1. ĐẠO HÀM ROMBERG

Đạo hàm theo phương pháp Romberg là một phương pháp ngoại suy để xác định đạo hàm với một độ chính xác cao . Ta xét khai triển Taylor của hàm f(x) tại (x+h) và (x-h) :

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) + \cdots \quad (1)$$

$$f(x-h) = f(x) - hf'(x) + \frac{h^2}{2}f''(x) - \frac{h^3}{3!}f'''(x) + \frac{h^4}{4!}f^{(4)}(x) - \cdots \quad (2)$$

Trừ (1) cho (2) ta có :

$$f(x+h) - f(x-h) = 2hf'(x) + \frac{2h^3}{3!}f'''(x) + \frac{2h^5}{5!}f^{(5)}(x) + \cdots \quad (3)$$

Như vậy rút ra :

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h} - \frac{h^2}{3!}f'''(x) - \frac{h^4}{5!}f^{(5)}(x) - \cdots \quad (4)$$

hay ta có thể viết lại :

$$f'(x) = \frac{1}{2h}\left[f(x+h) - f(x-h)\right] + a_2h^2 + a_4h^4 + a_6h^6 + \cdots \quad (5)$$

trong đó các hệ số a_i phụ thuộc f và x .
Ta đặt :

$$\varphi(h) = \frac{1}{2h}[f(x+h) - f(x-h)] \quad (6)$$

Như vậy từ (5) và (6) ta có :

$$D(1,1) = \varphi(h) = f'(x) - a_2h^2 - a_4h^4 - a_6h^6 - \cdots \quad (7)$$

$$D(2,1) = \varphi\left(\frac{h}{2}\right) = f'(x) - a_2\frac{h^2}{4} - a_4\frac{h^4}{16} - a_6\frac{h^6}{64} - \cdots \quad (8)$$

và tổng quát với $h_i = h/2^{i-1}$ ta có :

$$D(i,1) = \varphi(h_i) = f'(x) - a_2h_i^2 - a_4h_i^4 - a_6h_i^6 - \cdots \quad (9)$$

Ta tạo ra sai phân D(1,1) - 4D(2,1) và có :

$$\varphi(h) - 4\varphi\left(\frac{h}{2}\right) = -3f'(x) - \frac{3}{4}a_4h^4 - \frac{15}{16}a_6h^6 - \cdots \quad (10)$$

Chia hai vế của (10) cho -3 ta nhận được :

$$D(2,2) = \frac{4D(2,1) - D(1,1)}{4} = f'(x) + \frac{1}{4}a_4h^4 + \frac{5}{16}a_6h^6 + \cdots \quad (11)$$

Trong khi D(1,1) và D(2,1) sai khác f′(x) phụ thuộc vào h^2 thì D(2,2) sai khác f′(x) phụ thuộc vào h^4 . Bây giờ ta lại chia đôi bước h và nhận được :

$$D(2,2) = f'(x) + \frac{1}{4}a_4(h/2)^4 + \frac{5}{16}a_6(h/2)^6 + \ldots \quad (12)$$

và khử số hạng có h^4 bằng cách tạo ra :

$$D(2,2) - 16D(3,2) = -15f'(x) + \frac{15}{64}a_6(h)^6 + \ldots \quad (13)$$

Chia hai vế của (13) cho -15 ta có :

$$D(3,3) = \frac{16D(3,2) - D(2,2)}{15} = f'(x) - \frac{1}{64}a_6h^6 - \ldots \quad (14)$$

Với lần tính này sai số của đạo hàm chỉ còn phụ thuộc vào h^6 . Lại tiếp tục chia đôi bước h và tính D(4,4) thì sai số phụ thuộc h^8 . Sơ đồ tính đạo hàm theo phương pháp Romberg là :

D(1,1)
D(2,1) D(2,2)
D(3,1) D(3,2) D(3,3)
D(4,1) D(4,2) D(4,3) D(4,4)
. . . . . .

trong đó mỗi giá trị sau là giá trị ngoại suy của giá trị trước đó ở hàng trên .
Với $2 \le j \le i \le n$ ta có :

$$D(i,j) = \frac{4^{j-1}D(i,j-1) - D(i-1,j-1)}{4^{j-1}-1}$$

và giá trị khởi đầu là :

$$D(i,1) = \varphi(h_i) = \frac{1}{2h_i}[f(x+h_i) - f(x-h_i)]$$

với $h_i = h/2^{i-1}$.

Chúng ta ngừng lại khi hiệu giữa hai lần ngoại suy đạt độ chính xác yêu cầu.

Ví dụ : Tìm đạo hàm của hàm $f(x) = x^2 + \arctan(x)$ tại x = 2 với bước tính h = 0.5 . Trị chính xác của đạo hàm là 4.2

$$D(1,1) = \frac{1}{2\times 0.5}[f(2.5) - f(1.5)] = 4.207496266$$
$$D(2,1) = \frac{1}{2\times 0.25}[f(2.25) - f(1.75)] = 4.201843569$$
$$D(3,1) = \frac{1}{2\times 0.125}[f(2.125) - f(1.875)] = 4.200458976$$
$$D(2,2) = \frac{4^1 D(2,1) - D(1,1)}{4^1 - 1} = 4.19995935$$
$$D(3,2) = \frac{4^1 D(3,1) - D(2,1)}{4^1 - 1} = 4.200458976$$
$$D(3,3) = \frac{4^2 D(3,2) - D(2,2)}{4^{21} - 1} = 4.200492284$$

Chương trình tính đạo hàm như dưới đây . Dùng chương trình tính đạo hàm của hàm cho trong function với bước h = 0.25 tại $x_o = 0$ ta nhận được giá trị đạo hàm là 1.000000001.

Chương trình12-.1

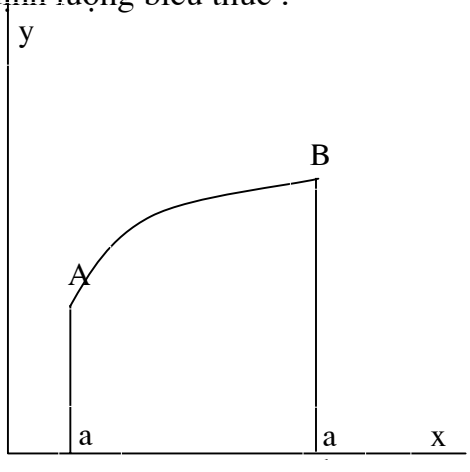
```
//Daoham_Romberg;
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define max   11
float h;
void main()
 {
        float d[max];
        int j,k,n;
        float x,p;

        float y(float),dy(float);
```

```
        clrscr();
        printf("Cho diem can tim dao ham x = ");
        scanf("%f",&x);
        printf("Tinh dao ham theo phuong phap Romberg\n");
        printf("cua ham f(x) = th(x) tai x = %4.2f\n",x);
        n=10;
        h=0.2;
        d[0]=dy(x);
        for (k=2;k<=n;k++)
          {
                h=h/2;
                d[k]=dy(x);
                p=1.0;
                for (j=k-1;j>=1;j--)
                  {
                        p=4*p;
                        d[j]=(p*d[j+1]-d[j])/(p-1);
                  }
          }
        printf("y'= %10.5f\n",d[1]);
        getch();
}

float y(float x)
 {
        float a=(exp(x)-exp(-x))/(exp(x)+exp(-x));
        return(a);
 }

float dy(float x)
 {
        float b=(y(x+h)-y(x-h))/(2*h);
        return(b);
 }
```

§2. KHÁI NIỆM VỀ TÍCH PHÂN SỐ

Mục đích của tính tích phân xác định là đánh giá định lượng biểu thức :

$$J = \int_a^b f(x)dx$$



trong đó f(x) là hàm liên tục trong khoảng [a,b] và có thể biểu diễn bởi đường cong y= f(x). Như vậy tích phân xác định J là diện tích S_{ABba} , giới hạn bởi đường cong f(x) , trục hoành , các đường thẳng x = a và x = b . Nếu ta chia đoạn [a,b] thành n phần bởi các điểm x_i thì J là gới hạn của tổng diện tích các hình chữ nhật $f(x_i).(x_{i+1} - x_i)$ khi số điểm chia tiến tới ∝, nghĩa là :

$$J = \lim_{n\to\infty} \sum_{i=0}^{n} f(x_i)(x_{i+1} - x_i)$$

Nếu các điểm chia x_i cách đều , thì (x_{i+1}- x_i) = h . Khi đặt $f(x_o) = f_o$, $f(x_1) = f_1$,... ta có tổng :

$$S_n = h \sum_{i=0}^{n} f_i$$

Khi n rất lớn , S_n tiến tới J . Tuy nhiên sai số làm tròn lại được tích luỹ . Do vậy cần phải tìm phương pháp tính chính xác hơn . Do đó người ta ít khi dùng phương pháp hình chữ nhật như vừa nêu .

