Ch¬ng 4

Ng¨n xÕp, hµng ®îi vµ danh s¸ch mãc nèi

(stack, queue, link list)

4.1- KiÓu d÷ liÖu ng¨n xÕp vµ øng dông

4.1.1- §Þnh nghÜa vµ khai b¸o

Ng¨n xÕp (Stack) hay bé xÕp chång lµ mét kiÓu danh s¸ch tuyÕn tÝnh ®Æc biÖt mµ phÐp bæ xung phÇn tö vµ lo¹i bá phÇn tö lu«n lu«n ®îc thùc hiÖn ë mét ®Çu gäi lµ ®Ønh (top).

Cã thÓ h×nh dung stack nh mét chång ®Üa ®îc xÕp vµo hép hoÆc mét b¨ng ®¹n ®îc n¹p vµo khÈu sóng liªn thanh. Qu¸ tr×nh xÕp ®Üa hoÆc n¹p ®¹n chØ ®îc thùc hiÖn ë mét ®Çu, chiÕc ®Üa hoÆc viªn ®¹n cuèi cïng l¹i chiÕm vÞ trÝ ë ®Ønh ®Çu tiªn cßn ®Üa ®Çu hoÆc viªn ®¹n ®Çu l¹i ë ®¸y cña hép (bottom), khi lÊy ra th× ®Üa cuèi cïng hoÆc viªn ®¹n cuèi cïng l¹i ®îc lÊy ra tríc tiªn. Nguyªn t¾c vµo sau ra tríc cña stack cßn ®îc gäi díi mét tªn kh¸c LIFO (Last- in- First- Out).

Stack cã thÓ rçng hoÆc bao gåm mét sè phÇn tö. Cã hai thao t¸c chÝnh cho stack lµ thªm mét nót vµo ®Ønh stack (push) vµ lo¹i bá mét nót t¹i ®Ønh stack (pop). NÕu ta muèn thªm mét nót vµo ®Ønh stack th× tríc ®ã ta ph¶i kiÓm tra xem stack ®· ®Çy (full) hay cha, nÕu ta muèn lo¹i bá mét nót cña stack th× ta ph¶i kiÓm tra stack cã rçng hay kh«ng. H×nh 4.1 minh häa sù thay ®æi cña stack th«ng qua c¸c thao t¸c thªm vµ bít ®Ønh trong stack.

Gi¶ sö ta cã mét stack S lu tr÷ c¸c kÝ tù. Tr¹ng th¸i b¾t ®Çu cña stack ®îc m« t¶ trong h×nh a. Khi ®ã thao t¸c:

push(S,'G') (h×nh b)

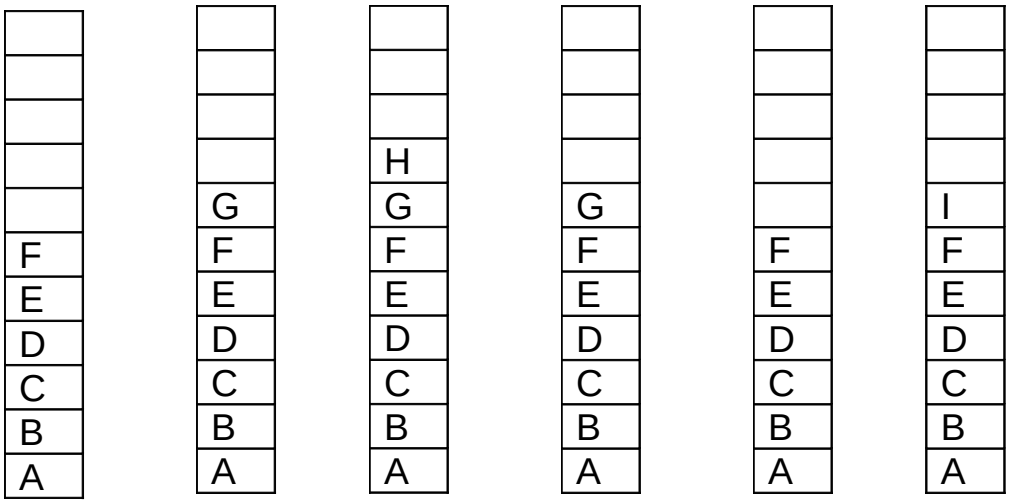
push(S,'H') (h×nh c)

pop(S) (h×nh d)

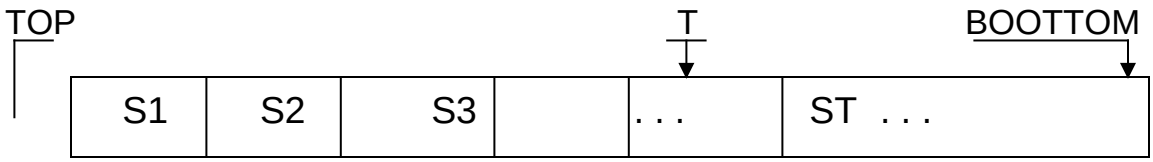
pop(S) (h×nh e)

push(S,'I') (h×nh f)

(h×nh a) (h×nh b) (h×nh c) (h×nh d) (h×nh e) (h×nh f)



Cã thÓ lu tr÷ stack díi d¹ng mét vector S gåm n thµnh phÇn li¹n tiÕp nhau. NÕu T lµ lµ ®Þa chØ cña phÇn tö ®Ønh stack th× T sÏ cã gi¸ trÞ biÕn ®æi khi stack ho¹t ®éng. Ta gäi phÇn tö ®Çu ti¹n cña stack lµ phÇn tö thø 0, nh vËy stack rçng khi T cã gi¸ trÞ nhá h¬n 0 ta qui íc lµ -1. Stack trµn khi T cã gi¸ trÞ lµ n-1. Mçi khi mét phÇn tö ®îc th¹m vµo stack, gi¸ trÞ cña T ®îc t¨ng l¹n 1 ®¬n vÞ, khi mét phÇn tö bÞ lo¹i bá khái stack gi¸ trÞ cña T sÏ gi¶m ®i mét ®¬n vÞ.



→

§Ó khai b¸o mét stack, chóng ta cã thÓ dïng mét m¶ng mét chiÒu. PhÇn tö thø 0 lµ ®¸y stack, phÇn tö cuèi cña m¶ng lµ ®Ønh stack. Mét stack tæng qu¸t lµ mét cÊu tróc gåm hai trêng, trêng top lµ mét sè nguyªn chØ ®Ønh stack. Trêng node: lµ mét m¶ng mét chiÒu gåm MAX phÇn tö trong ®ã mçi phÇn tö lµ mét nót cña stack. Mét nót cña stack cã thÓ lµ mét biÕn ®¬n hoÆc mét cÊu tróc ph¶n ¸nh tËp th«ng tin vÒ nót hiÖn t¹i. VÝ dô, khai b¸o stack dïng ®Ó lu tr÷ c¸c sè nguyªn.

```
#define     TRUE1
#define     FALSE     0
#define     MAX       100
typedef struct     {
        int     top;
        int     nodes[MAX];
} stack;
```

4.1.2- C¸c thao t¸c víi stack

Trong khi khai b¸o mét stack dïng danh s¸ch tuyÕn tÝnh, chóng ta cÇn ®Þnh nghÜa MAX ®ñ lín ®Ó cã thÓ lu tr÷ ®îc mäi ®Ønh cña stack. Mét stack ®· bÞ trµn (TOP = MAX- 1) th× nã kh«ng thÓ thªm vµo phÇn tö trong stack, mét stack rçng th× nã kh«ng thÓ ®a ra phÇn tö. V× vËy, chóng ta cÇn x©y dùng thªm c¸c thao t¸c kiÓm tra stack cã bÞ trµn hay kh«ng (full) vµ thao t¸c kiÓm tra stack cã rçng hay kh«ng (empty).

Thao t¸c Empty: KiÓm tra stack cã rçng hay kh«ng:

```
int     Empty(stack *ps) {
        if (ps ->top == -1)// liÖu ®©y cã ph¶I lµ danh s¸ch liªn kÕt kh«ng. NÕu kh«ng ph¶I th× c¸I nµy nã cã ý nghÜa g× ®©y?
                return(TRUE);
        return(FALSE);
}
```

Thao t¸c Push: Thªm nót míi x vµo ®Ønh stack vµ thay ®æi ®Ønh stack.

```
void    Push (stack *ps, int x) {
```

```
        if ( ps ->top == -1) {
                printf("\n stack full");
                return;
        }
        ps -> top = ps ->top + 1;
        ps -> nodes[ps->top] = x;
}
```

Thao t¸c Pop : Lo¹i bá nót t¹i ®Ønh stack.

```
int     Pop ( stack *ps) {
        if (Empty(ps) {
                printf("\n stack empty");
                return(0);
        }
        return( ps -> nodes[ps->top --]);
}
```

4.1.3- øng dông cña stack

VÝ dô 4.1. Ch¬ng tr×nh ®¶o ngîc x©u kÝ tù: qu¸ tr×nh ®¶o ngîc mét x©u kÝ tù gièng nh viÖc ®a vµo (push) tõng kÝ tù trong x©u vµo stack, sau ®ã ®-a ra (pop) c¸c kÝ tù trong stack ra cho tíi khi stack rçng ta ®îc mét x©u ®¶o ngîc. Ch¬ng tr×nh sau sÏ minh häa c¬ chÕ LIFO ®¶o ngîc x©u kÝ tù sö dông stack.

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <conio.h>
#include    <dos.h>
#include    <string.h>
#define     MAX  100
#define     TRUE1
#define     FALSE       0
typedef     struct{
```

```
        int top;
        char node[MAX];
} stack;

/* nguyen mau cua ham*/
int     Empty(stack *);
void Push(stack *, char);
char  Pop(stack *);
/* Mo ta ham */
int Empty(stack *ps){
        if (ps->top==-1)
                return(TRUE);
        return(FALSE);
}
void Push(stack *ps, char x){
        if (ps->top==MAX-1 ){
                printf("\n Stack full");
                delay(2000);
                return;
        }
        (ps->top)= (ps->top) + 1;
        ps->node[ps->top]=x;
}
char Pop(stack *ps){
        if (Empty(ps)){
                printf("\n Stack empty");
                delay(2000);return(0);
        }
        return( ps ->node[ps->top--]);
}
void main(void){
```

```
        stack s;
        char c, chuoi[MAX];
        int i, vitri,n;s.top=-1;clrscr();
        printf("\n Nhap String:");gets(chuoi);
        vitri=strlen(chuoi);
        for (i=0; i<vitri;i++)
                Push(&s, chuoi[i]);
        while(!Empty(&s))
                printf("%c", Pop(&s));
        getch();
}
```

VÝ dô 4.2: ChuyÓn ®æi sè tõ hÖ thËp ph©n sang hÖ c¬ sè bÊt kú.

§Ó chuyÓn ®æi mét sè ë hÖ thËp ph©n thµnh sè ë hÖ c¬ sè bÊt kú, chóng ta lÊy sè ®ã chia cho c¬ sè cÇn chuyÓn ®æi, lu tr÷ l¹i phÇn d cña phÐp chia, sau ®ã ®¶o ngîc l¹i d·y c¸c sè d ta nhËn ®îc sè cÇn chuyÓn ®æi, viÖc lµm nµy gièng nh c¬ chÕ LIFO cña stack.

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>
#define     MAX  100
#define     TRUE1
#define     FALSE     0
typedef     struct{
        int top;
        unsigned int node[MAX];
} stack;
int     Empty(stack *);
void Push( stack *, int);
int Pop(stack *);
int Empty(stack *ps) {
        if (ps->top==-1){
```

```
                printf("\n Stack empty");
                delay(2000);return(TRUE);
        }
        return(FALSE);
}
void Push(stack *ps, int p){
        if (ps ->top==MAX-1){
                printf("\n Stack full");
                delay(2000);return;
        }
        ps->top=(ps->top) + 1;
        ps->node[ps->top]=p;
}
int Pop(stack *ps ){
        if (Empty(ps)){
                printf("\n Stack Empty");
                delay(2000); return(0);
        }
        return(ps->node[ps->top--]);
}
void main(void){
        int n, coso, sodu;
        stack s;s.top=-1;
        clrscr();
        printf("\n Nhap mot so n=");scanf("%d",&n);
        printf("\n Co so can chuyen:");scanf("%d",&coso);
        while(n!=0){
                sodu= n % coso;
                Push(&s,sodu);
                n=n/coso;
        }
```

```
        while(!Empty(&s))
                printf("%X", Pop(&s));
        getch();
}
```

VÝ dô 4.3- TÝnh gi¸ trÞ mét biÓu thøc d¹ng hËu tè.

