Bài 1:

NGÔN NGỮ LẬP TRÌNH & PHƯƠNG PHÁP LẬP TRÌNH

1.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Ý nghĩa, các bước lập trình.
- Xác định dữ liệu vào, ra.
- Phân tích các bài toán đơn giản.
- Khái niệm so sánh, lặp.
- Thể hiện bài toán bằng lưu đồ.

1.2 Lý thuyết

1.2.1 Ngôn ngữ lập trình (Programming Language)

Phần này chúng ta sẽ tìm hiểu một số khái niệm căn bản về thuật toán, chương trình, ngôn ngữ lập trình. Thuật ngữ "thuật giải" và "thuật toán" dĩ nhiên có sự khác nhau song trong nhiều trường hợp chúng có cùng nghĩa.

1.2.1.1Thuật giải (Algorithm)

Là một dãy các thao tác xác định trên một đối tượng, sao cho sau khi thực hiện một số hữu hạn các bước thì đạt được mục tiêu. Theo R.A.Kowalski thì bản chất của thuật giải:

Thuật giải = Logic + Điều khiển

* *Logic*: Đây là phần khá quan trọng, nó trả lời câu hỏi "Thuật giải làm gì, giải quyết vấn đề gì?", những yếu tố trong bài toán có quan hệ với nhau như thế nào v.v… Ở đây bao gồm những kiến thức chuyên môn mà bạn phải biết để có thể tiến hành giải bài toán.

1.2.1.1Thuật giải (Algorithm)

Ví dụ 1: Để giải một bài toán tính diện tích hình cầu, mà bạn không còn nhớ công thức tính hình cầu thì bạn không thể viết chương trình cho máy để giải bài toán này được.

* ***Điều khiển***: Thành phần này trả lời câu hỏi: giải thuật phải làm như thế nào?. Chính là cách thức tiến hành áp dụng thành phần logic để giải quyết vấn đề..

1.2.1.2 Chương trình (Program)

Là một tập hợp các mô tả, các phát biểu, nằm trong một hệ thống qui ước về ý nghĩa và thứ tự thực hiện, nhằm điều khiển máy tính làm việc. Theo Niklaus Wirth thì:

Chương trình = Thuật toán + Cấu trúc dữ liệu

Các thuật toán và chương trình đều có cấu trúc dựa trên ***3 cấu trúc điều khiển cơ bản***:

* ***Tuần tự*** (Sequential): Các bước thực hiện tuần tự một cách chính xác từ trên xuống,

mỗi bước chỉ thực hiện đúng một lần.

* ***Chọn lọc*** (Selection): Chọn 1 trong 2 hay nhiều thao tác để thực hiện.

* ***Lặp lại*** (Repetition): Một hay nhiều bước được thực hiện lặp lại một số lần.

Muốn trở thành lập trình viên chuyên nghiệp bạn hãy làm đúng trình tự để có thói quen tốt và thuận lợi sau này trên nhiều mặt của một người làm máy tính. Bạn hãy làm theo các bước sau:

® Tìm, xây dựng thuật giải (trên giấy)

® viết chương trình trên máy

® dịch chương trình

® chạy và thử chương trình

1.2.1.3 Ngôn ngữ lập trình (Programming language)

Ngôn ngữ lập trình là hệ thống các ký hiệu tuân theo các qui ước về ngữ pháp và ngữ

nghĩa, dùng để xây dựng thành các chương trình cho máy tính.

Một chương trình được viết bằng một ngôn ngữ lập trình cụ thể (ví dụ Pascal, C…) gọi là

chương trình nguồn, chương trình dịch làm nhiệm vụ dịch chương trình nguồn thành chương

trình thực thi được trên máy tính

1.2.2 Các bước lập trình

Bước 1: Phân tích vấn đề và xác định các đặc điểm. (xác định I-P-O)

Bước 2: Lập ra giải pháp. (đưa ra thuật giải)

Bước 3: Cài đặt. (viết chương trình)

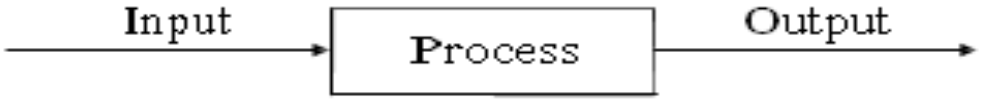
Bước 4: Chạy thử chương trình. (dịch chương trình)

Bước 5: Kiểm chứng và hoàn thiện chương trình. (thử nghiệm bằng nhiều số liệu và đánh giá)

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

Quy trình xử lý cơ bản của máy tính gồm I-P-O.