§3. PHƯƠNG PHÁP HÌNH THANG

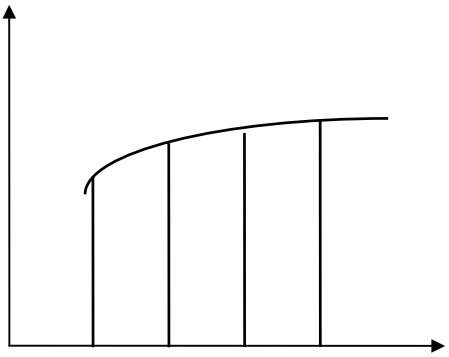
Trong phương pháp hình thang , thay vì chia diện tích S_{ABba} thành các hình chữ nhật , ta lại dùng hình thang . Ví dụ nếu chia thành 3 đoạn như hình vẽ thì :

$S_3 = t_1 + t_2 + t_3$

trong đó t_i là các diện tích nguyên tố . Mỗi diện tích này là một hình thang :

$t_i = [f(x_i) + f(x_{i-1})]/ (2h)$

$= h(f_i - f_{i-1}) / 2$



Như vậy :

$S_3 = h[(f_o+f_1)+(f_1+f_2)+(f_2+f_3)] / 2$

$= h[f_o+2f_1+2f_2+f_3] / 2$

Một cách tổng quát chúng ta có :

$$S_n = \frac{b-a}{n}(f_o + 2f_1 + \cdots + 2f_{n-1} + 2f_n)$$

hay :

$$S_n = \frac{b-a}{n}\{f_o+f_n+2\sum_{i=1}^{n} f_i\}$$

Một cách khác ta có thể viết :

$$\int_a^b f(x)dx = \sum_{k=1}^{n-1} \int_{a+kh}^{a+(k+1)h} f(x)dx \approx \sum_{k=0}^{n-1} \{hf(a+kh)/2 + f[a+(k+1)h]/2\}$$

hay :

$$\int_a^b f(x)dx = h\{f(a)/2 + f(a+h) + \cdots + f[a+(n-1)h] + f(b)/2\}$$

Chương trình tính tích phân theo phương pháp hình thang như sau :

Chương trình 12-2

```
//tinh tich phan bang phuong phap hinh_thang;
#include <conio.h>
#include <stdio.h>
#include <math.h>

float f(float x)
  {
        float a=exp(-x)*sin(x);
        return(a);
  };
```

```
void main()
 {
        int i,n;
        float a,b,x,y,h,s,tp;

        clrscr();
        printf("Tinh tich phan theo phuong phap hinh thang\n");
        printf("Cho can duoi a = ");
        scanf("%f",&a);
        printf("Cho can tren b = ");
        scanf("%f",&b);
        printf("Cho so buoc n = ");
        scanf("%d",&n);
        h=(b-a)/n;
        x=a;
        s=(f(a)+f(b))/2;
        for (i=1;i<=n;i++)
          {
                x=x+h;
                s=s+f(x);
          }
        tp=s*h;
        printf("Gia tri cua tich phan la : %10.6f\n",tp);
        getch();
 }
```

Dùng chương trình này tính tích phân của hàm cho trong function trong khoảng [0 , 1] với 20 điểm chia ta có J = 0.261084.

§4. CÔNG THỨC SIMPSON

Khác với phương pháp hình thang , ta chia đoạn [a,b] thành 2n phần đều nhau bởi các điểm chia x_i :

$a = x_o < x_1 < x_2 < ....< x_{2n} = b$

$x_i = a+ih$; $h = (b - a)/ 2n$ với i =0 , . . , 2n

Do $y_i = f(x_i)$ nên ta có :

$$\int_a^b f(x)dx = \int_{x_0}^{x_2} fdx + \int_{x_2}^{x_4} fdx + ... + \int_{x_{2n-2}}^{x_{2n}} fdx$$

Để tính tích phân này ta thay hàm f(x) ở vế phải bằng đa thức nội suy Newton tiến bậc 2 :

$$P_2(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2t}\Delta^2 y_0$$

và với tích phân thứ nhất ta có :

$$\int_{x_0}^{x_2} f(x)dx = \int_{x_0}^{x_2} P_2(x)dx$$

Đổi biến $x = x_0+th$ thì $dx = hdt$, với x_0 thì t =0 và với x_2 thì t = 2 nên :

$$\int_{x_0}^{x_2} P_2(x)dx = h\int_0^2 (y_0 + t\Delta y_0 + \frac{1(t-1)}{2}\Delta^2 y_0)dt$$

$$= h[y_0 t + \frac{t^2}{2}\Delta y_0 + \frac{1}{2}(\frac{t^3}{3} - \frac{t^2}{2})\Delta^2 y_0]|_{t=0}^{t=2}$$

$$= h[2y_0 + 2\Delta y_0 + \frac{1}{2}(\frac{8}{3} - \frac{4}{2})\Delta^2 y_0]$$

$$= \frac{h}{3}[y_0 + 4y_1 + y_2]$$

Đối với các tích phân sau ta cũng có kết quả tương tự :

$$\int_{x_{2i}}^{x_{2i+2}} f(x)dx = \frac{h}{3}[y_{2i} + 4y_{2i+1} + y_{2i+2}]$$

Cộng các tích phân trên ta có :

$$\int_a^b f(x)dx = \frac{h}{3}[y_o + 4(y_1 + y_3 + \cdots + y_{2n-1}) + 2(y_2 + y_4 + \cdots + y_{2n-2}) + y_{2n}]$$

Chương trình dùng thuật toán Simpson như sau :

Chương trình 12-3

```
//Phuong phap Simpson;
#include <conio.h>
#include <stdio.h>
#include <math.h>

float y(float x)
 {
        float a=4/(1+x*x);
        return(a);
 }

void main()
 {
        int i,n;
        float a,b,e,x,h,x2,y2,x4,y4,tp;

        clrscr();
        printf("Tinh tich phan theo phuong phap Simpson\n");
        printf("Cho can duoi a = ");
        scanf("%f",&a);
        printf("Cho can tren b = ");
        scanf("%f",&b);
        printf("Cho so diem tinh n = ");
        scanf("%d",&n);
        h=(b-a)/n;
        x2=a+h;
        x4=a+h/2;
        y4=y(x4);
        y2=y(x2);
        for (i=1;i<=n-2;i++)
          {
```

```
            x2+=h;
            x4+=h;
            y4+=y(x4);
            y2+=y(x2);
        }
    y2=2*y2;
    y4=4*(y4+y(x4+h));
    tp=h*(y4+y2+y(a)+y(b))/6;
    printf("Gia tri cua tich phan la : %10.8f\n",tp);
    getch();
}
```

Dùng chương trình này tính tích phân của hàm trong function trong đoạn [0,1] với 20 khoảng chia cho ta kết quả J = 3.14159265.