XĐt mét biÓu thøc d¹ng hËu tè chØ chøa c¸c phĐp to¸n céng (+), trõ (-), nh©n (*), chia (/), lòy thõa ($). CÇn ph¶i nh¾c l¹i r»ng, nhµ logic häc Lewinski ®· chøng minh ®îc r»ng, mäi biÓu thøc ®Òu cã thÓ biÓu diÔn díi d¹ng hËu tè mµ kh«ng cÇn dïng thªm c¸c kÝ hiÖu phô. VÝ dô :

23+5*2$ = ((2 + 3) *5) ^{2} = 625

§Ó tÝnh gi¸ trÞ cña biÓu thøc d¹ng hËu tè, chóng ta sö dông mét stack lu tr÷ biÓu thøc qu¸ tr×nh tÝnh to¸n ®îc thùc hiÖn nh sau:

LÊy to¸n h¹ng 1 (2) -> LÊy to¸n h¹ng 2 (3) -> LÊy phĐp to¸n ‘+’ -> LÊy to¸n h¹ng 1 céng to¸n h¹ng 2 vµ ®Èy vµo stack (5) -> LÊy to¸n h¹ng tiÕp theo (5), lÊy phĐp to¸n tiÕp theo (*), nh©n víi to¸n h¹ng 1 råi ®Èy vµo stack (25), lÊy to¸n h¹ng tiÕp theo (2), lÊy phĐp to¸n tiÕp theo ($) vµ thùc hiÖn, lÊy luü thõa råi ®Èy vµo stack. Cuèi cïng ta nhËn ®îc 25 ^{2}= 625. Ch¬ng tr×nh tÝnh gi¸ trÞ biÓu thøc d¹ng hËu tè ®îc thùc hiÖn nh sau:

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <conio.h>
#include    <dos.h>
#include    <string.h>
#include    <math.h>
#define     MAX  100
#define     TRUE1
#define     FALSE     0
typedef     struct{
        int top;
        double node[MAX];
```

```
} stack;
int      Empty(stack *);
void Push( stack *, double);
double Pop(stack *);
double Dinhtri(char *);
int lakyso(char);
double tinh(int, double, double);
int Empty(stack *ps) {
        if (ps->top==-1){
                printf("\n Stack empty");
                delay(2000);return(TRUE);
        }
        return(FALSE);
}
void Push(stack *ps, double p){
        if (ps ->top==MAX-1){
                printf("\n Stack full");
                delay(2000);return;
        }
        ps->top=(ps->top) + 1;
        ps->node[ps->top]=p;
}
double Pop(stack *ps ){
        if (Empty(ps)){
                printf("\n Stack Empty");
                delay(2000); return(0);
        }
        return(ps->node[ps->top--]);
}
double Dinhtri(char *Bieuthuc){
        int i,c, vitri;
```

```
        double toanhang1, toanhang2, giatri;
        stack s;
        s.top=-1;vitri=strlen(Bieuthuc);
        for(i=0;i<vitri;i++){
                if (lakyso(Bieuthuc[i]))
                        Push(&s,(double)(Bieuthuc[i]-'0'));
                else {
                        toanhang2=Pop(&s);
                        toanhang1=Pop(&s);
                        giatri=tinh(Bieuthuc[i],toanhang1, toanhang2);
                        Push(&s, giatri);
                }
        }
        return(Pop(&s));
}
int lakyso(char kitu) {
        return(kitu>='0' && kitu<='9');
}
double tinh(int toantu, double toanhang1, double toanhang2){
        double ketqua=0;
        switch(toantu){
                case '+': ketqua=toanhang1+toanhang2;break;
                case '-': ketqua=toanhang1-toanhang2;break;
                case '*': ketqua=toanhang1*toanhang2;break;
                case '/': ketqua=toanhang1/toanhang2;break;
                case '$': ketqua=pow(toanhang1,toanhang2);break;
        }
        return(ketqua);
}
void main(void){
        char c, bieuthuc[MAX];
```

```
    int vitri;clrscr();
    printf("\n Nhap mot bieu thuc:");gets(bieuthuc);
    printf("\n Gia tri = %f",Dinhtri(bieuthuc));
}
```

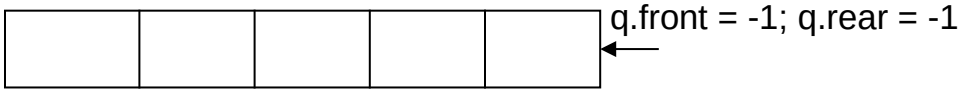
4.2- Hµng ®îi (Queue)

4.2.1- Giíi thiÖu hµng ®îi

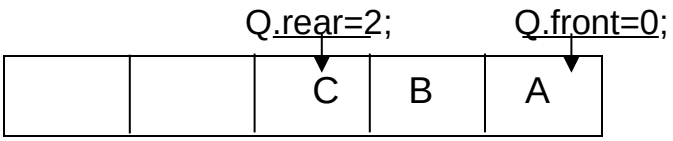
Kh¸c víi stack, hµng ®îi (queue) lµ mét danh s¸ch tuyÕn tÝnh mµ thao t¸c bæ sung phÇn tö ®îc thùc hiÖn ë mét ®Çu gäi lµ lèi vµo (rear). PhÐp lo¹i bá phÇn tö ®îc thùc hiÖn ë mét ®Çu kh¸c gäi lµ lèi ra (front). Nh vËy, c¬ chÕ cña queue gièng nh mét hµng ®îi, ®i vµo ë mét ®Çu vµ ®i ra ë mét ®Çu hay FIFO (First- In- First- Out).

§Ó truy nhËp vµo hµng ®îi, chóng ta sö dông hai biÕn con trá front chØ lèi tríc vµ rear chØ lèi sau. Khi lèi tríc trïng víi lèi sau (q.rear = q.front) th× queue ë tr¹ng th¸i rçng (h×nh a), ®Ó thªm d÷ liÖu vµo hµng ®îi c¸c phÇn tö A, B, C ®îc thùc hiÖn th«ng qua thao t¸c insert(q,A), insert(q,B), insert(q,C) ®îc m« t¶ ë h×nh b, thao t¸c lo¹i bá phÇn tö khái hµng ®îi ®îc m« t¶ ë h×nh c, nh÷ng thao t¸c tiÕp theo ®îc m« t¶ t¹i h×nh d, e, f.

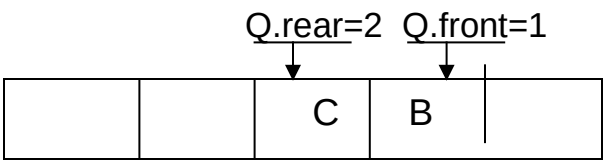
Tr¹ng th¸i rçng cña queue (h×nh a)



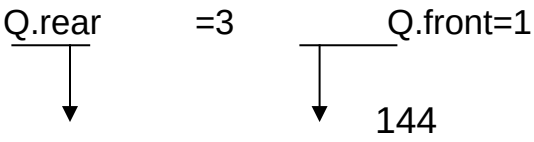
insert(Q, A); insert(Q,B), insert(Q,C) : h×nh b



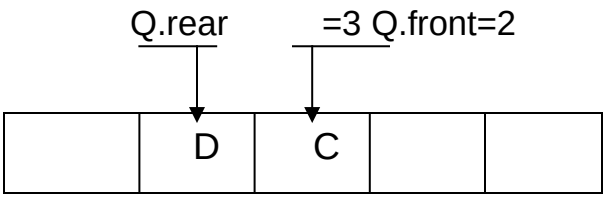
remove(Q): h×nh c



insert(Q,D): h×nh d

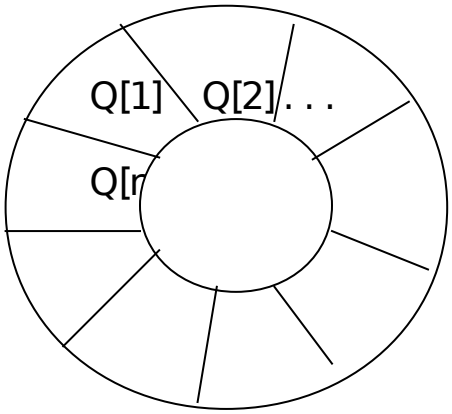


remove(Q): h×nh e



C¸ch tæ chøc nµy sÏ dÉn tíi trêng hîp c¸c phÇn tö di chuyÓn kh¾p kh«ng gian nhí khi thùc hiÖn bæ sung vµ lo¹i bá. VÝ dô: cø mçi phÐp bæ sung kÌm theo mét phÐp lo¹i bá sÏ dÉn tíi trêng hîp

Q.front = Q.rear = MAXQUE-1; vµ phÐp bæ xung vµ lo¹i bá kh«ng thÓ tiÕp tôc thùc hiÖn. §Ó kh¾c phôc t×nh tr¹ng nµy, chóng ta cã thÓ tæ chøc queue nh mét vßng trßn, khi ®ã Q[1] coi nh ®øng sau Q[MAXQUE-1].



Trong nhiÒu trêng hîp, khi thùc hiÖn thªm hoÆc lo¹i bá phÇn tö cña hµng ®îi chóng ta cÇn xÐt tíi mét thø tù u tiªn nµo ®ã, khi ®ã hµng ®îi ®îc gäi lµ hµng ®îi cã ®é u tiªn (Priority Queue). Víi priority queue, th× nót nµo cã ®é u tiªn cao nhÊt ®îc thùc hiÖn lo¹i bá tríc nhÊt, cßn víi thao t¸c thªm phÇn tö vµo hµng ®îi trë thµnh thao t¸c thªm phÇn tö vµo hµng ®îi cã xÐt tíi ®é u tiªn.

4.2.2- øng dông hµng ®îi

Mäi vÊn ®Ò cña thùc tÕ liªn quan tíi c¬ chÕ FIFO nh c¬ chÕ göi tiÒn, rót tiÒn trong ng©n hµng, ®Æt vÐ m¸y bay . . .®Òu cã thÓ øng dông ®îc b»ng hµng ®îi. Hµng ®îi cßn cã nh÷ng øng dông trong viÖc gi¶i quyÕt c¸c

bµi to¸n cña HÖ ®iÒu hµnh vµ ch¬ng tr×nh dÞch nh bµi to¸n ®iÒu khiÓn c¸c qu¸ tr×nh, ®iÒu khiÓn n¹p ch¬ng tr×nh vµo bé nhí hay bµi to¸n lËp lÞch. Sau ®©y lµ nh÷ng vÝ dô minh häa vÒ øng dông cña hµng ®îi.

VÝ dô 4.4- Gi¶i quyÕt bµi to¸n "N***gêi s¶n xuÊt vµ nhµ tiªu dïng*** " víi sè c¸c vïng ®Öm h¹n chÕ.

Chóng ta h·y m« t¶ qu¸ tr×nh s¶n xuÊt vµ tiªu dïng nh hai qu¸ tr×nh riªng biÖt vµ thùc hiÖn song hµnh, ngêi s¶n xuÊt cã thÓ s¶n xuÊt tèi ®a n mÆt hµng, ngêi tiªu dïng còng chØ ®îc phÐp sö dông trong sè n mÆt hµng. Tuy nhiªn, ngêi s¶n xuÊt khi s¶n xuÊt mét mÆt hµng anh ta chØ cã thÓ lu tr÷ vµo kho khi vµ chØ khi kho cha bÞ ®Çy, ®ång thêi khi ®ã, nÕu kho hµng kh«ng rçng (kho cã hµng) ngêi tiªu dïng cã thÓ tiªu dïng nh÷ng mÆt hµng trong kho theo nguyªn t¾c hµng nµo nhËp vµo kho tríc ®îc tiªu dïng tríc gièng nh c¬ chÕ FIFO cña queue. Sau ®©y lµ nh÷ng thao t¸c chñ yÕu trªn hµng ®îi ®Ó gi¶i quyÕt bµi to¸n:

§Þnh nghÜa hµng ®îi nh mét danh s¸ch tuyÕn tÝnh gåm MAX phÇn tö mçi phÇn tö lµ mét cÊu tróc, hai biÕn front, rear trá lèi vµo vµ lèi ra trong queue:

```
typedef       struct{
        int mahang;
        char ten[20];
} hang;
typedef struct {
        int front, rear;
        hang node[MAX];
} queue;
```

Thao t¸c Initialize: thiÕt lËp tr¹ng th¸i ban ®Çu cña hµng ®îi. ë tr¹ng th¸i nµy, font vµ rear cã cïng mét gi¸ trÞ :MAX-1.

```
void Initialize ( queue *pq){
        pq->front = pq->rear = MAX -1;
}
```

Thao t¸c Empty: kiÓm tra hµng ®îi cã ë tr¹ng th¸i rçng hay kh«ng. Hµng ®îi rçng khi front == rear.

```
int Empty(queue *pq){
        if (pq->front==pq->rear)
                return(TRUE);
        return(FALSE);
}
```

Thao t¸c Insert: thªm X vµo hµng ®îi Q. NÕu viÖc thªm X vµo hµng ®îi ®îc thùc hiÖn ë ®Çu hµng th× rear cã gi¸ trÞ 0, nÕu rear kh«ng ph¶i ë ®Çu hµng ®îi th× gi¸ trÞ cña nã ®îc t¨ng lªn 1 ®¬n vÞ.