<u>Ví dụ 2</u>: Xác định Input, Process, Output của việc làm 1 ly nước chanh nóng

Input : ly, đường, chanh, nước nóng, muỗng.

Process:

- cho hỗn hợp đường, chanh, nước nóng vào ly.
- dùng muỗng khuấy đều.

Output: ly chanh nóng đã sẵn sàng để dùng.

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

Ví dụ 3: Xác định Input, Process, Output của chương trình tính tiền lương công nhân

tháng 10/2002 biết rằng lương = lương căn bản * ngày công

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

Input **: lương căn bản, ngày công**

Process **: nhân lương căn bản với ngày công**

Output **: lương**

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

***Ví dụ 4**: Xác định Input, Process, Output của chương trình giải phương trình bậc nhất*

$ax + b = 0$

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

Input **:** ***hệ số a, b***

Process **:** ***chia – b cho a***

Output **:** ***nghiệm x***

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

***Ví dụ 5**: Xác định Input, Process, Output của chương trình tìm số lớn nhất của 2 số a và b.*

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

Input ***: a, b***

Process ***: Nếu a > b thì Output = a lớn nhất***

Ngược lại Output = b lớn nhất

1.2.3 Kỹ thuật lập trình

1.2.3.1 I-P-O Cycle (Input-Pprocess-Output Cycle) (Quy trình nhập-xử lý-xuất)

@ Bài tập:

Xác định Input, Process, Output của các chương trình sau:

1. Đổi từ tiền VND sang tiền USD.Biết tỉ giá 1USD = 18.84VND

2. Tính điểm trung bình của học sinh gồm các môn Toán, Lý, Hóa.

3. Giải phương trình bậc 2: ax2 + bx + c = 0

4. Đổi từ độ sang radian và đổi từ radian sang độ

(công thức a/p = a/180, với a: radian, a: độ)

5. Kiểm tra 2 số a, b giống nhau hay khác nhau

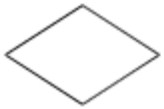
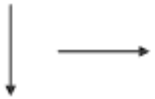
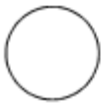
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

Để dễ hơn về quy trình xử lý, các nhà lập trình đưa ra dạng lưu đồ để minh họa từng bước quá trình xử lý một vấn đề (bài toán).

1.2.3 Kỹ thuật lập trình

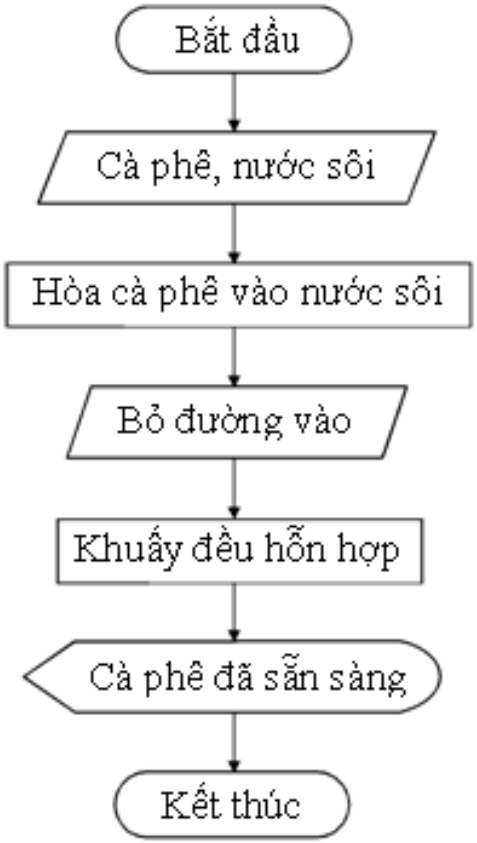
1.2.3.2 Sử dụng lưu đồ (Flowchart)

Hình dạng (symbol)	Hành động (Activity)
	Dữ liệu vào (Input)
	Xử lý (Process)
	Dữ liệu ra (Output)
	Quyết định (Decision), sử dụng điều kiện
	Luồng xử lý (Flow lines)
	Gọi CT con, hàm… (Procedure, Function…)
	Bắt đầu, kết thúc (Begin, End)
	Điểm ghép nối (Connector)