CHƯƠNG 13 : GIẢI PHƯƠNG TRÌNH VI PHÂN

§1.BÀI TOÁN CAUCHY

Một phương trình vi phân cấp 1 có thể viết dưới dạng giải được $y' = f(x,y)$ mà ta có thể tìm được hàm y từ đạo hàm của nó.Tồn tại vô số nghiệm thoả mãn phương trình trên.Mỗi nghiệm phụ thuộc vào một hằng số tuỳ ý.Khi cho trước giá trị ban đầu của y là y_o tại giá trị đầu x_o ta nhận được một nghiệm riêng của phương trình.Bài toán Cauchy (hay bài toán có điều kiện đầu) tóm lại như sau : cho x sao cho $b \geq x \geq a$,tìm y(x) thoả mãn điều kiện :

$$\begin{cases} y'(x) = f(x,y) \\ y(a) = \alpha \end{cases} \qquad (1)$$

Người ta chứng minh rằng bài toán này có một nghiệm duy nhất nếu f thoả mãn điều kiện Lipschitz :

$$\left| f(x,y_1) - f(x,y_2) \right| \leq L \left| y_1 - y_2 \right|$$

với L là một hằng số dương.

Người ta cũng chứng minh rằng nếu f'_y (đạo hàm của f theo y) là liên tục và bị chặn thì f thoả mãn điều kiện Lipschitz.

Một cách tổng quát hơn,người ta định nghĩa hệ phương trình bậc 1 :

$$y_1' = f_1(x, y_1, y_2, \ldots, y_n)$$

$$y_2' = f_2(x, y_1, y_2, \ldots, y_n)$$

.....

$$y_n' = f_n(x, y_1, y_2, \ldots, y_n)$$

Ta phải tìm nghiệm $y_1,y_2,...,y_n$ sao cho :

$$\begin{cases} Y'(x) = f(x,Y) \\ Y(a) = \alpha \end{cases}$$

với :

$$Y' = \begin{pmatrix} y_1' \\ y_2' \\ . \\ . \\ y_n' \end{pmatrix} \qquad F = \begin{pmatrix} f_1 \\ f_2 \\ . \\ . \\ f_n \end{pmatrix} \qquad Y = \begin{pmatrix} y_1 \\ y_2 \\ . \\ . \\ y_n \end{pmatrix}$$

Nếu phương trình vi phân có bậc cao hơn (n),nghiệm sẽ phụ thuộc vào n hằng số tuỳ ý.Để nhận được một nghiệm riêng,ta phải cho n điều kiện đầu.Bài toán sẽ có giá trị đầu nếu với giá trị x_o đã cho ta cho $y(x_o),y'(x_o),y''(x_o),....$

Một phương trình vi phân bậc n có thể đưa về thành một hệ phương trình vi phân cấp 1.Ví dụ nếu ta có phương trình vi phân cấp 2 :

$$\begin{cases} y'' = f(x,y,y') \\ y(a) = \alpha \quad , \quad y'(a) = \beta \end{cases}$$

Khi đặt u = y và v = y′ ta nhận được hệ phương trình vi phân cấp 1 :

$$\begin{cases} u' = v \\ v' = g(x,u,v) \end{cases}$$

tới điều kiện đầu : $u(a) = \alpha$ và $v(a) = \beta$

Các phương pháp giải phương trình vi phân được trình bày trong chương này là

các phương pháp rời rạc : đoạn [a,b] được chia thành n đoạn nhỏ bằng nhau được gọi là các "bước" tích phân h = (b - a) / n.

§2.PHƯƠNG PHÁP EULER VÀ EULER CẢI TIẾN

Giả sử ta có phương trình vi phân :

$$\begin{cases} y'(x)=f(x,y) \\ y(a)=\alpha \end{cases} \qquad (1)$$

và cần tìm nghiệm của nó.Ta chia đoạn $[x_o,x]$ thành n phần bởi các điểm chia :

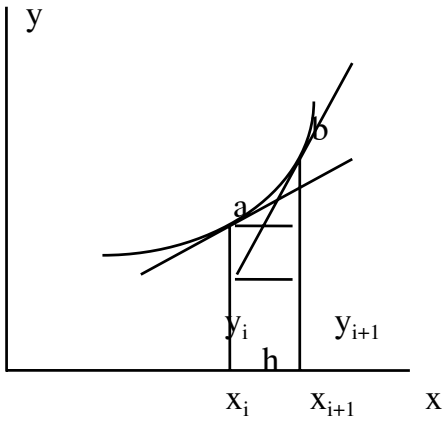
$$x_o < x_1 < x_2 <...< x_n = x$$

Theo công thức khai triển Taylor một hàm lân cận x_i ta có :

$$y(x_{i+1}) = y(x_i)+(x_{i+1}-x_i)y'(x_i) + \frac{(x_{i+1}-x_i)^2 y''(x_i)}{2} + \frac{(x_{i+1}-x_i)^3 y'''(x_i)}{6} + \cdots$$

Nếu $(x_{i+1} - x_i)$ khá bé thì ta có thể bỏ qua các số hạng $(x_{i+1} - x_i)^2$ và các số hạng bậc cao

$$y(x_{i+1}) = y(x_i) + (x_{i+1}- x_i)\, y'(x_i)$$



Trường hợp các mốc cách đều : $(x_{i-1} - x_i) = h = (x - x_o)/ n$ thì ta nhận được công thức Euler đơn giản :

$$y_{i+1} = y_i + hf(x_i,y_i) \qquad (2)$$

Về mặt hình học ta thấy (1) cho kết quả càng chính xác nếu bước h càng nhỏ.Để tăng độ chính xác ta có thể dùng công thức Euler cải tiến.Trước hết ta nhắc lại định lí Lagrange:

Giả sử f(x) là hàm liên tục trong[a,b] và khả vi trong (a,b)thì có ít nhất một điểm $c\in(a,b)$ *để cho :*

$$f'(c) = \frac{f(b)-f(a)}{b-a}$$

Theo định lí Lagrange ta có :

$$y(x_{i+1}) = y(x_i) + hf(c_i, y(c_i))$$

Như vậy với $c\in(x_i,x_{i+1})$ ta có thể thay :

$$f(c_i, y(c_i)) = \frac{1}{2}\left[f(x_i, y_i) + f(x_{i+1}, y_{i+1})\right]$$

Từ đó ta có công thức Euler cải tiến :

$$y_{i+1} = y_i + \frac{h}{2}[f(x_i,y_i) + f(x_{i+1},y_{i+1})] \qquad (3)$$

Trong công thức này giá trị y_{i+1} chưa biết.Do đó khi đã biết y_i ta phải tìm y_{i+1} bằng cách giải phương trình đại số tuyến tính (3).Ta thường giải (3) bằng cách lặp như sau:trước hết chọn xấp xỉ đầu tiên của phép lặp $y_{i+1}^{(0)}$ chính là giá trị y_{i+1} tính được theo phương pháp Euler sau đó dùng (3) để tính các $y_{i+1}^{(s)}$,cụ thể là :

$$y_{i+1}^{(0)} = y_i + hf(x_i, y_i)$$
$$y_{i+1}^{(s)} = y_i + \frac{h}{2}\left[f(x_i, y_i) + f(x_{i+1}, y_{i+1}^{(s-1)})\right]$$

Quá trình tính kết thúc khi $y_i^{(s)}$ đủ gần $y_i^{(s-1)}$
Chương trình giải phương trình vi phân theo phương pháp Euler như sau :

Chương trình 13-1

```
//pp_Euler;
#include <conio.h>
#include <stdio.h>
#include <math.h>

float f(float x,float y)
  {
        float a=x+y;
        return(a);
  }

void main()
  {
    int i,n;
        float a,b,t,z,h,x0,y0,c1,c2;
        float x[100],y[100];

        clrscr();
        printf("Cho can duoi a = ");
        scanf("%f",&a);
        printf("Cho can tren b = ");
        scanf("%f",&b);
        printf("Cho so buoc tinh n = ");
        scanf("%d",&n);
        printf("Cho so kien x0 = ");
        scanf("%f",&x0);
        printf("Cho so kien y0 = ");
        scanf("%f",&y0);
        printf("\n");
        printf("Bang ket qua\n");
        printf("\n");
        printf("Phuong phap Euler\n");
        h=(b-a)/n;
        x[1]=x0;
        y[1]=y0;
        printf("  x           y");
        printf("\n");
        for (i=1;i<=n+1;i++)
          {
                x[i+1]=x[i]+h;
                y[i+1]=y[i]+h*f(x[i],y[i]);
                printf("%3.2f%16.3f",x[i],y[i]);
                printf("\n");
          }
```

```
        printf("\n");
        getch();
        printf("Phuong phap Euler cai tien\n");
        printf("  x          y");
        printf("\n");
        for (i=1;i<=n+1;i++)
          {
                x[i+1]=x[i]+h;
                c1=h*f(x[i],y[i]);
                c2=h*f(x[i]+h,y[i]+c1);
                y[i+1]=y[i]+(c1+c2)/2;
                printf("%3.2f%15.5f",x[i],y[i]);
                printf("\n");
          }
        getch();
}
```