```
void Insert(queue *pq, hang x){
        if (pq->rear==MAX-1 )
                pq->rear=0;
        else
                (pq->rear)++;
        if (pq->rear ==pq->front){
                printf("\n Queue full");
                delay(2000);return;
        }
        else
                pq->node[pq->rear]=x;
}
```

Thao t¸c Remove: lo¹i bá phÇn tö ë vÞ trÝ front khái hµng ®îi. NÕu hµng ®îi ë tr¹ng th¸i rçng th× thao t¸c Remove kh«ng thÓ thùc hiÖn ®îc, trong trêng hîp kh¸c front ®îc t¨ng lªn mét ®¬n vÞ.

```
hang  Remove(queue *pq){
        if (Empty(pq)){
                printf("\n Queue Empty");
                delay(2000);
        }
        else {
                if (pq->front ==MAX-1)
```

```
                pq->front=0;
            else
                pq->front++;
        }
        return(pq->node[pq->front]);
    }
```

Thao t¸c Traver: DuyÖt tÊt c¶ c¸c nót trong hµng ®îi.

```
void Traver( queue *pq){
    int i;
    if(Empty(pq)){
        printf("\n Queue Empty");
        return;
    }
    if (pq->front ==MAX-1)
        i=0;
    else
        i = pq->front+1;
    while (i!=pq->rear){
        printf("\n %11d % 15s", pq->node[i].mahang, pq->node[i].ten);
        if(i==MAX-1)
            i=0;
        else
            i++;
    }
    printf("\n %11d % 15s", pq->node[i].mahang, pq->node[i].ten);
}
```

Sau ®©y lµ toµn bé v¨n b¶n ch¬ng tr×nh:

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

```
#include      <dos.h>
#include      <string.h>
#include      <math.h>
#define       MAX  50
#define       TRUE1
#define       FALSE       0
typedef       struct{
        int mahang;
        char ten[20];
} hang;
typedef struct {
        int front, rear;
        hang node[MAX];
} queue;
/* nguyen mau cua ham*/
void   Initialize( queue *pq);
int     Empty(queue *);
void Insert(queue *, hang x);
hang  Remove(queue *);
void   Traver(queue *);

/* Mo ta ham */
void Initialize ( queue *pq){
        pq->front = pq->rear = MAX -1;
}
int Empty(queue *pq){
        if (pq->front==pq->rear)
                return(TRUE);
        return(FALSE);
}
void Insert(queue *pq, hang x){
```

```
        if (pq->rear==MAX-1 )
                pq->rear=0;
        else
                (pq->rear)++;
        if (pq->rear ==pq->front){
                printf("\n Queue full");
                delay(2000);return;
        }
        else
                pq->node[pq->rear]=x;
}
hang  Remove(queue *pq){
        if (Empty(pq)){
                printf("\n Queue Empty");
                delay(2000);
        }
        else {
                if (pq->front ==MAX-1)
                        pq->front=0;
                else
                        pq->front++;
        }
        return(pq->node[pq->front]);
}
void Traver( queue *pq){
        int i;
        if(Empty(pq)){
                printf("\n Queue Empty");
                return;
        }
        if (pq->front ==MAX-1)
```

```
            i=0;
        else
            i = pq->front+1;
        while (i!=pq->rear){
            printf("\n %11d % 15s", pq->node[i].mahang, pq->node[i].ten);
            if(i==MAX-1)
                i=0;
            else
                i++;
        }
        printf("\n %11d % 15s", pq->node[i].mahang, pq->node[i].ten);
}

void main(void){
        queue q;
        char chucnang, front1; char c; hang mh;
        clrscr();
        Initialize(&q);
        do {
            clrscr();
            printf("\n NGUOI SAN XUAT/ NHA TIEU DUNG");
            printf("\n 1- Nhap mot mat hang");
            printf("\n 2- Xuat mot mat hang");
            printf("\n 3- Xem mot mat hang");
            printf("\n 4- Xem hang moi nhap");
            printf("\n 5- Xem tat ca");
            printf("\n 6- Xuat toan bo");
            printf("\n Chuc nang chon:");chucnang=getch();
            switch(chucnang){
                case ‘1’:
```

```
                printf("\n Ma mat hang:"); scanf("%d",
                &mh.mahang);
                printf("\n Ten hang:");scanf("%s", mh.ten);
                Insert(&q,mh);break;
            case '2':
                if (!Empty(&q)){
                    mh=Remove(&q);
                    printf("\n %5d %20s",mh.mahang, mh.ten);
                }
                else {
                    printf("\n Queue Empty");
                    delay(1000);
                }
                break;
            case '3':
                front1=(q.front==MAX-1)?0:q.front+1;
                printf("\n Hang xuat");
                printf("\n %6d %20s",q.node[front1].mahang,
                q.node[front1].ten);
                break;
            case '4':
                printf("\n Hang moi nhap");
                printf("\n %5d %20s",
                q.node[q.rear].mahang,q.node[q.rear].ten);
                break;
            case '5':
                printf("\ Hang trong kho");
                Traverse(&q);delay(2000);break;
        }
    } while(chucnang!='0');
}
```

VÝ dô 4.5: Bµi to¸n lËp lÞch cã u tiª n: Gi¶ sö cã n qu¸ tr×nh thùc hiÖn song hµnh trong hÖ thèng, ë mçi thêi ®iÓm CPU chØ ®¸p øng ®îc cho mét qu¸ tr×nh. H·y lËp lÞch ®Ó CPU ®¸p øng cho mçi qu¸ tr×nh ®ang thùc hiÖn trong hÖ, sao cho qu¸ tr×nh nµo cã ®é u tiª n cao nhÊt ®îc ®¸p øng tríc nhÊt.

§Ó gi¶i quyÕt bµi to¸n, chóng ta tæ chøc c¸c qu¸ tr×nh ®ang ®îi CPU ®¸p øng nh mét hµng ®îi. Mçi phÇn tö cña hµng ®îi lµ mét qu¸ tr×nh vµ cã thÓ ®îc HÖ ®iÒu hµnh qu¶n lý b»ng mét khèi ®iÒu khiÓn c¸c qu¸ tr×nh PCB (Proccess Control Block), mçi PCB ®îc ph¶n ¸nh b»ng nh÷ng th«ng tin sau:

Pointer	Register
Memory Limited	
I/O Driver	
List Open File	
. . . . .	
. . . . .	
Priority:	

§Ó ®¬n gi¶n, chóng ta coi tÊt c¶ th«ng tin ph¶n ¸nh vÒ qu¸ tr×nh nh mét sè nguyª n vµ sè nguyª n ®ã trïng víi ®é u tiª n cña qu¸ tr×nh. Khi ®ã, viÖc thùc hiÖn n¹p qu¸ tr×nh vµo hµng ®îi nh nhËp mét sè nguyª n vµ n¹p vµo hµng ®îi sao cho sè lín h¬n sÏ ®îc n¹p vµo phÇn tö ®Çu tiª n, víi c¸ch lµm nh vËy d·y c¸c qu¸ tr×nh sÏ tù ®éng s¾p xÕp theo thø tù gi¶m dÇn cña ®é u tiª n. Qu¸ tr×nh CPU ®¸p øng gièng nh viÖc lo¹i bá qu¸ tr×nh khái hµng ®îi, qu¸ tr×nh nµo cã ®é u tiª n lín nhÊt sÏ ®îc CPU ®¸p øng sím nhÊt. Sau ®©y lµ ch-¬ng tr×nh gi¶i quyÕt bµi to¸n lËp lÞch ®¬n gi¶n cho CPU:

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

```
#include	<math.h>
#define	MAX  100
#define	TRUE1
#define	FALSE	0
typedef struct {
	int rear;
	int node[MAX];
} pqueue;
void Pqinitialize(pqueue *pq){
	pq->rear=-1;// vi khong xem hang doi giong mot vong tron nhu o truong hop tren nen khi khoi tao rear = -1
}
int Pqempty(pqueue *pq){
	if (pq->rear==-1)// do viec khoi tao rear = -1 nen khi no bang gia tri nay thi queue rong
		return(TRUE);
	return(FALSE);
}
int Pqueuesize(pqueue *pq){
	return(pq->rear+1);?// day la gi
}
void Pqinsert(pqueue *pq, int x){
	int i, j;
	if (pq->rear ==MAX - 1){
		printf("\n Queue full");
		delay(2000); return;
	}
	for (i=0;i<Pqueuesize(pq) && pq->node[i]>=x; i++);
	for(j=Pqueuesize(pq);j>i;j--)
		pq->node[j] = pq->node[j-1];
	pq->node[i]=x;
```

```
        pq->rear++;
}
int Pqremove(pqueue *pq){
        int x, i;
        if (Pqempty(pq)){
                printf("\n Queue full");
                delay(2000); return(x);
        }
        else {
                x=pq->node[0];
                for (i=0;i<pq->rear;i++)
                        pq->node[i]=pq->node[i+1]++;
                pq->rear--;
        }
        return(x);
}
void Pqtraver(pqueue *pq){
        int i;
        if (Pqempty(pq)) {
                printf("\n Queue Empty");
                delay(2000); return;
        }
        for (i=0;i<=pq->rear;i++)
                printf("%5d",pq->node[i]);
}
void main(void){
        pqueue *pq;
        int douutien, chucnang;char c;
        Pqinitialize(pq);
        do {
                clrscr();
```

```
            printf("\n QUEUE PRIORITY");
            printf("\n 1- Them nut uu tien");
            printf("\n 2- Xoa nut uu tien");
            printf("\n 3- Xoa hang doi");
            printf("\n 4- Duyet hang doi");
            printf("\n 0- Ket thuc");
            chucnang=getch();
            switch(chucnang){
                  case '1':
                        printf("\n Do uu tien:");scanf("%d",&douutien);
                        Pqinsert(pq, douutien);
                        break;
                  case '2':
                        Pqremove(pq); break;
                  case '3':
                        Pqinitialize(pq); break;
                  case '4':Pqtraver(pq); delay(2000); break;
            }
      } while(chucnang!='0');
}
```

4.3- Danh s¸ch liªn kÕt ®¬n

4.3.1- Giíi thiÖu vµ ®Þnh nghÜa

Mét danh s¸ch mãc nèi, hoÆc ng¾n gän h¬n, mét danh s¸ch, lµ mét d·y cã thø tù c¸c phÇn tö ®îc gäi lµ ®Ønh. Danh s¸ch cã ®iÓm b¾t ®Çu, gäi lµ tiªu ®Ò hay ®Ønh ®Çu, mét ®iÓm cuèi cïng gäi lµ ®Ønh cuèi. Mäi ®Ønh trong danh s¸ch ®Òu cã cïng kiÓu ngay c¶ khi kiÓu nµy cã nhiÒu d¹ng kh¸c nhau.

B¶n chÊt ®éng lµ mét trong nh÷ng tÝnh chÊt chÝnh cña danh s¸ch mãc nèi. Cã thÓ thªm hoÆc bít ®Ønh trong danh s¸ch vµo mäi lóc, mäi vÞ trÝ. V× sè ®Ønh cña danh s¸ch kh«ng thÓ dù kiÕn tríc ®îc, nªn khi thùc hiÖn, chóng

ta ph¶i dïng con trá mµ kh«ng dïng ®îc m¶ng ®Ó b¶o ®¶m viÖc thùc hiÖn hiÖu qu¶ vµ tin cËy.

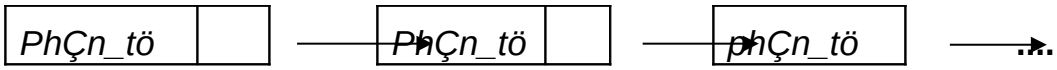
Mçi ®Ønh trong danh s¸ch ®Òu gåm hai phÇn. PhÇn thø nhÊt chøa d÷ liÖu. D÷ liÖu cã thÓ chØ lµ mét biÕn ®¬n hoÆc lµ mét cÊu tróc (hoÆc con trá cÊu tróc) cã kiÓu nµo ®ã. PhÇn thø hai cña ®Ønh lµ mét con trá chØ vµo ®Þa chØ cña ®Ønh tiÕp theo trong danh s¸ch. V× vËy cã thÓ dÔ dµng sö dông c¸c ®Ønh cña danh s¸ch qua mét cÊu tróc tù trá hoÆc ®Ö qui.

Xem nh mét thÝ dô ®¬n gi¶n, ta h·y xÐt trêng hîp mçi ®Ønh cña danh s¸ch chØ lu gi÷ mét biÕn nguyªn. Cã thÓ ®Þnh nghÜa ®Ønh nh sau:

```
/*®Ønh cña danh s¸ch ®¬n chØ chøa mét sè nguyªn*/

struct don {
        int phantu;
        struct don *tiep;
};
typedef struct don don_t;
```

Trong trêng hîp nµy, biÕn nguyªn phantu cña tõng ®Ønh chøa d÷ liÖu cßn biÕn con trá tiep chøa ®Þa chØ cña ®Ønh tiÕp theo. S¬ ®å biÓu diÔn danh s¸ch mãc nèi ®¬n ®îc biÓu diÔn nh h×nh díi ®©y



H×nh 4.3.1. Danh s¸ch mãc nèi ®¬n

Tæng qu¸t h¬n, mçi ®Ønh cña danh s¸ch cã thÓ chøa nhiÒu phÇn tö d÷ liÖu. Trong trêng hîp nµy, hîp lý h¬n c¶ lµ ®Þnh nghÜa mét kiÓu cÊu tróc t¬ng øng víi d÷ liÖu cÇn lu gi÷ t¹i mçi ®Ønh. Ph¬ng ph¸p nµy ®îc sö dông trong ®Þnh nghÜa kiÓu sau ®©y:

```
/*®Ønh cña danh s¸ch tæng qu¸t */

struct tq {
        thtin_t phantu;
        struc tq*tiep;
};
typedef struct tq tq_t;
```

KiÓu cÊu tróc ***thtin_t*** ph¶i ®îc ®Þnh nghÜa tríc ®ã ®Ó t¬ng øng víi c¸c d÷ liÖu sÏ ®îc lu tr÷ t¹i tõng ®Ønh. Danh s¸ch ®îc t¹o nªn tõ kiÓu ®Ønh nµy gièng nh ë s¬ ®å trong H×nh 4.3.1, ngo¹i trõ viÖc mçi phantu lµ mét biÕn nguyªn.

4.3.2- C¸c thao t¸c trªn danh s¸ch mãc nèi

Thao t¸c c¸c danh s¸ch mãc nèi bao gåm viÖc cÊp ph¸t bé nhí cho c¸c ®Ønh (th«ng qua c¸c hµm MALLOC hoÆc CALLOC) vµ g¸n d÷ liÖu cho con trá. §Ó danh s¸ch ®îc t¹o nªn ®óng ®¾n, ta biÓu diÔn cho phÇn tö cuèi danh s¸ch lµ mét con trá NULL. Con trá NULL lµ tÝn hiÖu th«ng b¸o kh«ng cßn phÇn tö nµo tiÕp theo trong danh s¸ch n÷a.

TiÖn h¬n c¶ lµ chóng ta ®Þnh nghÜa mét con trá tíi danh s¸ch nh sau:

```
struct  node {
        int infor;
        struct node *next;
};
typedef struct node *NODEPTR; // Con trá tíi node
```

CÊp ph¸t bé nhí cho mét node

```
NODEPTR Getnode(void) {
        NODEPTR p;
        P = (NODEPTR) malloc(sizeof( struct node));
        Return(p);
}
```

Gi¶i phãng bé nhí cña mét node

```
NODEPTR Freenode( NODEPTR p){
        free(p);
}
```

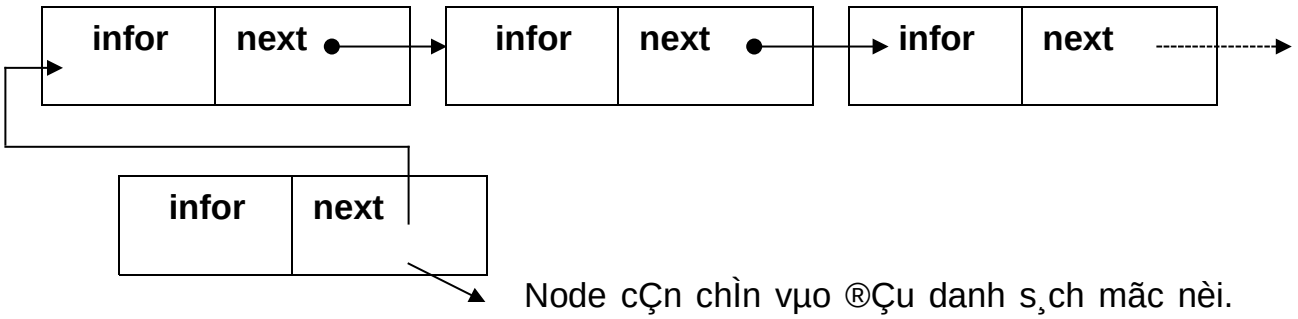
ChÌn mét phÇn tö míi vµo ®Çu danh s¸ch

C¸c bíc ®Ó chÌn mét phÇn tö míi vµo ®Çu danh s¸ch cÇn thùc hiÖn lµ:

- CÊp kh«ng gian bé nhí ®ñ lu gi÷ mét ®Ønh míi;
- G¸n c¸c gi¸ trÞ con trá thÝch hîp cho ®Ønh míi;
- ThiÕt lËp liªn kÕt víi ®Ønh míi.

S¬ ®å biÓu diÔn phÐp thªm mét ®Ønh míi vµo ®Çu danh s¸ch ®îc thÓ hiÖn nh h×nh 4.3.2.

H×nh 4.3.2. Th³m ®Ønh míi vµo ®Çu danh s¸ch mãc nèi ®¬n



Node cÇn chÌn vµo ®Çu danh s¸ch mãc nèi.

```
void Push_Top( NODEPTR *plist, int x) {
        NODEPTR p;
        p= Getnode(); // cÊp kh«ng gian nhí cho ®Ønh míi
        p -> infor = x; // g¸n gi¸ trÞ thÝch hîp cho ®Ønh míi
        p ->next = *plist;//dua con tro ve dau danh sach
        *plist = p; // thiÕt lËp li³n kÕt
}
```

Th³m mét phÇn tö míi vµo cuèi danh s¸ch

§Ó th³m mét node vµo cuèi danh s¸ch, ta cÇn thùc hiÖn qua c¸c bíc sau:

CÊp ph¸t bé nhí cho node míi;

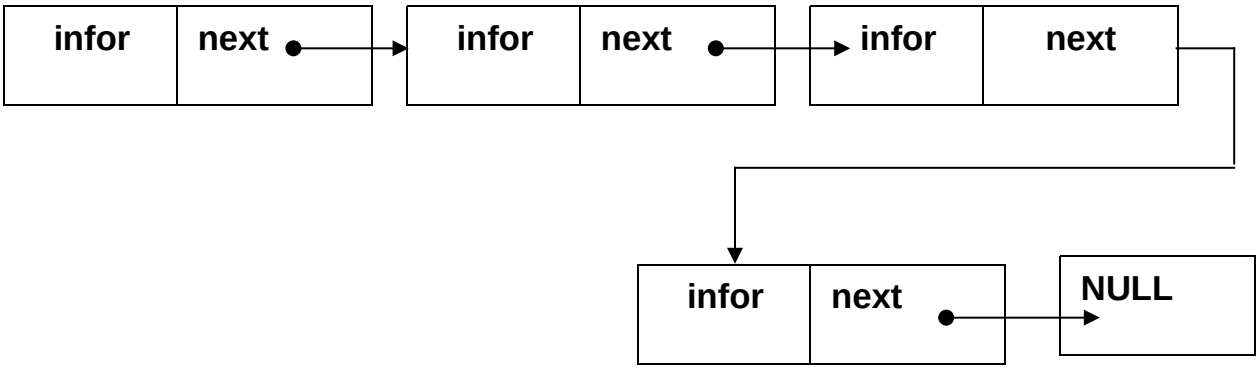
G¸n gi¸ trÞ thÝch hîp cho node míi;

Di chuyÓn con trá tíi phÇn tö cuèi danh s¸ch;

ThiÕt lËp li³n kÕt cho node míi.

S¬ ®å thÓ hi³n phÐp th³m mét phÇn tö míi vµo cuèi danh s¸ch ®îc thÓ hiÖn nh trong h×nh 4.3.3.

H×nh 4.3.3. Thªm node míi vµo cuèi danh s¸ch.




```
void Push_Bottom( NODEPTR *plist, int x) {
        NODEPTR p, q;
        p= Getnode(); // cÊp ph¸t bé nhí cho node míi
        p->infor = x; // g¸n gi¸ trÞ th«ng tin thÝch hîp
        q = *plist; // cho q bat dau di tu diem dau tien cua danh sac
        while (q-> next != NULL)
                q = q -> next;
// sau khi xong vong while thi q se la diem cuoi cung
        // q lµ node cuèi cïng cña danh s¸ch liªn kÕt
        q -> next = p; //node cuèi b©y giê lµ node p;
        p ->next = NULL; // de dam bao p la node cuoi thi  node tiep theo
cua p phai la rong
        }
```

Thªm node míi vµo gi÷a danh s¸ch (tríc node p)

§Ó thªm node q vµo tríc node p, chóng ta cÇn lu ý node p ph¶i cã thùc trong danh s¸ch. Gi¶ sö node p lµ cã thùc, khi ®ã x¶y ra hai t×nh huèng: hoÆc node p lµ node cuèi cïng cña danh s¸ch liªn kÕt tøc p->next =NULL,

hoÆc node p cha ph¶i lµ cuèi cïng hay p->next!=NULL. Trêng hîp thø nhÊt, chóng ta chØ cÇn gäi tíi thao t¸c Push_Bottom(). Trêng hîp thø 2, chóng ta thùc hiÖn theo c¸c bíc nh sau:

CÊp ph¸t bé nhí cho node míi;

G¸n gi¸ trÞ thÝch hîp cho node;

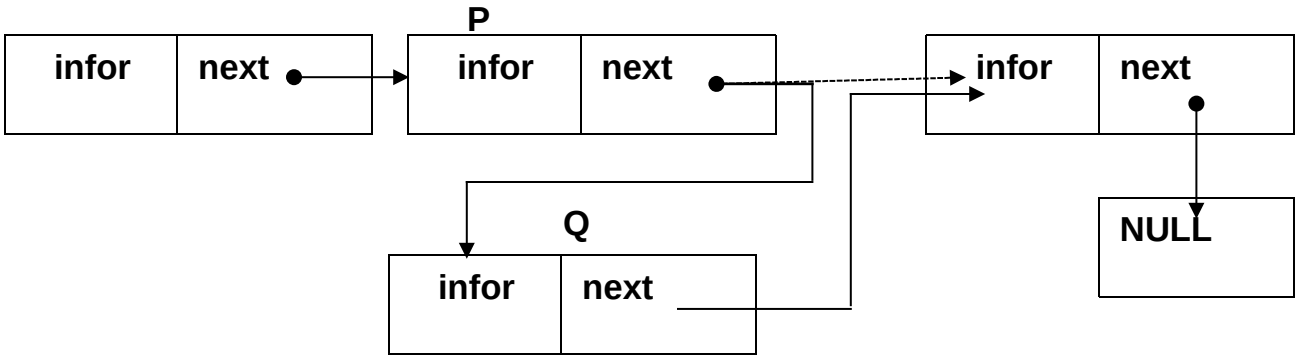
ThiÕt lËp liªn kÕt node q víi node kÕ tiÕp p;

ThiÕt lËp liªn kÕt node node p víi node q;

```
void Push_Before( NODEPTR p, int x ){
        NODEPTR q;
        if (p->next==NULL)
                Push_Bottom(p, x);
        else {
                q= Getnode(); // cÊp ph¸t bé nhí cho node míi
                q -> infor = x; // g¸n gi¸ trÞ th«ng tin thÝch hîp
                q-> next = p-> next; // thiÕt lËp liªn kÕt  node q víi node kÕ tiÕp
p;
                p->next = q; // thiÕt lËp liªn kÕt  node p víi node  q;
        }
}
```

S¬ ®å thªm node vµo gi÷a danh s¸ch ®îc thÓ hiÖn nh sau:

H×nh 4.3.4. PhÐp thªm phÇn tö vµo gi÷a danh s¸ch liªn kÕt ®¬n.



Xo¸ mét node ra khái ®Çu danh s¸ch

Khi lo¹i bá node khái ®Çu danh s¸ch liªn kÕt, chóng ta cÇn chó ý r»ng nÕu danh s¸ch ®ang rçng th× kh«ng cÇn ph¶i lo¹i bá. Trong trêng hîp cßn l¹i, ta thùc hiÖn nh sau:

Dïng node p trá tíi ®Çu danh s¸ch;

DÞch chuyÓn vÞ trÝ ®Çu danh s¸ch tíi node tiÕp theo;

Lo¹i bá liªn kÕt víi p;

Gi¶i phãng node p;

```
void Del_Top( NODEPTR *plist) {
    NODEPTR p;
    p = *plist; // node p trá tíi ®Çu danh s¸ch;
    if (p==NULL) return; // danh s¸ch rçng
    (*plist) = (*plist) -> next; // dÞch chuyÓn node gèc lªn node kÕ tiÕp
    p-> next = NULL; //lo¹i bá liªn kÕt víi p
    Freenode(p); // gi¶i phãng p;
}
```

Lo¹i bá node ë cuèi danh s¸ch

Mét node ë cuèi danh s¸ch cã thÓ x¶y ra ba t×nh huèng sau:

Danh s¸ch rçng: ta kh«ng cÇn thùc hiÖn lo¹i bá;

Danh s¸ch chØ cã ®óng mét node: øng víi trêng hîp lo¹i bá node gèc;

Trêng hîp cßn l¹i danh s¸ch cã nhiÒu h¬n mét node, khi ®ã ta ph¶i dÞch chuyÓn tíi node gÇn node cuèi cïng nhÊt ®Ó thùc hiÖn lo¹i bá.