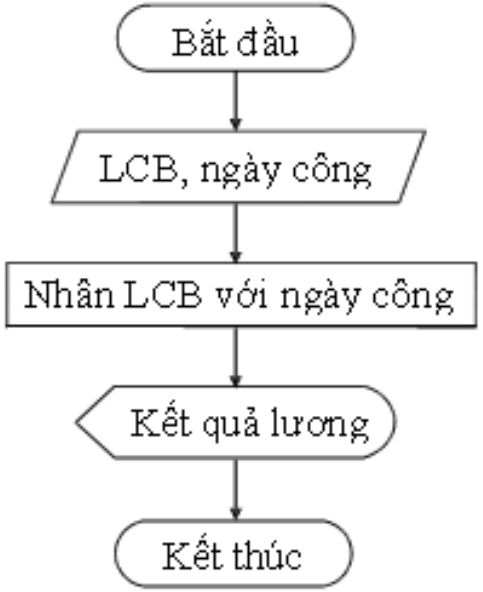
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

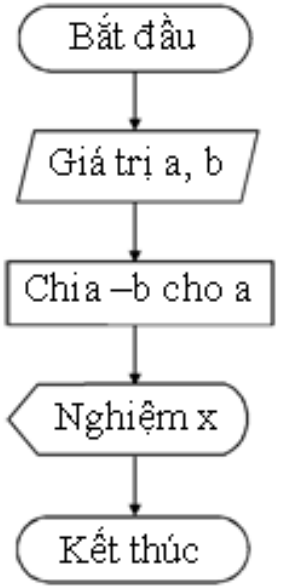
Ví dụ6: Chuẩn bị cà phê



Ví dụ7: Mô tả ví dụ 3



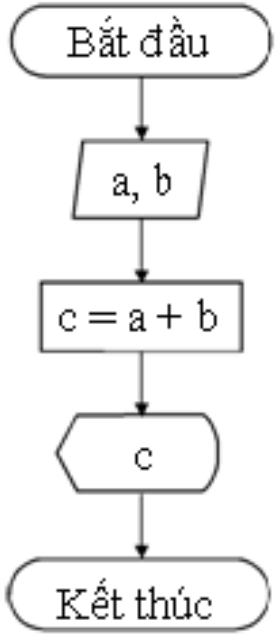
Ví dụ8: Mô tả ví dụ 4



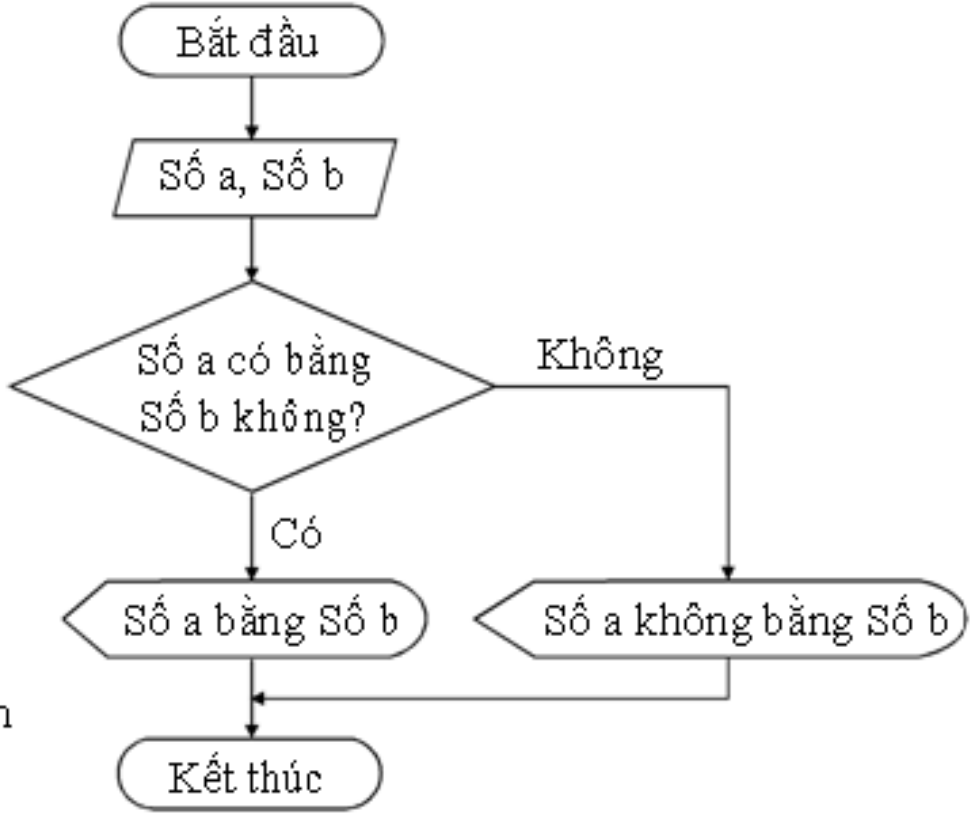
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