Với phương trình cho trong function và điều kiện đầu $x_o = 0, y_o = 0$, nghiệm trong đoạn [0,1] với 10 điểm chia là :

x	y(Euler)	y(Euler cải tiến)
0.0	0.00	0.00
0.1	0.00	0.01
0.2	0.01	0.02
0.3	0.03	0.05
0.4	0.06	0.09
0.5	0.11	0.15
0.6	0.17	0.22
0.7	0.25	0.31
0.8	0.34	0.42
0.9	0.46	0.56
1.0	0.59	0.71

§3.PHƯƠNG PHÁP RUNGE-KUTTA

Xét bài toán Cauchy (1).Giả sử ta đã tìm được giá trị gần đúng y_i của $y(x_i)$ và muốn tính y_{i+1} của $y(x_{i+1})$.Trước hết ta viết công thức Taylor :

$$y(x_{i+1}) = y(x_i) + hy'(x_i) + \frac{h^2}{2}y''(x_i) + ... + \frac{h^m}{m!}y^{(m)}(x_i) + \frac{h^{(m+1)}}{(m+1)!}y^{(m+1)}(c) \qquad (11)$$

với $c \in (x_i, x_{i+1})$ và :

$$y'(x_i) = f[x_i, y(x_i)]$$

$$y^{(k)}(x_i) = \frac{d^{k-1}}{dx^{k-1}}[f(x, y(x)]_{x=x_i}$$

Ta viết lại (11) dưới dạng :

$$y_{i+1} - y_i = hy_i' + \frac{h^2}{2}y_i'' + ... + \frac{h^m}{m!}y_i^{(m)} + \frac{h^{(m+1)}}{(m+1)!}y^{(m+1)}(c) \qquad (12)$$

Ta đã kéo dài khai triển Taylor để kết quả chính xác hơn.Để tính y'_i, y''_i v.v.ta có thể dùng phương pháp Runge-Kutta bằng cách đặt :

$$y_{i+1} - y_i = r_1 k_1^{(i)} + r_2 k_2^{(i)} + r_3 k_3^{(i)} + ... + r_s k_s^{(i)} \quad (13)$$

trong đó :

$$\begin{cases} k_1^{(i)} = hf(x_i, y_i) \\ k_2^{(i)} = hf(x_i + ah\ , y_i + \alpha k_1^{(i)}) \\ k_3^{(i)} = hf(x_i + bh\ , y_i + \beta k_1^{(i)} + \gamma k_2^{(i)}) \\ \cdots\cdots\cdots\cdots\cdots \end{cases} \quad (14)$$

và ta cần xác định các hệ số a,b,..;α,β,γ,...; r_1,r_2,..sao cho vế phải của (13) khác với vế phải của (12) một vô cùng bé cấp cao nhất có thể có đối với h.

Khi dùng công thức Runge-Kutta bậc hai ta có :

$$\begin{cases} k_1^{(i)} = hf(x_i, y_i) \\ k_2^{(i)} = hf(x_i + ah\ , y_i + \alpha k_1^{(i)}) \end{cases} \quad (15)$$

và $$y_{i+1} - y_i = r_1 k_1^{(i)} + r_2 k_2^{(i)} \quad (16)$$

Ta có : $y'(x) = f[x,y(x)]$

$y''(x) = f'_x[x,y(x)] + f'_y[x,y(x)]y'(x)$

.....

Do đó vế phải của (12) là :

$$hf(x_i, y_i) + \frac{h^2}{2}[f'_x(x_i, y_i) + f'_y(x_i, y_i)\ y'(x)] + ... \quad (17)$$

Mặt khác theo (15) và theo công thức Taylor ta có :

$$k_1^{(i)} = hf(x_i, y_i) = hy'_i$$

$$k_2^{(i)} = h[f(x_i, y_i) + ahf'_x(x_i, y_i) + \alpha k_1^{(i)} f'_y(x_i, y_i) + ...]$$

Do đó vế phải của (16) là :

$$h(r_1 + r_2)f(x_i, y_i) + h^2[ar_2 f'_x(x_i, y_i) + \alpha r_2 y'_i f'_x(x_i, y_i)] + ... \quad (18)$$

Bây giờ cho (17) và (18) khác nhau một vô cùng bé cấp $O(h^3)$ ta tìm được các hệ số chưa biết khi cân bằng các số hạng chứa h và chứa h^2 :

$r_1 + r_2 = 1$

$a.r_1 = 1/2$

$\alpha.r_2 = 1$

Như vậy : $\alpha = a, r_1 = (2a - 1)/2a, r_2 = 1/2a$ với a được chọn bất kì.

Nếu a = 1 / 2 thì $r_1 = 0$ và $r_2 = 1$.Lúc này ta nhận được công thức Euler.Nếu a = 1 thì $r_1 = 1/2$ và $r_2 = 1/2$.Lúc này ta nhận được công thức Euler cải tiến.

Một cách tương tự chúng ta nhận được công thức Runge - Kutta bậc 4.Công thức này hay được dùng trong tính toán thực tế :

$k_1 = h.f(x_i, y_i)$

$k_2 = h.f(x_i + h/2, y_i + k_1/2)$

$k_3 = h.f(x_i + h/2, y_i + k_2/2)$

$k_4 = h.f(x_i + h, y_i + k_3)$

$y_{i+1} = y_i + (k_1 + 2k_2 + 2k_3 + k_4) / 6$

Chương trình giải phương trình vi phân bằng công thức Runge - Kutta bậc 4 như sau :

Chương trình 11-2

```
//Phuong phap Runge_Kutta;
```

```
#include <conio.h>
#include <stdio.h>
#include <math.h>
#define k 10

float f(float x,float y)
  {
        float a=x+y;
        return(a);
  }

void main()
  {
        float a,b,k1,k2,k3,k4;
        int i,n;
        float x0,y0,h,e;
        float x[k],y[k];

        clrscr();
        printf("Phuong phap Runge - Kutta\n");
        printf("Cho can duoi a = ");
        scanf("%f",&a);
        printf("Cho can tren b = ");
        scanf("%f",&b);
        printf("Cho so kien y0 = ");
        scanf("%f",&y[0]);
        printf("Cho buoc tinh h = ");
        scanf("%f",&h);
        n=(int)((b-a)/h);
        printf("          x              y\n");
        for (i=0;i<=n+1;i++)
          {
                x[i]=a+i*h;
                k1=h*f(x[i],y[i]);
                k2=h*f((x[i]+h/2),(y[i]+k1/2));
                k3=h*f((x[i]+h/2),(y[i]+k2/2));
                k4=h*f((x[i]+h),(y[i]+k3));
                y[i+1]=y[i]+(k1+2*k2+2*k3+k4)/6;
                printf("%12.1f%16.4f\n",x[i],y[i]);
          }
        getch();
  }
```

Kết quả tính toán với f = x + y,h = 0.1,a = 0,b =1,y_o = 1 là :

x	y
0.0	1.0000
0.1	1.1103
0.2	1.2427
0.3	1.3996
0.4	1.5834

0.5	1.7971
0.6	2.0440
0.7	2.3273
0.8	2.6508
0.9	3.0190
1.0	3.4362

CHƯƠNG 14 : TỐI ƯU HOÁ

§1.PHƯƠNG PHÁP TỈ LỆ VÀNG

Trong chương 8 chúng ta đã xét bài toán tìm nghiệm của phương trình phi tuyến tức là tìm giá trị của x mà tại đó hàm triệt tiêu.Trong phần này chúng ta sẽ đặt vấn đề tìm giá trị của x mà tại đó hàm đạt giá trị cực trị(cực đại hay cực tiểu).Phương pháp tiết diện vàng là một phương pháp đơn giản và hiệu quả để tìm giá trị cực trị của hàm.