```
void Del_Bottom(NODEPTR *plist) {
```

```
        NODEPTR p, q;
        if (*plist==NULL) return; //kh«ng lµm g×
        else if ( (*plist)->next==NULL))  // danh s¸ch cã mét node
                Del_Top(plist);
        else {
                p = *plist;
                while (p->next!=NULL){
                        q = p;
                        p = p->next; // q lµ node sau node p;
                }
                // p lµ node cuèi danh s¸ch;
                q->next =NULL; //node cuèi cïng lµ q
                Freenode(p); //gi¶i phãng p;
        }
}
```

Lo¹i bá node ë gi÷a danh s¸ch (tríc node p)

CÇn ®Ó ý r»ng, nÕu tríc node p lµ NULL (p->next==NULL) th× ta kh«ng thùc hiÖn lo¹i bá ®îc. Trêng hîp cßn l¹i chóng ta thùc hiÖn nh sau:

Dïng node q trá tíi node tríc node p;

Lo¹i bá liªn kÕt cña q;

Gi¶i phãng q.

```
void Del_before(NODEPTR p){
        NODEPTR q;
        if (p->next==NULL) return; // kh«ng lµm g×
        q = p ->next;
```

```
        p->next = q->next;
        Freenode(q);
}
```

4.3.4- øng dông cña danh s¸ch liªn kÕt ®¬n

VÝ dô viÕt ch¬ng tr×nh qu¶n lý sinh viªn sau sÏ minh ho¹ ®Çy ®ñ cho c¸c thao t¸c trªn danh s¸ch ®¬n.

VÝ dô 4.6- ViÕt ch¬ng tr×nh qu¶n lý sinh viªn b»ng danh s¸ch mãc nèi ®¬n.

§Ó ®¬n gi¶n, chóng ta chØ qu¶n lý hai thuéc tÝnh m· sinh viªn (masv) vµ hä tªn sinh viªn (hoten), cßn viÖc më réng bµi to¸n coi nh mét bµi tËp thùc hµnh.

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <conio.h>
#include    <dos.h>
#include    <string.h>
#include    <math.h>
#include <alloc.h>
#define TRUE 1
#define FALSE 0
typedef struct {
        int masv;
        char hoten[20];
} sinhvien;
typedef struct node{
        sinhvien infor;
        struct node *next;
} *NODEPTR;
void Initialize(NODEPTR *plist){// khoi tao danh sach
        *plist=NULL;
```

```
}
NODEPTR Getnode(void){// cap phat bo nho cho node
	NODEPTR p;
	p=(NODEPTR) malloc(sizeof(struct node));
	return(p);
}
void Freenode(NODEPTR p){// giai phong node
	free(p);
}
int Emptynode(NODEPTR *plist){// kiem tra danh sach rong
	if(*plist==NULL)
		return(TRUE);
	return(FALSE);
}
NODEPTR Inserttop(NODEPTR *plist, sinhvien x){// them mot node vao dau
danh sach
	NODEPTR p;
	p=Getnode();
	p->infor=x;
	if(Emptynode(plist)){// truong hop la danh sach rong thi node ke tiep
cua node p them vao co gia tri null
		p->next=NULL;
		*plist=p;
		return(p);
	}
	p->next=*plist;// truong hop khong phai la cay rong thi node them vao
dung dau
	*plist=p;// hai cau lenh nay khong duoc hieu ro  nghia lam nhung mang
mang co nghia la lien ket gia node p vao danh sach. Du sao list o day cung la
```

```
gia tri dong chu khong phai la gia tri tinh nen khong nen xem list va list duoi
nhu nhau. Tot nhat phai hieu list la mot con tro chu khong phai la gia tri cua
mang hay struct dua ra
        return(p);
}
int Bottomnode(NODEPTR *plist){// xac dinh I o cuoi danh sach
        int i; NODEPTR p;
        if(Emptynode(plist))
                return(-1);
        p=*plist;i=0;
        while(p!=NULL){
                i=i+1;
                p=p->next;
        }
        return(i);
}
NODEPTR Insertbottom(NODEPTR *plist, sinhvien x){// them  mot node vao
cuoi danh sach
        NODEPTR p, q;int n,i;
        n=Bottomnode(plist);// lay n la diem cuoi cung cua danh sach
        if(n==-1){// neu n = -1 cay rong
                Inserttop(plist,x);// them node cuoi chinh la them node dau
                return(*plist);
        }
        p=*plist;i=0;q=Getnode();q->infor=x;//
// phan trong vong while thuc hien viec day con tro ve cuoi danh sach
        while(i<n-1){
                p=p->next;
                i=i+1;
```

```
        }
        p->next=q;q->next=NULL;// them node q vao cuoi danh sach
        delay(2000);return(q);
}
NODEPTR Insertafter(NODEPTR *plist, sinhvien x, int n){// them vao giua
danh sach
        NODEPTR p,q; int i;
        if(n<0){
                printf("\n Vi tri khong hop le");
                delay(2000);return(NULL);
        }
        p=*plist;i=0;
// doan trong vong while thuc hien viec dua con tro den ngay sau node p va
them node q vao vi tri do
        while(p!=NULL && i<n){
                i=i+1;
                p=p->next;
        }
        if(p==NULL){
                printf("\n Vi tri khong hop le");
                delay(2000); return(NULL);
        }
        q=Getnode();q->infor=x;
        q->next= p->next;// diem phia sau q chinh la diem phia sau p trong day
ban dau
        p->next=q;// diem ngay tiep sau p chinh la diem q them vao
        return(q);
}
void Deltop(NODEPTR *plist){// xoa bo diem dau danh sach
```

```
        NODEPTR p, q;
        p=*plist;
        if(Emptynode(plist)){
                printf("\n Danh sach rong");
                delay(2000); return;
        }
        q=p;p=p->next// gan cho node p la node tiep theo cua no ;*plist=p;
        printf("\n Node bi loai bo");
        printf("\n%-5d%-20s",q->infor.masv, q->infor.hoten);
        delay(2000);Freenode(q);
}
void Delbottom(NODEPTR *plist){// xoabo node cuoi danh sach
        NODEPTR p,q; int i,n;
        n=Bottomnode(plist);
        if(n==-1){
                printf("\n Danh sach rong");
                delay(2000); return;
        }
        if(n==1){
                Deltop(plist);return;// khi n = 1 thi danh sach chi co mot node
        }
        p=*plist;i=0;
        while(i<n-2){
                p=p->next;
                i=i+1;
        }
        q=p->next;p->next=NULL;
        printf("\n Node duoc loai bo");
        printf("\n %-5d%-20s",q->infor.masv,q->infor.hoten);
```

```
        delay(2000); Freenode(q);
}
void Delcurrent(NODEPTR *plist, int n){// xoa node giua
        NODEPTR p,q; int i;
        if(Emptynode(plist)){
                printf("\n Danh sach rong");
                delay(2000);return;
        }
        if(n==0){
                Deltop(plist); return;
        }
        p=*plist; i=0;
        while(p!=NULL && i<n-1){
                i=i+1;
                p=p->next;
        }
        if(p->next==NULL){
                printf("\n Node khong hop le");
                delay(2000); return;
        }
        q=p->next;// q la node tiep theo cua p can duoc xoa bo
        p->next=q->next;// gan node tiep theo cho p chinh la node tiep theo
cua q
        printf("\n Node duoc loai bo");
        printf("\n %-5d%-20s",q->infor.masv, q->infor.hoten);
        delay(2000); Freenode(q);
}
void Travenode(NODEPTR *plist){
        NODEPTR p;
```

```
	if(Emptynode(plist)){
		printf("\n Danh sach rong");
		delay(2000);return;
	}
	p=*plist;
	while(p!=NULL){
		printf("\n %-5d%-20s",p->infor.masv, p->infor.hoten);
		p=p->next;
	}
	delay(2000);
}
void Sortnode(NODEPTR *plist){// danh sach duoc sap xep theo ma sinh vien
	NODEPTR p,q;sinhvien temp;
	for(p=*plist; p!=NULL; p=p->next){
		for(q=p->next; q!=NULL; q=q->next){
			if(p->infor.masv>q->infor.masv){
				temp=p->infor; p->infor=q->infor;
				q->infor=temp;
			}
		}
	}
	printf("\n Danh sach duoc sap xep");
	for(p=*plist// day la chi dau danh sach lien ket; p!=NULL; p=p->next){
		printf("\n %-5d%-20s",p->infor.masv,p->infor.hoten);
	}
	delay(2000);
}
void Searchnode(NODEPTR *plist, int masv){// tim kiem sinh vien theo ma
sinh vien
```

```
	NODEPTR p;
	p=*plist;// bat dau di tu dau danh sach
	while(p!=NULL && p->infor.masv!=masv)// khi tim  gap dung ma sinh
vien thi in sinh vien do ra
		p=p->next;
	if(p==NULL)
		printf("\n Node khong ton tai");
	else {
		printf("\n Node can tim");
		printf("\n %-5d%-20s",p->infor.masv,p->infor.hoten);
	}
	delay(2000);
}

void Thuchien(void){
	NODEPTR plist; sinhvien x,y;int vitri; char c;
	Initialize(&plist);
	do {
		clrscr();
		printf("\n THAO TAC VOI SINGLE LINK LIST");
		printf("\n 1- Them node dau danh sach");
		printf("\n 2- Them node cuoi danh sach");
		printf("\n 3- Them node giua danh sach");
		printf("\n 4- Loai bo node dau danh sach");
		printf("\n 5- Loai bo node cuoi danh sach");
		printf("\n 6- Loai node giua danh sach");
		printf("\n 7- Duyet danh sach");
		printf("\n 8- Sap xep danh sach");
		printf("\n 9- Tim kiem danh sach");
```

```
printf("\n 0- Tro ve");
c=getch();
switch(c){
        case '1':
                printf("\n Ma sinh vien:");scanf("%d", &x.masv);
                fflush(stdin); printf("\n Ho va ten:");gets(x.hoten);
                Inserttop(&plist,x); break;
        case '2':
                printf("\n Ma sinh vien:");scanf("%d", &x.masv);
                fflush(stdin); printf("\n Ho va ten:");gets(x.hoten);
                Insertbottom(&plist,x); break;
        case '3':
                printf("\n Vi tri tren:"); scanf("%d",&vitri);
                printf("\n Ma sinh vien:");scanf("%d", &x.masv);
                fflush(stdin); printf("\n Ho va ten:");gets(x.hoten);
                Insertafter(&plist,x,vitri-1); break;
        case '4': Deltop(&plist);break;
        case '5': Delbottom(&plist);break;
        case '6':
                fflush(stdin);printf("\n Vi tri loai bo:");
                scanf("%d",&vitri);
                Delcurrent(&plist,vitri-1);break;
        case '7': Travenode(&plist); break;
        case '8': Sortnode(&plist);break;
        case '9':
                fflush(stdin);printf("\n Ma sinh vien:");
                scanf("%d",&vitri);
                Searchnode(&plist, vitri);break;
}
```

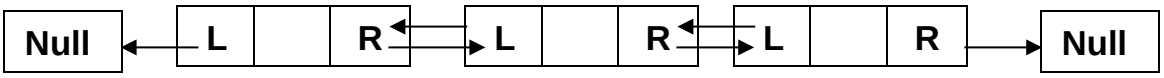
```
    } while(c!='0');
}
void main(void){
    Thuchien();
}
```

4.4- Danh s¸ch liªn kÕt kÐp

Mçi khi thao t¸c trªn danh s¸ch, viÖc duyÖt danh s¸ch theo c¶ hai chiÒu tá ra thuËn tiÖn h¬n cho ngêi sö dông. §«i khi chóng ta ph¶i di chuyÓn trong danh s¸ch tõ node cuèi lªn node ®Çu hoÆc ngîc l¹i b»ng c¸ch ®i qua mét lo¹t c¸c con trá. §iÒu nµy cã thÓ dÔ dµng gi¶i quyÕt ®îc nÕu ta t¨ng th«ng tin chøa t¹i tõng ®Ønh cña danh s¸ch. Ngoµi con trá chøa ®Þa chØ ®Ønh tiÕp theo, *ta thªm con trá tríc ®Ó chøa ®Þa chØ ®øng sau ®Ønh nµy*(cho ®Ønh ®øng tríc hay ®Ønh ®øng sau ®©y – h×nh nh lµ ®Ønh ®øng tríc nã míi ®óng chø). Lµm nh vËy, chóng ta thu ®îc mét cÊu tróc d÷ liÖu míi gäi lµ danh s¸ch liªn kÕt kÐp.

```
struct node {
    int infor;
    struct node *right;// con trá tíi node sau
    struct node *left; // con trá tíi node kÕ tiÕp
};
typedef struct node *NODEPTR; // ®Þnh nghÜa con trá tíi node
```

H×nh 4.4.1 m« t¶ mét danh s¸ch liªn kÕt kÐp.