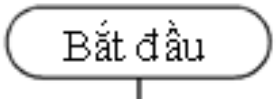
Ví dụ9: Cộng 2 số



Ví dụ10: so sánh 2 số



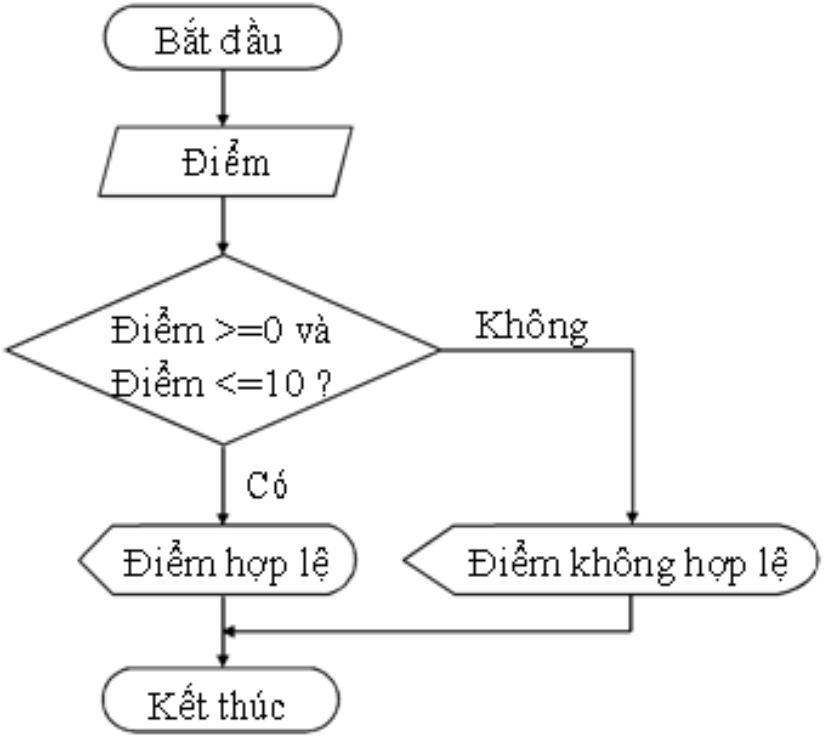
Ví dụ11: Kiểm tra tính hợp lệ của điểm



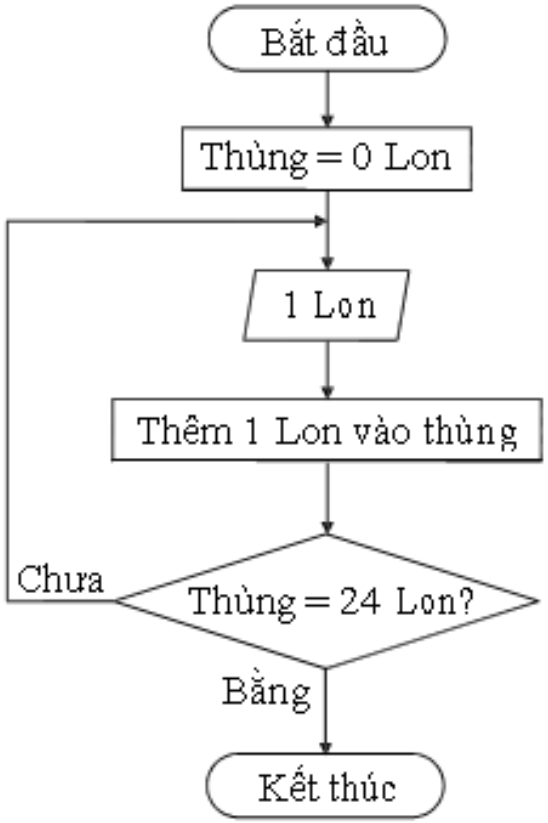
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