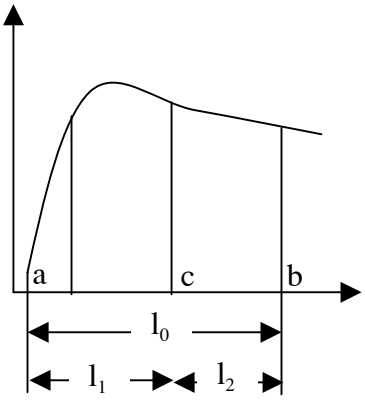
Giả sử ta có hàm y = f(x) và cần tìm giá trị cực trị trong khoảng [a,b].Khi tìm nghiệm chỉ cần biết 2 giá trị của hàm là ta khẳng định được nghiệm có nằm trong khoảng đã cho hay không bằng cách xét dấu của hàm.Khi tìm giá trị cực trị ta phải biết thêm một giá trị nữa của hàm trong khoảng [a,b] thì mới khẳng định được hàm có đạt cực trị trong đoạn đã cho hay không.Sau đó ta chọn thêm một điểm thứ tư và xác định xem giá trị cực trị của hàm sẽ nằm trong đoạn nào.

Theo hình vẽ,khi chọn điểm trung gian c ta có :

$$l_1 + l_2 = l_0 \qquad (1)$$

và để tiện tính toán ta chọn :

$$\frac{l_1}{l_0} = \frac{l_2}{l_1} \qquad (2)$$



Thay thế (1) vào (2) ta có :

$$\frac{l_1}{l_1 + l_2} = \frac{l_2}{l_1} \qquad (3)$$

Gọi $r = \frac{l_2}{l_1}$,ta nhận được phương trình :

$$1 + r = \frac{1}{r} \qquad (4)$$

hay :

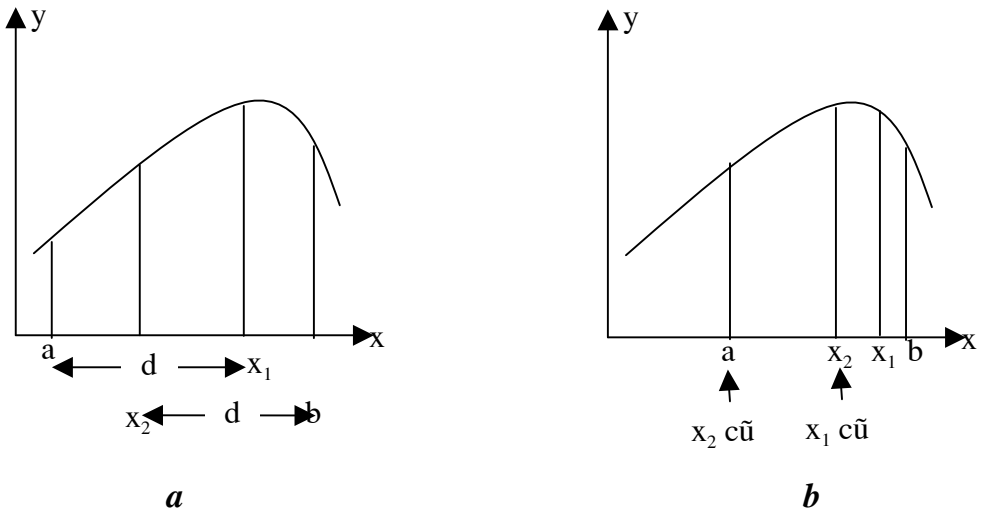
$$r^2 + r - 1 = 0 \qquad (5)$$

Nghiệm của phương trình (5) là :

$$r = \frac{-1 + \sqrt{1 - 4(-1)}}{2} = \frac{\sqrt{5} - 1}{2} = 0.61803... \qquad (6)$$

Giá trị này đã được biết từ thời cổ đại và được gọi là "tỉ lệ vàng".Như trên đã nói,phương pháp tỉ lệ vàng được bắt đầu bằng 2 giá trị đã cho của biến x là a và b.Sau đó ta chọn 2 điểm x_1 và x bên trong khoảng [a,b] theo tỉ lệ vàng:

$$d = \frac{\sqrt{5} - 1}{2} = 0.61803...$$



a ***b***

Ta tính giá trị của hàm tại các điểm bên trong đoạn [a,b].Kết quả có thể là một trong các khả năng sau :

1. Nếu,như trường hợp hình a,$f(x_1) > f(x_2)$ thì giá trị cực trị của hàm nằm trong $[x_2,b]$ và x_2 trở thành a và ta tính tiếp.
2. Nếu $f(x_1) < f(x_2)$ thì thì giá trị cực trị của hàm nằm trong $[a,x_1]$ và x_1 trở thành b và ta tính tiếp.

Cái lợi của phương pháp tỉ lệ vàng theo hình a là giá trị x_1 cũ trở thành giá trị x_2 mới nên giá trị $f(x_2)$ mới chính là giá trị $f(x_1)$ cũ nên ta không cần tính lại nó.Chương trình mô tả thuật toán trên như sau:

Chương trình 14-1

```
//tiet_dien_vang;
#include <conio.h>
#include <stdio.h>
#include <math.h>

float eps=1e-6;

float f(float x)
  {
        float a=2*sin(x)-x*x/10;
        return(a);
  };

float max(float xlow,float xhigh)
  {
        float xl,xu,r,d,x1,x2,f1,f2,xopt,s;
        int lap;
        xl=xlow;
        xu=xhigh;
        lap=1;
```

```
        r=(sqrt(5.0)-1.0)/2.0;
        d=r*(xu-xl);
        x1=xl+d;
        x2=xu-d;
        f1=f(x1);
        f2=f(x2);
        if (f1>f2)
          xopt=x1;
        else
          xopt=x2;
        do
          {
                d=r*d;
                if (f1>f2)
                  {
                        xl=x2;
                        x2=x1;
                        x1=xl+d;
                        f2=f1;
                        f1=f(x1);
                  }
                else
                  {
                        xu=x1;
                        x1=x2;
                        x2=xu-d;
                        f1=f2;
                        f2=f(x2);
                  }
                lap=lap+1;
                if (f1>f2)
                  xopt=x1;
                else
                  xopt=x2;
                if (xopt!=0)
                  s=(1.0-r)*fabs((xu-xl)/xopt)*100;
          }
        while((s>eps)&&(lap<=20));
        float k=xopt;
        return(k);
  }

float min(float xlow,float xhigh)
  {
        float xl,xu,r,d,x1,x2,f1,f2,fx,xopt,s;
        int lap;
        xl=xlow;
```

```
        xu=xhigh;
        lap=1;
        r=(sqrt(5.0)-1.0)/2,0;
        d=r*(xu-xl);
        x1=xl+d;
        x2=xu-d;
        f1=f(x1);
        f2=f(x2);
        if (f1<f2)
          xopt=x1;
        else
          xopt=x2;
        do
          {
                d=r*d;
                if (f1<f2)
                  {
                        xl=x2;
                        x2=x1;
                        x1=xl+d;
                        f2=f1;
                        f1=f(x1);
                  }
                else
                  {
                        xu=x1;
                        x1=x2;
                        x2=xu-d;
                        f1=f2;
                        f2=f(x2);
                  }
                lap=lap+1;
                if (f1<f2)
                  xopt=x1;
                else
                  xopt=x2;
                if (xopt!=0)
                  s=(1.0-r)*fabs((xu-xl)/xopt)*100;
          }
        while ((s>eps)&&(lap<=20));
        float r1=xopt;
        return(r1);
  }

void main()
  {
        float x,y,xlow,xhigh,eps;
```

```
        clrscr();
        printf("TIM CUC TRI CUA HAM BANG PHUONG PHAP TIET DIEN VANG\n");
        printf("\n");
        printf("Cho khoang can tim cuc tri\n");
        printf("Cho can duoi a = ");
        scanf("%f",&xlow);
        printf("Cho can tren b = ");
        scanf("%f",&xhigh);

        if (f(xlow)<f(xlow+0.1))
          {
                x=max(xlow,xhigh);
                y=f(x);
                printf("x cuc dai = %10.5f\n",x);
                printf("y cuc dai = %10.5f\n",y);
          }
        else
          {
                x=min(xlow,xhigh);
                y=f(x);
                printf("x cuc tieu = %10.5f y cuc tieu = %10.5f",x,y);
          }
        getch();
}
```