C¸c thao t¸c trªn danh s¸ch liªn kÕt kÐp còng t¬ng tù nh danh s¸ch liªn kÕt ®¬n. Nhng cÇn chó ý r»ng, mçi node p cña danh s¸ch liªn kÕt kÐp cã hai ®ên liªn kÕt lµ p-> left vµ p->right;

Thao t¸c thªm node míi vµo ®Çu danh s¸ch liªn kÕt kÐp

CÊp ph¸t bé nhí cho node míi;

```
        G¸n gi¸ trÞ thÝch hîp cho node míi;
        ThiÕt lËp liªn kÕt cho node míi;
void Push_Top(NODEPTR *plist, int x){
        NODEPTR p;
        p = Getnode(); //cÊp ph¸t bé nhí cho node
        p ->infor = x; //g¸n gi¸ trÞ thÝch hîp;
        p -> right = *plist; // thiÕt lËp liªn kÕt ph¶i
        (*plist) ->left = p; // thiÕt lËp liªn kÕt víi *plist
        p-> left = NULL;// thiÕt lËp liªn kÕt tr¸i
        *plist = p;
}
```

Thao t¸c thªm node vµo cuèi danh s¸ch

NÕu danh s¸ch rçng th× thao t¸c nµy trïng víi thao t¸c thªm node míi vµo ®Çu danh s¸ch.

NÕu danh s¸ch kh«ng rçng chóng ta thùc hiÖn nh sau:

- CÊp ph¸t bé nhí cho node;
- G¸n gi¸ trÞ thÝch hîp cho node;
- ChuyÓn con trá tíi node cuèi trong danh s¸ch;
- ThiÕt lËp liªn kÕt tr¸i;
- ThiÕt lËp liªn kÕt ph¶i;

```
void Push_Bottom(NODEPTR *plist, int x){
        NODEPTR p, q;
        if (*plist ==NULL)
                Push_Top(plist, x);
        else {
                p= Getnode();// cÊp ph¸t bé nhí cho node
                p->infor =x; //g¸n gi¸ trÞ thÝch hîp
                //chuyÓn con trá tíi node cuèi danh s¸ch
```

```
            q = *plist;
            while (q->right!=NULL)
                  q = q->right;
            //q lµ node cuèi cïng trong danh s¸ch
            q -> right =p; // liªn kÕt ph¶i
            p->left = q; // liªn kÕt tr¸i
            p->right =NULL; //liªn kÕt ph¶i
      }
}
```

Thªm node vµo tríc node p:

Muèn thªm node vµo tríc node p th× node p ph¶i tån t¹i trong danh s¸ch. NÕu node p tån t¹i th× cã thÓ x¶y ra hai trêng hîp: hoÆc node p lµ node cuèi cïng cña danh s¸ch hoÆc node p lµ node cha ph¶i lµ cuèi cïng. Tr-êng hîp thø nhÊt øng víi thao t¸c Push_Bottom. Trêng hîp thø hai, chóng ta lµm nh sau:

CÊp ph¸t bé nhí cho node;

G¸n gi¸ trÞ thÝch hîp;

ThiÕt lËp liªn kÕt tr¸i cho node míi;

ThiÕt lËp liªn kÕt ph¶i cho node míi;

Qu¸ tr×nh ®îc m« t¶ bëi thñ tôc sau:

```
void Push_Before(NODEPTR p, int x){
      NODEPTR q;
      if (p==NULL) return; //kh«ng lµm g×
      else if (p->next==NULL)
            Push_Bottom(p, x);
      else {
            q = Getnode(); // cÊp ph¸t bé nhí cho node míi
            q ->infor = x; //g¸n gi¸ trÞ th«ng tin thÝch hîp
            q ->right = p->right; //thiÕt lËp liªn kÕt ph¶i
            (p ->right) ->left =q;
            q -> left = p; //thiÕt lËp liªn kÕt tr¸i
            p ->right = q;
```

```
        }
}
```

Lo¹i bá node ®Çu danh s¸ch

NÕu danh s¸ch rçng th× kh«ng cÇn lo¹i bá;

Dïng node p trá tíi ®Çu danh s¸ch;

ChuyÓn gèc lªn node kÕ tiÕp;

Lo¹i bá liªn kÕt víi node p;

Gi¶i phãng p;

```
void Del_Top(NODEPTR *plist){
        NODEPTR p;
        if ( (*plist)==NULL) return; //kh«ng lµm g×
        p = *plist; //p lµ node ®Çu tiªn trong danh s¸ch
        (*plist) = (*plist) -> right; // chuyÓn node gèc tíi node kÕ tiÕp//
        p ->right =NULL; // ng¾t liªn kÕt ph¶i cña p;
        (*plist) ->left ==NULL;//ng¾t liªn kÕt tr¸i víi p
        Freenode(p); //gi¶i phãng p
}
```

Lo¹i bá node ë cuèi danh s¸ch

NÕu danh s¸ch rçng th× kh«ng cÇn lo¹i bá;

NÕu danh s¸ch cã mét node th× nã lµ truêng hîp lo¹i phÇn tö ë ®Çu danh s¸ch;

NÕu danh s¸ch cã nhiÒu h¬n mét node th×

ChuyÓn con trá tíi node cuèi cïng;

Ng¾t liªn kÕt tr¸i cña node;

Ng¾t liªn kÕt ph¶i cña node;

Gi¶i phãng node.

```
void Del_Bottom(NODEPTR *plist) {
        NODEPTR p, q;
        if ((*plist)==NULL) return; //kh«ng lµm g×
        else if ( (*plist) ->right==NULL) Del_Top(plist);
```

```
        else {
            p = *plist; // chuyÓn con trá tíi node cuèi danh s¸ch
            while (p->right!=NULL)
                p =p->right;
            // p lµ node cuèi cña danh s¸ch
            q = p ->left; //q lµ node sau p ;// liÖu ®øng sau hay lµ ®øng tríc (theo m×nh hiÓu mét c¸ch ®¬n gi¶n nh s¬ ®å trªn th× lµ ®øng tríc chø nhØ)
            q ->right =NULL; //ng¾t liªn kÕt ph¶i cña      q//
            p -> left = NULL; //ng¾t liªn kÕt tr¸i cña p
            Freenode(p); //gi¶i phãng p
        }
    }
```

Lo¹i node tríc node p(h×nh nh ®øng tríc cña hä m×nh hiÓu lµ ®øng sau hay sao Êy)

NÕu node p kh«ng cã thùc th× kh«ng thÓ lo¹i bá;

NÕu node p lµ node cuèi th× còng kh«ng thÓ lo¹i bá;

Trêng hîp cßn l¹i ®îc thùc hiÖn nh sau:

Ng¾t liªn kÕt tr¸i víi node p ®ång thêi thiÕt lËp liªn kÕt ph¶i víi node (p->right)->right;

Ng¾t liªn kÕt ph¶i víi node p ®ång thêi thiÕt lËp liªn kÕt tr¸i víi node (p->right)->right;

Gi¶i phãng node p->right.

```
    void Del_Before(NODEPTR p){
        NODEPTR q, r;
        if (p==NULL || p->right==NULL) return;
        /*kh«ng lµm g×
         nÕu node p lµ kh«ng cã thùc hoÆc lµ node cuèi cïng */
        q = (p->right)->right; //q lµ node tríc node p ->right
        r = p->right; // r lµ node cÇn lo¹i bá
        r -> left =NULL; //ng¾t liªn kÕt tr¸i cña r
        r->right ==NULL;//ng¾t liªn kÕt ph¶i cña r
```

```
            p->right =q; //thiÕt lËp liªn kÕt ph¶i míi cho p
            q ->left = p; // thiÕt lËp liªn kÕt tr¸i míi cho p
            Freenode(r); //gi¶i phãng node
        }
```

Chóng ta cã thÓ x©y dùng thªm c¸c thao t¸c lo¹i bá node bªn tr¸i, duyÖt tr¸i, duyÖt ph¶i trªn danh s¸ch mãc nèi kÐp. Nh÷ng thao t¸c ®ã ®îc thÓ hiÖn th«ng qua vÝ dô sau.