Vi du11: Kiêm tra tính hợp lệ của điêm



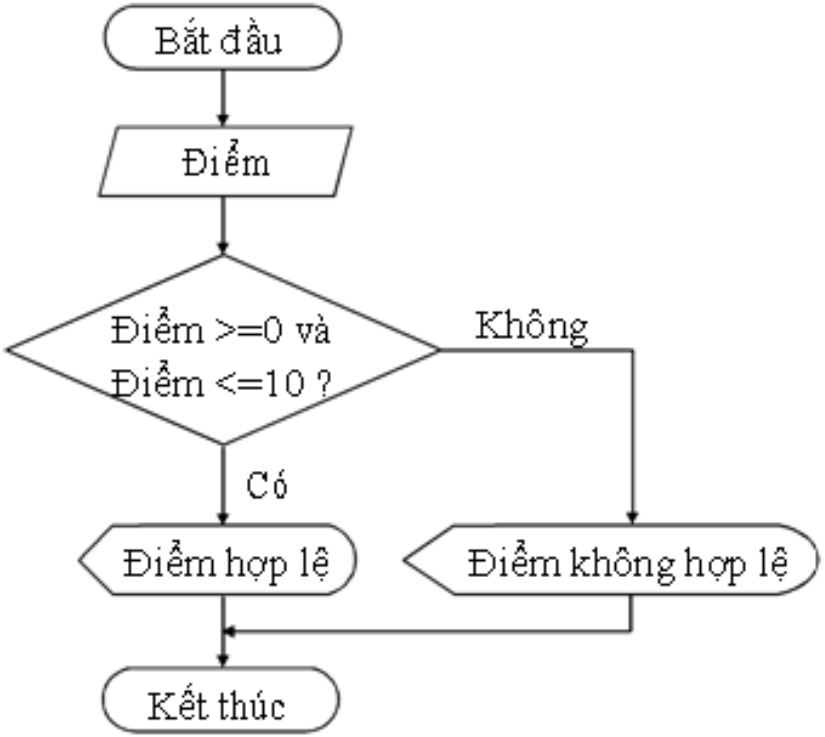
Vi du12: Xếp lon vào thùng



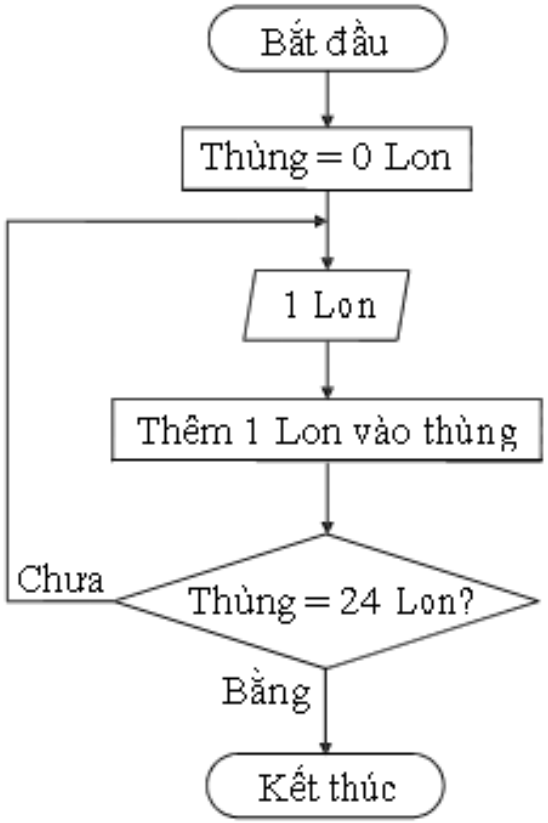
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