Trong chương trình này ta cho a = 0 ; b =4 và tìm được giá trị cực đại y = 1.7757 tại x = 1.4276

§2.PHƯƠNG PHÁP NEWTON

Khi tính nghiệm của phương trình f(x) = 0 ta dùng công thức lặp Newton-Raphson :

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

Một cách tương tự,để tìm giá trị cực trị của hàm f(x) ta đặt g(x)=f′(x).Như vậy ta cần tìm giá trị của x để g(x) = 0.Như vậy công thức lặp Newton-Raphson sẽ là :

$$x_{i+1} = x_i - \frac{g(x_i)}{g'(x_i)} = x_i - \frac{f'(x_i)}{f''(x_i)}$$

Các đạo hàm $f'(x_i)$ và $f''(x_i)$ được xác định theo các công thức :

$$f'(x_i) = \frac{f(x_i + h) - f(x_i - h)}{2h}$$

$$f''(x_i) = \frac{f(x_i + h) - 2f(x_i) + f(x_i - h)}{h^2}$$

Tại giá trị f′(x) = 0 hàm đạt giá trị cực đại nếu f″(x) < 0 và cực tiểu nếu f″(x) > 0.Chương trình sau mô tả thuật toán trên.

Chương trình 14-2

```
//Phuong phap New_ton;
#include <conio.h>
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

float f(float x)
  {
        float a=2*sin(x)-x*x/10;
        return(a);
  }

float f1(float x)
  {
        float a=2*cos(x)-x/5.0;
        return(a);
  }

float f2(float x)
  {
        float a=-2*sin(x)-1.0/5.0;
        return(a);
  }

void main()
  {
        float a,eps,x[50],y1,t;
        clrscr();
        printf("TINH CUC TRI BANG PHUONG PHAP NEWTON\n");
        printf("\n");
        printf("Cho diem bat dau tinh a = ");
        scanf("%f",&a);
        eps=1e-6;
        int i=1;
        x[i]=a;
        do
          {
                x[i+1]=x[i]-f1(x[i])/f2(x[i]);
                t=fabs(x[i+1]-x[i]);
                x[i]=x[i+1];
                i++;
                if (i>1000)
                  {
                        printf("Khong hoi tu sau 1000 lan lap");
                        getch();
```

```
                    exit(1);
                }
            }
        while (t>=eps);
        printf("\n");
          y1=f2(x[i]);
        if (y1>0)
          printf("x cuc tieu = %10.5f  y cuc tieu = %10.5f",x[i],f(x[i]));
        else
          printf("x cuc dai = %10.5f y cuc dai = %10.5f",x[i],f(x[i]));
        getch();
 }
```

Ta có kết quả x = 1.42755,y= 1.77573

§3.PHƯƠNG PHÁP PARABOL

Nội dung của phương pháp parabol là ta thay đường cong y = f(x) bằng một đường cong parabol mà ta dễ dàng tìm được giá trị cực trị của nó.Như vậy trong khoảng [a,b] ta chọn thêm một điểm x bất kì và xấp xỉ hàm f(x) bằng parabol qua 3 điểm a,x,và b.Sau đó ta đạo hàm và cho nó bằng 0 để tìm ra điểm cực trị của parabol này.Giá trị đó được tính bằng công thức:

$$x_1 = \frac{f(a)(x^2 - b^2) + f(x)(b^2 - a^2) + f(b)(b^2 - x^2)}{2f(a)(x - b) + 2f(x)(b - a) + 2f(b)(a - x)}$$

Sau đó tương tự phương pháp tỉ lệ vàng ta loại trừ vùng không chứa giá trị cực trị và tiếp tục quá trình trên cho đến khi đạt độ chính xác mong muốn.Chương trình được viết như sau:

Chương trình 14-3

```
//phuong phap parabol
#include <conio.h>
#include <stdio.h>
#include <math.h>

float f(float x)
 {
        float f1=2*sin(x)-x*x/10;
        return(f1);
 }

void main()
 {
        float a,b,x0,x1,x2,x3,f3;

        clrscr();
        printf("TIM CUC TRI BANG PHUONG PHAP PARABOL\n");
        printf("\n");
```

```
        printf("Cho doan can tim cuc tri [a,b]\n");
        printf("Cho diem dau a = ");
        scanf("%f",&a);
        printf("Cho diem cuoi b = ");
        scanf("%f",&b);
        x0=a;
        x2=b;
        x1=(x0+x2)/4;
        do
          {
                x3=(f(x0)*(x1*x1-x2*x2)+f(x1)*(x2*x2-x0*x0)+f(x2)*(x0*x0-x1*x1))
    /(2*f(x0)*(x1-x2)+2*f(x1)*(x2-x0)+2*f(x2)*(x0-x1));
                f3=f(x3);
                if (x3>x1)
                  x0=x1;
                else
                  x2=x1;
                x1=x3;
          }
        while (fabs(x2-x0)>1e-5);
        printf("\n");
        f3=(f(x2+0.01)-2*f(x2)+f(x2-0.01))/(0.01*0.01);
        if (f3<0)
          printf("x cuc dai = %10.5f   y cuc dai = %10.5f",x2,f(x2));
        else
          printf("x cuc tieu = %10.5f  y cuc tieu = %10.5",x2,f(x2));
        getch();
}
```

Chạy chương trình này với a = 0 và b = 4 ta có x cực đại là 1.42755 và y cực đại là 1.77573.

§4. PHƯƠNG PHÁP ĐƠN HÌNH(SIMPLEX METHOD)

Trong thực tế nhiều bài toán kinh tế,vận tải có thể được giải quyết nhờ phương pháp quy hoạch tuyến tính.Trước hết ta xét bài toán lập kế hoạch sản xuất sau:

Một công ty muốn sản xuất 2 loại sản phẩm mới là A và B bằng các nguyên liệu 1,2,và 3.Suất tiêu hao nguyên liệu để sản xuất các sản phẩm cho ở bảng sau:

	Sản phẩm A	Sản phẩm B
Nguyên liệu 1	2	1
Nguyên liệu 2	1	2
Nguyên liệu 3	0	1

Số liệu này cho thấy để sản xuất một đơn vị sản phẩm A cần dùng 2 đơn vị nguyên liệu 1,một đơn vị nguyên liệu 2 và để sản xuất một đơn vị sản phẩm B cần dùng 1 đơn vị

nguyên liệu 1,hai đơn vị nguyên liệu 2,1 đơn vị nguyên liệu 3.Trong kho của nhà máy hiện có dự trữ 8 đơn vị nguyên liệu 1,7 đơn vị nguyên liệu 2 và 3 đơn vị nguyên liệu 3.Tiền lãi một đơn vị sản phẩm A là 4.000.000 đ,một đơn vị sản phẩm B là 5.000.000đ.Lập kế hoạch sản xuất sao cho công ty thu được tiền lãi lớn nhất.

Bài toán này là bài toán tìm cực trị có điều kiện.Gọi x_1 là lượng sản phẩm A và x_2 là lượng sản phẩm B ta đi đến mô hình toán học:

$$f(x) = 4x_1 + 5x_2 \rightarrow \max$$

với các ràng buộc :
$2x_1 + x_2 \le 8$ (ràng buộc về nguyên liệu 1)
$x_1 + 2x_2 \le 7$ (ràng buộc về nguyên liệu 2)
$x_2 \le 3$ (ràng buộc về nguyên liệu 3)
$x_1 \ge 0, x_2 \ge 0$

Một cách tổng quát ta có bài toán được phát biểu như sau : Cho hàm mục tiêu $C^TX \rightarrow \max$ với điều kiện ràng buộc $AX \le B$ và $X \ge 0$.Thuật toán để giải bài toán gồm hai giai đoạn

- tìm một phương án cực biên một đỉnh
- kiểm tra điều kiện tối ưu đối với phương án tìm được ở giai đoạn 1.Nếu điều kiện tối ưu được thoả mãn thì phương án đó là tối ưu.Nếu không ta chuyển sang phương án mới.