VÝ dô 4.7: Cung cÊp th«ng tin vÒ tuyÕn xe löa. Bao gåm: th«ng tin vÒ mçi hµnh tr×nh, c¸c hµnh tr×nh ®i xu«i, c¸c hµnh tr×nh ®i ngîc cña mçi ®oµn tµu.

```
#include    <stdio.h>
#include    <stdlib.h>
#include    <conio.h>
#include    <dos.h>
#include    <string.h>
#include    <math.h>
#include <alloc.h>
#define     TRUE1
#define         FALSE     0
/* CÊu tróc th«ng tin chung vÒ ®oµn tµu */
typedef struct {
    char gatruoc[20];
    char gasau[20];
    int chieudai;
    int thoigian;
} doan;
/* CÊu tróc cña mét nót trong danh s¸ch liªn kÕt kÐp*/
typedef struct node{
    doan infor;
    struct node *left,*right;
};
typedef struct node *NODEPTR;
```

```
/* CÊp ph¸t mét nót cho danh s¸ch liªn kÕt kÐp*/
NODEPTR Getnode(void){
        NODEPTR p;
        p =(NODEPTR) malloc(sizeof(struct node));
        return(p);
}
/* Gi¶i phãng mét nót cña danh s¸ch liªn kÕt kÐp*/
void Freenode( NODEPTR p){
        free(p);
}
/* Khëi ®éng danh s¸ch liªn kÕt kÐp*/
void Initialize(NODEPTR *plist){
        *plist=NULL;
}
/* KiÓm tra tÝnh rçng cña danh s¸ch liªn kÕt kÐp*/
int Empty(NODEPTR *plist){
        if (*plist==NULL)
                return(TRUE);
        return(FALSE);
}
/* x¸c ®Þnh sè nót trong danh s¸ch liªn kÕt kÐp*/
int Listsize(NODEPTR *plist){
        NODEPTR p;int n;
        p=*plist;n=0;
        while(p!=NULL){
                p=p->right;
                n++;
        }
        return(n);
}
/* x¸c ®Þnh con trá chØ nót thø i trong danh s¸ch liªn kÕt kÐp*/
```

```
NODEPTR Nodepointer(NODEPTR *plist,int i){
        NODEPTR p;int vitri;
        p=*plist;vitri=0;
        while (p!=NULL && vitri<i){
                p=p->right;
                vitri++;
        }
        return(p);
}
/* x¸c ®Þnh vÞ trÝ cña nót p trong danh s¸ch liªn kÕt kÐp*/
int Position(NODEPTR *plist, NODEPTR p){
        int vitri; NODEPTR q;
        q = *plist;vitri=0;
        while (q!=NULL && q!=p){
                q = q->right;
                vitri++;
        }
        if (q==NULL)
                return(-1);
        return(vitri);
}
/* Thªm nót míi vµo ®Çu danh s¸ch liªn kÕt kÐp*/
void Push(NODEPTR *plist, doan x){
        NODEPTR p;p = Getnode();
        p ->infor = x;
        if(*plist==NULL){
                p->left=NULL;
                p->right=NULL;
                *plist=p;
        }
        else {
```

```
                p->right = *plist;
                (*plist)->left=p;
                p->left=NULL;
                *plist=p;
        }
}
/* Thªm nót míi vµo sau nót p */
void Insertright(NODEPTR p, doan x){
        NODEPTR q,r;
        if (p==NULL){
                printf("\n Nut khong co thuc");
                delay(2000);return;
        }
        else {
                q=Getnode();// cÊp ph¸t bé nhí cho node q
                q->infor=x;
                r=p->right;// r lµ node sau node p
                r->left=q;// g¸n node ben tr¸I cña r(hay lµ node tríc nã khi hiÓu
theo nghÜa  th«ng thêng) lµ q
                q->right=r;// g¸n node bªn ph¶I cña q lµ r
                q->left=p;// g¸n node bªn tr¸I cña q lµ p
                p->right=q;// g¸n node bªn ph¶I cña p lµ q
        }
}
/* thªm nót míi vµo tríc nót p*/
void Insertleft(NODEPTR *plist, NODEPTR p, doan x){
        NODEPTR q, r;
        if (p==NULL) {
                printf("\n Node khong co thuc");
                delay(2000); return;
        }
```

```
        if (p==*plist)// nÕu p lµ node ®øng ®Çu danh s¸ch
                Push(plist,x);// thªm vµo mét node ë ®Çu danh s¸ch liªn kÕt
        else {
                q=Getnode();// cÊp ph¸t bé nhí cho node q
                q->infor=x;// nhËp th«ng tin cho node q
                r=p->left;// g¸n node r lµ node tríc cña p
                r->right=q;// g¸n node q lµ node sau (ph¶I ) cña r
                q->left=r;// g¸n node bªn tr¸I (node tríc) cña q lµ node r
                q->right=p;// g¸n node bªn ph¶I (node sau) cña q lµ p
                p->left=q;// g¸n node bªn tr¸I (node tríc ) cña p chÝnh lµ q
        }
}
/* xo¸ nót ë ®Çu danh s¸ch*/
doan Pop(NODEPTR *plist){
        NODEPTR p;doan x;
        if (Empty(plist)){
                printf("\n Danh sach rong");
                delay(2000);
        }
        else {
                p = *plist;// cho p lµ node ®Çu danh s¸ch
                x=p->infor;// g¸n cho x th«ng tin cña node p
                if ((*plist)->right==NULL)// khi danh s¸ch chØ cã m«t node
                        *plist=NULL;// g¸n gi¸ trÞ cña danh s¸ch lµ NULL hay nãi
c¸ch kh¸c lµ ®· xo¸ chÝnh node ®ã
                else{
                        *plist=p->right;// ®iÓm ®Çu danh s¸ch dêi xuèng ®iÓm
®øng kÕ sau(bªn ph¶I ) c¶u p
                        (*plist)->left=NULL;// node cÇn xo¸ chÝnh lµ node bªn tr¸I
(node tríc còng chÝnh lµ node ®Çu) cña danh s¸ch – chÝnh lµ node p
                }
                Freenode(p);
```

```
        }
        return(x);
}
/* xo¸ nót cã con trá lµ p trong danh s¸ch*/
doan Delnode(NODEPTR *plist, NODEPTR p) {
        NODEPTR q, r;doan x;
        if (p==NULL){// kh«ng cã node thùc
                printf("\n Node khong co thuc");
                delay(2000);return(x);
        }
        if (*plist==NULL){
                printf("\n Danh sach rong");
                delay(2000);
        }
        else {
                x=p->infor;// g¸n cho gi¸ trÞ x th«ng tin vÒ p
                q = p->left;// g¸n cho q lµ node tr¸I (node liÒn tríc) cña p
                r = p->right;// g¸n cho r lµ node ph¶I (node liÒn sau) cña p
                r ->left = q;// ban ®Çu node tr¸I (®øng liÒn tríc r) chÝnh lµ node p nhng bay giê g¸n node tr¸I (®øng liÒn tríc r) lµ node q(lµ node liÒn tríc p nh ®· nãi ë trªn)
                q ->right=r;// ban ®Çu node ph¶I (®øng liÒn sau) node q chÝnh lµ node p nhng bay giê g¸n node ph¶I (®øng liÒn sau) cña node q chÝnh lµ node r
// nh vËy node p hoµn toµn bÞ xo¸ bá
                Freenode(p);
        }
        return(x);
}
/* duyÖt danh s¸ch tõ tr¸i sang ph¶i */
void Righttraverse(NODEPTR *plist){
        NODEPTR p; int stt;
```

```
        if (Empty(plist)){
                printf("\n Khong cos doan nao");
                delay(2000); return;
        }
        p = *plist;// ®a con trá vÒ ®Çu danh s¸ch
        stt=0;
        while (p!=NULL){
                printf ("\n %5d %20s %20s %7d%7d", stt++,p->infor.gatruoc,
                p->infor.gasau,p->infor.chieudai,p->infor.thoigian);
                p=p->right;
        }
}
/* duyÖt danh s¸ch tõ ph¶i sang tr¸i*/
void Lefttraverse(NODEPTR *plist){
        NODEPTR p;int stt;
        if (Empty(plist)){
                printf("\n Khong co doan nao");
                delay(2000); return;
        }
        stt =0;p = Nodepointer(plist,Listsize(plist)-1);// ®a con trá vÒ cuèi danh
s¸ch
        while (p!=NULL){
                printf("\n %5d %20s%20s%7d%7d", stt++, p->infor.gasau,
                p->infor.gatruoc, p->infor.chieudai, p->infor.thoigian);
                p = p->left;
        }
}
/* t×m th«ng tin vÒ ga tríc */
NODEPTR Search1(NODEPTR *plist, char x[]){
        NODEPTR p=*plist;// ®a con trá vÒ ®Çu danh s¸ch
```

```
        while (strcmp(p->infor.gatruoc,x)!=0 && p!=NULL)// ®Ó t×m th«ng tin vÒ ga tríc ta duyÖt danh s¸ch  tõ trªn xuèng
                p = p->right;
        return(p);
}
/* T×m th«ng tin vÒ ga sau */
NODEPTR Search2(NODEPTR *plist, char x[]){
        NODEPTR p=*plist;
        while (strcmp(p->infor.gasau,x)!=0 && p!=NULL)
                p = p->right;
        return(p);
} // ë ®©y ta còng duyÖt danh s¸ch tõ trªn xuèng hay nãi c¸ch kh¸c lµ tõ tr¸i sang ph¶i
/* lo¹i bá toµn bé c¸c nót cña danh s¸ch*/
void Clearlist(NODEPTR *plist){
        while (*plist!=NULL){
                Pop(plist);
        }
}
/* B¸o lé tr×nh c¸c tuyÕn */
void Message(NODEPTR *plist, char noidi[], char noiden[], char c){
        NODEPTR p, p1;int kc, tg;
        if (c=='x'){
                p=Search1(plist, noidi);
                if (p==NULL){
                        printf("\n Khong co lo trinh");
                        delay(2000);return;
                }
                if (strcmp(noidi, noiden)==0){
                                printf("\n Noi di trung noi den");
                                delay(2000); return;
```

```
        }
        p1= Search2(plist, noiden);
        if (p1==NULL){
                printf("\n Noi den khong co thuc");
                delay(2000); return;
        }
        if (Position(plist,p)<=Position(plist,p1) ) {
                kc=tg=0;
                while(p!=p1){
                        kc = kc + p->infor.chieudai;
                        tg = tg + p->infor.thoigian;
                        printf("\n %20s ->%20s: %7d km %7d gio",
                        p->infor.gatruoc,
                        p->infor.gasau, p->infor.chieudai,
                        p->infor.thoigian);
                        p = p->right;
                }
                kc= kc + p1->infor.chieudai;
                tg=tg + p1 ->infor.thoigian;
                printf("\n %20s ->%20s: %7d km %7d gio",
                p1->infor.gatruoc,
                p1->infor.gasau, p1->infor.chieudai, p1->infor.thoigian);
                printf("\n Tong chieu dai:% 7d Thoi gian:%7d", kc, tg);
                delay(2000);
        }
}
else{
        printf("\n Khong di xuoi duoc");
        delay(2000); return;
}
if (c=='n'){
```

```
p=Search2(plist, noidi);
if (p==NULL){
        printf("\n Khong co lo trinh");
        delay(2000);return;
}
if (strcmp(noidi, noiden)==0){
                printf("\n Noi di trung noi den");
                delay(2000); return;
}
p1= Search1(plist, noiden);
if (p1==NULL){
        printf("\n Noi den khong co thuc");
        delay(2000); return;
}
if (Position(plist,p)<=Position(plist,p1) ) {
        kc=tg=0;
        while(p!=p1){
                kc = kc + p->infor.chieudai;
                tg = tg + p->infor.thoigian;
                printf("\n %20s ->%20s: %7d km %7d gio",
                p->infor.gatruoc,
                p->infor.gasau, p->infor.chieudai,
                p->infor.thoigian);
                p = p->right;
        }
        kc= kc + p1->infor.chieudai;
        tg=tg + p1 ->infor.thoigian;
        printf("\n %20s ->%20s: %7d km %7d gio",
        p1->infor.gatruoc,
        p1->infor.gasau, p1->infor.chieudai, p1->infor.thoigian);
        printf("\n Tong chieu dai:% 7d Thoi gian:%7d", kc, tg);
```

```
                delay(2000);
            }
        }
        else{
            printf("\n Khong di nguoc duoc");
            delay(2000); return;
        }
}
void main (void){
        NODEPTR plist,p, p1;doan ga;char c, noidi[20], noiden[20];
        int vitri,chucnang;
        Initialize(&plist);
        do {
            clrscr();
            printf("\n QUAN SAT TREN TAU");
            printf("\n 1- Them mot doan");
            printf("\n 2- Loai bo mot doan");
            printf("\n 3- Xem lo trinh1");
            printf("\n 4- Xem lo trinh 2");
            printf("\n 5- Xem thong tin doan i");
            printf("\n 6- Hieu chinh thong tin doan i");
            printf("\n 7- Bao lo trinh");
            printf("\n 0- Ket thuc chuong trinh");
            printf("\n Lua chon chuc nang:"); scanf("%d",&chucnang);
            switch(chucnang){
                case 1:
                    printf("\n Vi tri can them:");scanf("%d",&vitri);
                    printf("\n Ten ga truoc:"); scanf("%s",ga.gatruoc);
                    printf("\n Ten ga sau:"); scanf("%s",ga.gasau);
                    printf("\n Chieu dai:"); scanf("%d",&ga.chieudai);
                    printf("\n Thoigian:"); scanf("%d",&ga.thoigian);
```

```
            if (vitri==0)
                  Push(&plist,ga);
            else
                  Insertright(Nodepointer(&plist,vitri-1),ga);
            break;
      case 2:
            printf("\n Vi tri:"); scanf("%d",&vitri);
            p=Nodepointer(&plist,vitri);
            if (p==NULL){
                  printf("\n Vi tri khong hop le");
            }
            else {
                  if (vitri==0) Pop(&plist);
                  else Delnode(&plist,p);
            }
            delay(2000); break;
      case 3:
            printf("\n LO TRINH DUYET XUOI");
            Righttraverse(&plist);delay(2000);break;
      case 4:
            printf("\n LO TRINH DUYET NGUOC");
            Lefttraverse(&plist);delay(2000);break;
      case 5:
            printf("\n Vi tri:");scanf("%d",&vitri);
            p = Nodepointer(&plist,vitri);
            if(p==NULL)
                  printf("\n Vi tri khong hop le");
            else {
                  printf("\n DOAN:%d Tu:%s Den:%s Chieu dai:
                  %d Thoigian :%d",
                  vitri, p->infor.gatruoc, p->infor.gasau,
```

```
                p->infor.chieudai,
                p->infor.thoigian);
        }
        delay(2000); break;
case 6:
        printf("\n Vi tri:"); scanf("%d",&vitri);
        p=Nodepointer(&plist, vitri);
        if(p==NULL)
                printf("\n Vi tri khong hop le");
        else {
                printf("\n DOAN:%d Tu:%s Den:%s Chieu dai:
                %d Thoigian :%d",
                vitri, p->infor.gatruoc, p->infor.gasau,
                p->infor.chieudai,
                p->infor.thoigian);
                printf("\n Ten ga truoc:%s"); scanf("%s",
                 ga.gatruoc);
                printf("\n Ten ga sau:%s"); scanf("%s",
                ga.gasau);
                printf("\n Chieu dai:%d"); scanf("%d",
                &ga.chieudai);
                printf("\n Thoi gian:%d"); scanf("%d",
                &ga.thoigian);
        }
        delay(2000); break;
case 7:
        printf("\n Di xuoi:x Di nguoc: n:");c=getche();
        printf("\n Noi di:");scanf("%s",noidi);
        printf("\n Noi den:"); scanf("%s", noiden);
        Message(&plist, noidi, noiden, c);
        delay(2000); break;
```

```
                }
        } while(chucnang!=0);
}
```

Bµi tËp ch¬ng 4

4.1. X©u thuËn nghÞch ®éc lµ x©u bÝt nhÞ ph©n cã ®é dµi n mµ khi ®¶o x©u ta vÉn nhËn ®îc chÝnh x©u ®ã. H·y liÖt k¸ tÊt c¶ c¸c x©u thuËn nghÞch ®éc cã ®é dµi n vµ ghi l¹i nh÷ng x©u ®ã vµo File thuang.out theo tõng dßng, dßng ®Çu ti¸n ghi l¹i gi¸ trÞ cña n, c¸c dßng tiÕp theo lµ nh÷ng x©u thuËn nghÞch ®éc cã ®é dµi n. VÝ dô: víi n=4, ta cã ®îc nh÷ng x©u thuËn nghÞch ®éc cã d¹ng sau:

4

0 0 0 0

0 1 1 0

1 0 0 1

1 1 1 1

4.2. ViÕt ch¬ng tr×nh qu¶n lý ®iÓm thi cña sinh vi¸n b»ng single (double) link list bao gåm nh÷ng thao t¸c sau:

- NhËp d÷ liÖu;
- HiÓn thÞ d÷ liÖu theo líp, xÕp lo¹i . . .;
- S¾p xÕp d÷ liÖu;
- T×m kiÕm d÷ liÖu;
- In Ên kÕt qu¶.