Chương trình giải bài toán được viết như sau :

Chương trình 14-4

```
//simplex;
#include <conio.h>
#include <stdio.h>

int m,n,n1,it,i,j,h1,h2,hi,m1,ps,pz,v,p;
float bv[20];
float a[20][20];
float h,mi,x,z;

void don_hinh()
  {
        int t;
        float hi;
        if (p!=2)
          for (i=1;i<=m;i++)
                bv[i]=n+i;
        if (p==2)
          {
                h1=n;
                h2=m;
          }
        else
          {
                h1=m;
                h2=n;
```

```
  }
for (i=1;i<=m1;i++)
  for (j=1;j<=h1;j++)
        {
          a[i][h2+j]=0.0;
          if (i==j)
                a[i][h2+j]=1.0;
        }
it=0;
t=1;
while (t)
  {
        it=it+1;
        if (it<(m*n*5))
          {
                mi=a[m1][1];
                ps=1;
                for (j=2;j<=n1-1;j++)
                  if (a[m1][j]<mi)
                        {
                          mi=a[m1][j];
                          ps=j;
                        }
                if (mi>-0.00001)
                  {
                        z=a[m1][n1];
                        t=0;
                  }
                mi=1e+20;
                pz=0;
                for (i=1;i<=m1-1;i++)
                  {
                        if (a[i][ps]<=0.0)
                          continue;
                        h=a[i][n1]/a[i][ps];
                        if (h<mi)
                          {
                                mi=h;
                                pz=i;
                          }
                  }
                if (pz==0)
                  {
                        if (p==2)
                          {
                                printf("Khong ton tai nghiem\n");
                                t=0;
```

```
                        }
                    else
                        {
                            printf("Nghiem khong bi gioi han\n");
                            t=0;
                        }
                }
            if (p==1)
              bv[pz]=ps;
            hi=a[pz][ps];
            for (j=1;j<=n1;j++)
              a[pz][j]=a[pz][j]/hi;
            if (pz!=1)
              for (i=1;i<=pz-1;i++)
                    {
                      hi=a[i][ps];
                      for (j=1;j<=n1;j++)
                            a[i][j]=a[i][j]-hi*a[pz][j];
                    }
            for (i=pz+1;i<=m1;i++)
              {
                    hi=a[i][ps];
                    for (j=1;j<=n1;j++)
                      a[i][j]=a[i][j]-hi*a[pz][j];
              }
          }
        else
          printf("Nghiem bat thuong");
    }
}

void main()
 {
    clrscr();
    printf("PHUONG PHAP DON HINH\n");
    printf("\n");
    flushall();
    printf("Cho bai toan tim max(1) hay min(2)(1/2)? : ");
    scanf("%d",&p);
    printf("Cho so bien n = ");
    scanf("%d",&n);
    printf("Cho so dieu kien bien m = ");
    scanf("%d",&m);
    n1=n+m+1;
    if (p==2)
      m1=n+1;
    else
```

```
  m1=m+1;
printf("Cho ma tran cac dieu kien bien\n");
for (i=1;i<=m;i++)
  for (j=1;j<=n;j++)
          if (p==2)
            {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[j][i]);
            }
          else
            {
                  printf("a[%d][%d] = ",i,j);
                  scanf("%f",&a[i][j]);
            }
printf("\n");
printf("Cho ma tran ve phai\n");
for (i=1;i<=m;i++)
  if (p==2)
          {
            printf("b[%d] = ",i);
            scanf("%f",&a[m1][i]);
  }
else
  {
          printf("b[%d] = ",i);
          scanf("%f",&a[i][n1]);
  }
printf("\n");
printf("Cho ham muc tieu\n");
for (j=1;j<=n;j++)
  if (p==2)
          {
            printf("z[%d] = ",j);
            scanf("%f",&a[j][n1]);
          }
  else
          {
            printf("z[%d] = ",j);
            scanf("%f",&a[m1][j]);
          }
if (p==2)
  hi=m;
else
  hi=n;
for (j=1;j<=hi;j++)
   a[m1][j]=-a[m1][j];
a[m1][n1]=0.0;
```

```
        don_hinh();
        printf("\n");
        printf("NGHIEM TOI UU HOA\n");
        if (p==2)
          printf("Bai toan cuc tieu tieu chuan\n");
        else
          printf("Bai toan cuc dai tieu chuan\n");
        printf("sau %d buoc tinh",it);
        printf("\n");
        for (j=1;j<=n;j++)
          {
                if (p==2)
                  x=a[m1][m+j];
                else
                  {
                        v=0;
                        for (i=1;i<=m;i++)
                          if (bv[i]==j)
                                {
                                  v=i;
                                  i=m;
                                }
                        if (v==0)
                          x=0.0;
                        else
                          x=a[v][n1];
                  }
                printf("x[%d] = %10.5f\n",j,x);
          }
        printf("\n");
        printf("Gia tri toi uu cua ham muc tieu = %10.5f\n",z);
        getch();
  }
```

Dùng chương trình này giải bài toán có hàm mục tiêu :

$$z = 80x_1 + 56x_2 + 48x_3 \rightarrow \min$$

với ràng buộc :

$$3x_1 + 4x_2 + 2x_3 \geq 15$$
$$2x_1 + 3x_2 + x_3 \geq 9$$
$$x_1 + 2x_2 + 6x_3 \geq 18$$
$$x_2 + x_3 \geq 5$$
$$x_1, x_2, x_3 \geq 0$$

Ta cần nhập vào chương trình là tìm min,với số biến n =3,số điều kiện biên m = 4,các hệ số a[1,1] = 3 ; a[1,2] = 4 ; a[1,3] = 2 ; a[2,1] = 2; a[2,2] = 3 ; a[2,3] = 1 ; a[3,1] = 1 ; a[3,2] = 2 ; a[3,3] = 6 ; a[4,1] = 0 ; a[4,2] = 1 ; a[4,3] = 1 ; b[1] = 15 ; b[2] = 9 ; b[3] = 18; b[4] = 5 ; z[1] = 80 ; z[2] = 56 ; z[3] = 48 và nhận được kết quả :

x[1] = 0 ; x[2] = 2.5 ; x[3] =2.5 và trị của hàm mục tiêu là 260

§5.PHƯƠNG PHÁP THẾ VỊ

Trong vận tải ta thường gặp bài toán vận tải phát biểu như sau : có n thùng hàng của một hãng xây dựng cần chuyển tới n địa điểm khác nhau.Giá vận tới tới mỗi địa điểm đã cho.Tìm phương án vận chuyển để giá thành là cực tiểu.

Một cách tổng quát bài toán được phát biểu :

$$\sum a_i p_i \rightarrow \min$$

Ví dụ : Cần vận chuyển 6 thùng hàng tới 6 địa điểm với giá thành cho ở bảng sau :

Thùng	1	2	3	4	5	6 → địa điểm
1	60	35	28	53	29	26
2	81	43	37	23	36	45
3	42	42	33	47	43	51
4	29	70	42	53	48	37
5	81	69	40	66	69	60
6	10	21	32	31	24	27

Để giả bài toán ta dùng thuật toán Hungary như sau :

- trừ mỗi dòng cho số min của dòng đó ta có :

34	9	2	27	3	0
58	20	14	0	13	22
9	9	0	14	10	18
0	41	13	24	19	8
41	29	0	26	29	20
0	11	22	21	14	17

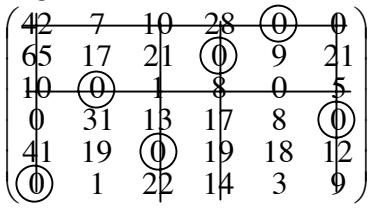
- trừ mỗi cột cho số min của cột đó

34	0	2	27	0	0
58	11	14	0	10	22
9	0	0	14	7	12
0	32	13	24	16	8
41	20	0	26	26	20
0	2	22	21	11	17