Trong ®ã, th«ng tin vÒ mçi sinh vi¸n ®îc ®Þnh nghÜa th«ng qua cÊu tróc sau:

```
typedef struct {
    int     masv; // m· sinh vi¸n;
    char    malop[12]; //m· líp
    char    hoten[30]; //hä t¸n sinh vi¸n
    float   diemki; // ®iÓm tæng kÕt kú 1
```

```
    float   diemkii;// ®iÓm tæng kÕt kú 2
    float   diemtk; // ®iÓm tæng kÕt c¶ n¨m
    char    xeploai[12]; // xÕp lo¹i
} sinhvien;
```

4.3. BiÓu diÔn biÓu thøc theo có ph¸p Ba Lan. BiÓu thøc nguyªn lµ mét d·y ®îc thµnh lËp tõ c¸c biÕn kiÓu nguyªn nèi víi nhau b»ng c¸c phÐp to¸n hai ng«i (céng: + , trõ : - , nh©n : *) vµ c¸c dÊu më ngoÆc ®¬n ‘(‘, ®ãng ngoÆc ®¬n ‘)’. Nguyªn t¾c ®Æt tªn biÕn vµ thø tù thùc hiÖn c¸c phÐp to¸n ®îc thùc hiÖn nh sau:

Qui t¾c ®Æt tªn biÕn: Lµ d·y c¸c kÝ tù ch÷ in thêng hoÆc kÝ tù sè ®é dµi kh«ng qu¸ 8, kÝ tù b¾t ®Çu ph¶i lµ mét ch÷ c¸i.

Qui t¾c thùc hiÖn phÐp to¸n: BiÓu thøc trong ngoÆc ®¬n ®îc tÝnh tríc, phÐp to¸n nh©n ‘*’ cã ®é u tiªn cao h¬n so víi hai phÐp to¸n céng vµ trõ. Hai phÐp to¸n céng ‘+’ vµ trõ cã cïng ®é u tiªn. VÝ dô : a * b + c ph¶i ®îc hiÓu lµ: (a * b) + c.

D¹ng viÕt kh«ng ngoÆc Ba Lan cho biÓu thøc nguyªn ®îc ®Þnh nghÜa nh sau:

NÕu e lµ tªn biÕn th× d¹ng viÕt Ba Lan cña nã chÝnh lµ e,

NÕu e_1 vµ e_2 lµ hai biÓu thøc cã d¹ng viÕt Ba Lan t¬ng øng lµ d_1 vµ d2 th× d¹ng viÕt Ba Lan cña $e_1 + e_2$ lµ d_1 d_2+, cña $e_1 - e_2$ lµ d_1 d_2-, cña e_1*e_2 lµ d_1 d_2* (Gi÷a d_1 vµ d_2 cã ®óng mét dÊu c¸ch, tríc dÊu phÐp to¸n kh«ng cã dÊu c¸ch),

NÕu e lµ biÓu thøc cã d¹ng viÕt Ba Lan lµ d th× d¹ng viÕt Ba Lan cña biÓu thøc cã ngoÆc ®¬n (e) chÝnh lµ d (kh«ng cßn dÊu ngoÆc n÷a) . VÝ dô: BiÓu thøc (c+b*(f-d)) cã d¹ng viÕt Ba Lan lµ : c b f d-*+.

Cho file d÷ liÖu balan.in ®îc tæ chøc thµnh tõng dßng, mçi dßng kh«ng dµi qu¸ 80 ký tù lµ biÓu diÔn cña biÓu thøc nguyªn A. H·y dÞch c¸c biÓu thøc nguyªn A thµnh d¹ng viÕt Ba Lan cña A ghi vµo file balan.out theo tõng dßng. VÝ dô: víi file balan.in díi ®©y sÏ cho ta kÕt qu¶ nh sau:

balan.in	balan.out
a+b	a b+
a-b	a b-
a*b	a b*
(a - b) +c	a b- c+

(a + b) * c	a b+ c*
(a + (b-c))	a b c-+
(a + b*(c-d))	a b c d-*+
((a + b) *c- (d + e) * f)	a b+c* d e+f*-

4.4. TÝnh to¸n gi¸ trÞ biÓu thøc Ba Lan. Cho file d÷ liÖu balan.in gåm 2 * n dßng trong ®ã, dßng cã sè thø tù lÎ (1, 3, 5, . .) ghi l¹i mét x©u lµ biÓu diÔn Ba Lan cña biÓu thøc nguyªn A, dßng cã sè thø tù ch½n (2,4,6, . .) ghi l¹i gi¸ trÞ cña c¸c biÕn xuÊt hiÖn trong A. H·y tÝnh gi¸ trÞ cña biÓu thøc A, ghi l¹i gi¸ trÞ cña A vµo file balan.out tõng dßng theo thø tù: Dßng cã thø tù lÎ ghi l¹i biÓu thøc Ba Lan cña A sau khi ®· thay thÕ c¸c gi¸ trÞ t¬ng øng cña biÕn trong A, dßng cã thø tù ch½n ghi l¹i gi¸ trÞ cña biÓu thøc A.

VÝ dô víi file balan.in díi ®©y sÏ cho ta kÕt qu¶ nh sau:

balan.in	balan.out
a b+	3 5+
3 5	8
a b-	7 3-
7 3	4
a b*	4 3 *
4 3	12
c a b-+	3 4 5-+
3 4 5	2

4.5. LËp lÞch víi møc ®é u tiªn. §Ó lËp lÞch cho CPU ®¸p øng cho c¸c qu¸ tr×nh ®ang ®îi cña hÖ thèng, ngêi ta biÓu diÔn mçi qu¸ tr×nh b»ng mét b¶n ghi bao gåm nh÷ng th«ng tin : sè qu¸ tr×nh(Num) lµ mét sè tù nhiªn nhá h¬n 1024, tªn qu¸ tr×nh (Proc) lµ mét x©u ký tù ®é dµi kh«ng qu¸ 32 kh«ng chøa dÊu trèng ë gi÷a, ®é u tiªn qu¸ tr×nh lµ mét sè nguyªn d¬ng (Pri) nhá h¬n 10, thêi gian thùc hiÖn cña qu¸ tr×nh (Time) lµ mét sè thùc. C¸c qu¸ tr×nh ®ang ®îi trong hÖ ®îc CPU ®¸p øng th«ng qua mét hµng ®îi ®îc gäi lµ hµng ®îi c¸c qu¸ tr×nh, hµng ®îi c¸c qu¸ tr×nh víi ®é u tiªn ®îc x©y dùng sao cho nh÷ng ®iÒu kiÖn sau ®îc tho¶ m·n:

- C¸c qu¸ tr×nh ®îc s¾p theo thø tù u tiªn;

- §èi víi nh÷ng qu¸ tr×nh cã cïng ®é u tiªn th× qu¸ tr×nh nµo cã thêi gian thùc hiÖn Ýt nhÊt ®îc xÕp lªn tríc nhÊt.

Cho file d÷ liÖu lich.in ®îc tæ chøc nh sau:

- Dßng ®Çu tiªn ghi l¹i mét sè tù nhiªn n lµ sè c¸c qu¸ tr×nh;

- n dßng kÕ tiÕp, mçi dßng ghi l¹i th«ng tin vÒ mét qu¸ tr×nh ®ang ®îi.

H·y x©y dùng hµng ®îi c¸c qu¸ tr×nh víi ®é u tiªn. Ghi l¹i thø tù c¸c qu¸ tr×nh mµ CPU ®¸p øng trªn mét dßng cña file lich.out, mçi qu¸ tr×nh ®îc ph©n biÖt víi nhau bëi mét hoÆc vµi ký tù trèng, dßng kÕ tiÕp ghi l¹i sè giê cÇn thiÕt mµ CPU cÇn ®¸p øng cho c¸c qu¸ tr×nh. VÝ dô víi file lich.in díi ®©y sÏ cho ta kÕt qu¶ nh sau:

lich.in

7

1 Data_Processing 1 10

2 Editor_Program 1 20

3 System_Call 3 0.5

4 System_Interative 3 1

5 System_Action 3 2

6 Writing_Data 2 20

7 Reading_Data 2 10

lich.out

3 4 5 7 6 1 2

63.5

4.6. **ThuËt to¸n RR (Round Robin):** ThuËt to¸n SJF ®¸p øng ®îc tèi ®a c¸c qu¸ tr×nh ho¹t ®éng trong hÖ, tuy nhiªn sÏ cã nhiÒu qu¸ tr×nh cã chi phÝ thêi gian lín ph¶i ®îi nhiÒu qu¸ tr×nh cã chi phÝ thêi gian nhá thùc hiÖn. Víi thuËt to¸n SJF , tÝnh c«ng b»ng cña hÖ bÞ vi ph¹m. §Ó kh¾c phôc ®iÒu trªn, thuËt to¸n Round Robin thùc hiÖn chän mét lîng tö thêi gian thÝch hîp, sau ®ã ®¸p øng cho mçi qu¸ tr×nh theo tõng vßng víi lîng tö thêi gian ®· chän. ¦u ®iÓm cña RR lµ tÝnh c«ng b»ng cña hÖ ®îc ®¶m b¶o, sè c¸c qu¸ tr×nh ®îc CPU ®¸p øng trªn mét ®¬n vÞ thêi gian chÊp nhËn ®îc. Nhîc ®iÓm lín nhÊt cña thuËt to¸n lµ viÖc lùa chän lîng tö thêi gian ®¸p øng cho mçi qu¸ tr×nh sao cho tèi u kh«ng ph¶i lµ ®¬n gi¶n. H·y viÕt ch¬ng tr×nh m« pháng thuËt to¸n lËp lÞch RR.

4.7. **Memory Management** (Qu¶n lý bé nhí)

Qu¶n lý bé nhí lµ qu¸ tr×nh ®iÒu khiÓn viÖc n¹p c¸c qu¸ tr×nh vµo bé nhí. Bé nhí cã thÓ ®îc coi lµ mét m¶ng mét chiÒu hoÆc nhiÒu chiÒu, mçi « nhí ®îc x¸c ®Þnh th«ng qua ®Þa chØ cña nã. §Þa chØ cña « nhí phô thuéc vµo ph¬ng ph¸p qu¶n lý bé nhí, c¸c ph¬ng ph¸p qu¶n lý bé nhí th«ng dông hiÖn nay lµ ph©n trang (Paging) hoÆc ph©n ®o¹n (Segmentation). ë mçi thêi ®iÓm, cã thÓ cã nhiÒu qu¸ tr×nh ®îc n¹p vµo nh÷ng vïng bé nhí kh¸c nhau lµm cho bé nhí cã nh÷ng lç hæng d thõa gäi lµ Hole. Bµi to¸n ®Æt ra lµ lµm thÓ nµo ®Ó cã thÓ n¹p tiÕp c¸c qu¸ tr×nh vµo c¸c Hole bé nhí. Nh÷ng thuËt to¸n c¬ b¶n sau sÏ gióp chóng ta minh häa ®iÒu ®ã:

ThuËt to¸n First Fit: DuyÖt theo danh s¸ch c¸c lç hæng bé nhí, chän lç hæng bé nhí ®Çu tiªn cã thÓ ®Ó thùc hiÖn viÖc n¹p ch¬ng tr×nh. ¦u ®iÓm cña ph¬ng ph¸p nµy lµ ph¬ng ph¸p cµi ®Æt ®¬n gi¶n, tèc ®é nhanh. Nhîc ®iÓm lín nhÊt cña nã lµ qu¸ tr×nh n¹p cã thÓ t¹o nªn c¸c Hole mÞn h¬n.

ThuËt to¸n Best Fit: DuyÖt theo danh s¸ch c¸c lç hæng bé nhí, t×m lç hæng bé nhí cã kÝch cì thÝch hîp nhÊt ®Ó thùc hiÖn n¹p ch¬ng tr×nh. ¦u ®iÓm cña ph¬ng ph¸p nµy lµ c¸c lç hæng bé nhí ®îc sö dông triÖt ®Ó. Nhîc ®iÓm cña nã lµ cµi ®Æt phøc t¹p, qu¸ tr×nh duyÖt cã thÓ mÊt nhiÒu thêi gian.

ThuËt to¸n Worsd Fit: Lu«n lÊy lç hæng bé nhí lín nhÊt ®Ó thùc hiÖn viÖc n¹p ch¬ng tr×nh. ¦u ®iÓm cña nã lµ ®¬n gi¶n, dÔ cµi ®Æt. Nhîc ®iÓm cña nã lµ g©y l·ng phÝ bé nhí.

H·y viÕt ch¬ng tr×nh m« pháng l¹i nh÷ng thuËt to¸n qu¶n lý bé nhí trªn.