Mục tiêu của thuật toán Hungary là biến đổi ma trận giá thành sao cho có thể đọc giá trị tối ưu từ ma trận.Điều này được thực hiện khi mỗi hànhg và cột chứa ít nhất một số 0.Nếu ta vẽ một đoạn thẳng qua mỗi hàng và cột chứa số 0 thì khi đó số đoạn thẳng tối thiểu qua tất cả các số 0 phải là 6.Trong ma trận trên ta chỉ mới dùng 5 đoạn thẳng nghĩa là chưa có giá trị tối ưu.Để biến đổi tiếp tục ta tìm trị min của các phần tử chưa nằm trên bất kì đoạn thẳng nào.Trị số đó là 7.Lấy các phần tử không nằm trên đoạn thẳng nào trừ đi 7 và công các phần tử nằm trên hai đoạn thẳng với 7 ta có ma trận :

41	7	9	27	0	0
65	18	21	0	10	22
9	0	0	7	0	5
0	32	13	17	9	1
41	20	0	19	19	13
0	2	22	14	4	10

Do số đoạn thẳng tối thiểu còn là 5 nên ta lặp lại bước trên và nhận được ma trận mới :



Số đoạn thẳng cần để qua hết các số 0 là 6 nghĩa là ta đã tìm được trị tối ưu.Ta đánh dấu 6 số 0 sao cho mỗi hàng và mỗi cột chỉ có 1 số được đánh dấu.Chỉ số các số 0 được đánh dấu cho ta trị tối ưu :

a_{15} = 0 nghĩa là thùng 1 được vận chuyển tới địa điểm 5
a_{24} = 0 nghĩa là thùng 2 được vận chuyển tới địa điểm 4
a_{32} = 0 nghĩa là thùng 3 được vận chuyển tới địa điểm 2
a_{46} = 0 nghĩa là thùng 4 được vận chuyển tới địa điểm 6
a_{53} = 0 nghĩa là thùng 5 được vận chuyển tới địa điểm 3
a_{61} = 0 nghĩa là thùng 6 được vận chuyển tới địa điểm 1

Chương trình viết theo thuật toán trên như sau :

Chương trình 14-5

```
// van_tru;
#include <conio.h>
#include <stdio.h>

void main()
 {
        int a[20][20],z[20][20],p[20][2];
        float x[20][20],w[20][20];
        float c[20],r[20];
        int t,c1,i,j,k,k2,k3,k5,l,l1,m,n,r1,s;
        float m1,q;

   clrscr();
        printf("Cho so an so n = ");
        scanf("%d",&n);
        printf("Cho cac he so cua ma tran x\n");
        for (i=1;i<=n;i++)
          for (j=1;j<=n;j++)
                {
                  printf("x[%d][%d] = ",i,j);
                  scanf("%f",&x[i][j]);
                  w[i][j]=x[i][j];
                }

        for (i=1;i<=n;i++)
          {
                c[i]=0.0;
```

```
            r[i]=0.0;
            p[i][1]=0.0;
            p[i][2]=0.0;
            a[i][1]=0.0;
            a[i][2]=0.0;
      }

    for (i=1;i<=2*n;i++)
      {
            z[i][1]=0.0;
            z[i][2]=0.0;
      }

    for (i=1;i<=n;i++)
      {
            m1=9999.0;
            for (j=1;j<=n;j++)
              if (x[i][j]<m1)
                    m1=x[i][j];
            for (j=1;j<=n;j++)
              x[i][j]=x[i][j]-m1;
      }

    for (j=1;j<=n;j++)
      {
            m1=9999.0;
            for (i=1;i<=n;i++)
              if (x[i][j]<m1)
                    m1=x[i][j];
            for (i=1;i<=n;i++)
              x[i][j]=x[i][j]-m1;
      }

    l=1;
    for (i=1;i<=n;i++)
      {
            j=1;
mot:  if (j>n)
              continue;
        if (x[i][j]!=0)
            {
                  j=j+1;
                  goto mot;
            }
          else
            if (i==1)
                  {
```

```
                        a[l][1]=i;
                        a[l][2]=j;
                        c[j]=1.0;
                        l=l+1;
                       }
                else
                       {
                        l1=l-1;
                        for (k=1;k<=l1;k++)
                               {
                                if (a[k][2]!=j)
                                        continue;
                                else
                                        {
                                         j=j+1;
                                         goto mot;
                                        }
                               }
                       }
          }

         l=l-1;
         if (l!=n)
           {
                 m=1;
hai:    for (i=1;i<=n;i++)
                  {
                       j=1;
ba:       if (j>n)
                         continue;
                       else
                         if ((x[i][j]!=0)||(c[j]!=0)||(r[i]!=0))
                               {
                                j=j+1;
                                goto ba;
                               }
                         else
                               {
                                p[m][1]=i;
                                p[m][2]=j;
                                m=m+1;
                                for (k=1;k<=l;k++)
                                        if (a[k][1]!=i)
                                          continue;
                                        else
                                          {
                                                r[i]=1.0;
```

```
                                        c[a[k][2]]=0.0;
                                        goto sau;
                                    }
                                }
                    }
                k2=m-1;
                r1=p[k2][1];
                c1=p[k2][2];
                k3=l;
                k=1;
                s=1;
bon:    if (k==1)
                    {
                        z[k][1]=r1;
                        z[k][2]=c1;
                        k=k+1;
                        goto bon;
                    }
                else
                    {
                        if (s==1)
                          {
                            for (j=1;j<=k3;j++)
                              if (a[j][2]==c1)
                                    {
                                      r1=a[j][1];
                                      s=2;
                                      z[k][1]=r1;
                                      z[k][2]=c1;
                                      k=k+1;
                                      goto bon;
                                    }
                            k=k-1;
                          }
                        else
                          {
                            for (j=1;j<=k2;j++)
                              if (p[j][1]==r1)
                                    {
                                      c1=p[j][2];
                                      s=1;
                                      z[k][1]=r1;
                                      z[k][2]=c1;
                                      k=k+1;
                                      goto bon;
                                    }
                              else
```

```
                                continue;
                        k=k-1;
                    }
                }
               k5=1;
nam:    if (k5==k)
                {
                    l=l+1;
                    a[l][1]=z[k][1];
                    a[l][2]=z[k][2];
                    if (l!=n)
                     {
                        for (i=1;i<=n;i++)
                          {
                            r[i]=0.0;
                            c[i]=0.0;
                            p[i][1]=0;
                            p[i][2]=0;
                          }
                        for (i=1;i<=l;i++)
                          c[a[i][2]]=1.0;
                        m=1;
                        goto hai;
sau:            m1=9999;
                        for (i=1;i<=n;i++)
                          if (r[i]==0.0)
                            for (j=1;j<=n;j++)
                              if (c[j]==0.0)
                                if (x[i][j]<m1)
                                  m1=x[i][j];
                        for (i=1;i<=n;i++)
                          for (j=1;j<=n;j++)
                            {
                              if ((r[i]!=0.0)||(c[j]!=0.0))
                                if ((r[i]!=1.0)||(c[j]!=1.0))
                                  continue;
                                else
                                  x[i][j]=x[i][j]+m1;
                              else
                                x[i][j]=x[i][j]-m1;
                            }
                        goto hai;
                    }
                }
               else
                {
                    for (i=1;i<=l;i++)
```

```
                        if ((a[i][1]==z[k5+1][1]))
                                if ((a[i][2]==z[k5+1][2]))
                                  break;
                        a[i][1]=z[k5][1];
                        a[i][2]=z[k5][2];
                        k5=k5+2;
                        goto nam;
                }
          }
        q=0.0;
        for (i=1;i<=n;i++)
          q+=w[a[i][1]][a[i][2]];
        printf("Gia thanh cuc tieu : %10.5f\n",q);
        printf("\n");
        printf("Cuc tieu hoa\n");
        for (i=1;i<=n;i++)
          printf("%d%10c%d\n",a[i][1],' ',a[i][2]);
        getch();
}
```

Chạy chương trình ta nhận được giá thành cực tiểu là 181