Vi du11: Kiêm tra tính hợp lệ của điêm



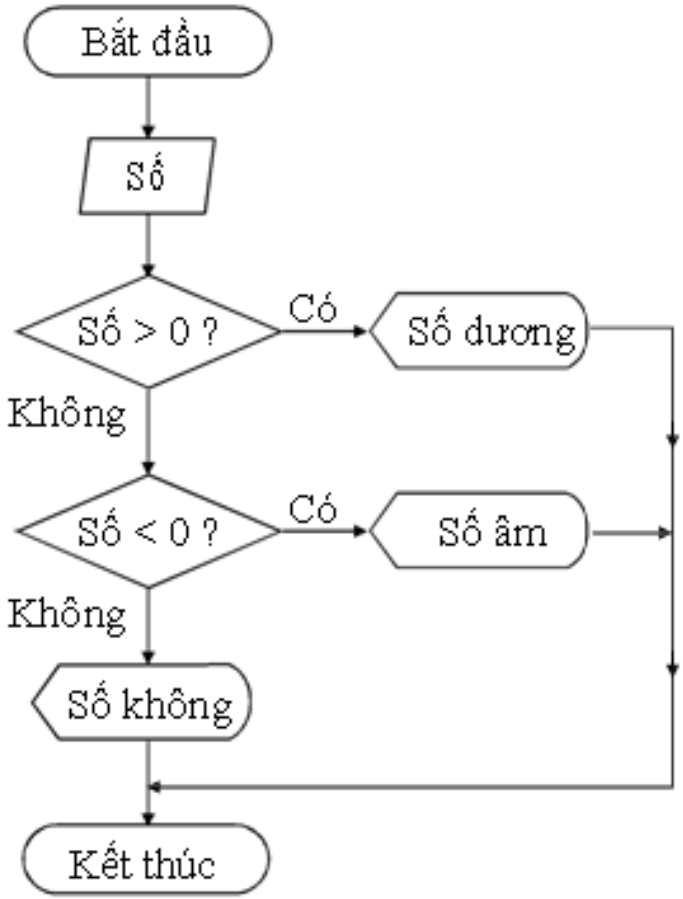
Vi du12: Xếp lon vào thùng



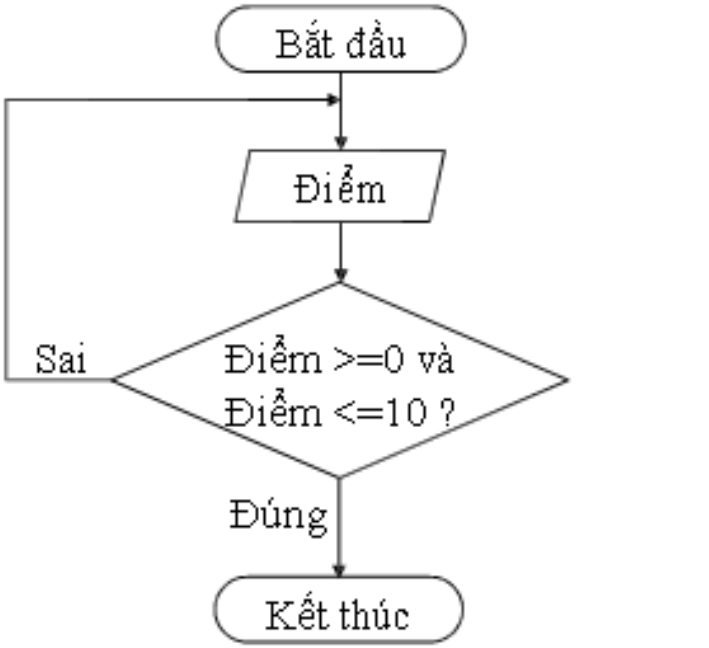
1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

Vi du13: Kiểm tra loại số



Vi du14: Kiểm tra tính hợp lệ của điểm



1.2.3 Kỹ thuật lập trình

1.2.3.2 Sử dụng lưu đồ (Flowchart)

@ **Bài tập**

Vẽ lưu đồ cho các chương trình sau:

1. Đổi từ tiền VND sang tiền USD.

2. Tính điểm trung bình của học sinh gồm các môn Toán, Lý, Hóa.

3. Giải phương trình bậc 2: ax2 + bx + c = 0

4. Đổi từ độ sang radian và đổi từ radian sang độ

(công thức a/p = a/180, với a: radian, a: độ)

5. Kiểm tra 2 số a, b giống nhau hay khác nhau.

Bài 2 :
LÀM QUEN LẬP TRÌNH C QUA CÁC VÍ DỤ ĐƠN GIẢN

2.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Ngôn ngữ C.
- Một số thao tác cơ bản của trình soạn thảo C.
- Cách lập trình trên C.
- Tiếp cận một số lệnh đơn giản thông qua các ví dụ.
- Nắm bắt được một số kỹ năng đơn giản.

2.2 Nội dung

2.2.1 Khởi động và thoát BorlandC

2.2.1.1 Khởi động

Nhập lệnh tại dấu nhắc DOS: **gõ BC (Enter)** (nếu đường dẫn đã được cài đặt bằng lệnh **path** trong đó có chứa đường dẫn đến thư mục chứa tập tin BC.EXE). Nếu đường dẫn chưa được cài đặt ta tìm xem thư mục BORLANDC nằm ở ổ đĩa nào. Sau đó ta gõ lệnh sau:

<ổ đĩa>:\BORLANDC\BIN\BC ¿ (Enter)

2.2 Nội dung

2.2.1 Khởi động và thoát BorlandC

2.2.1.1 Khởi động

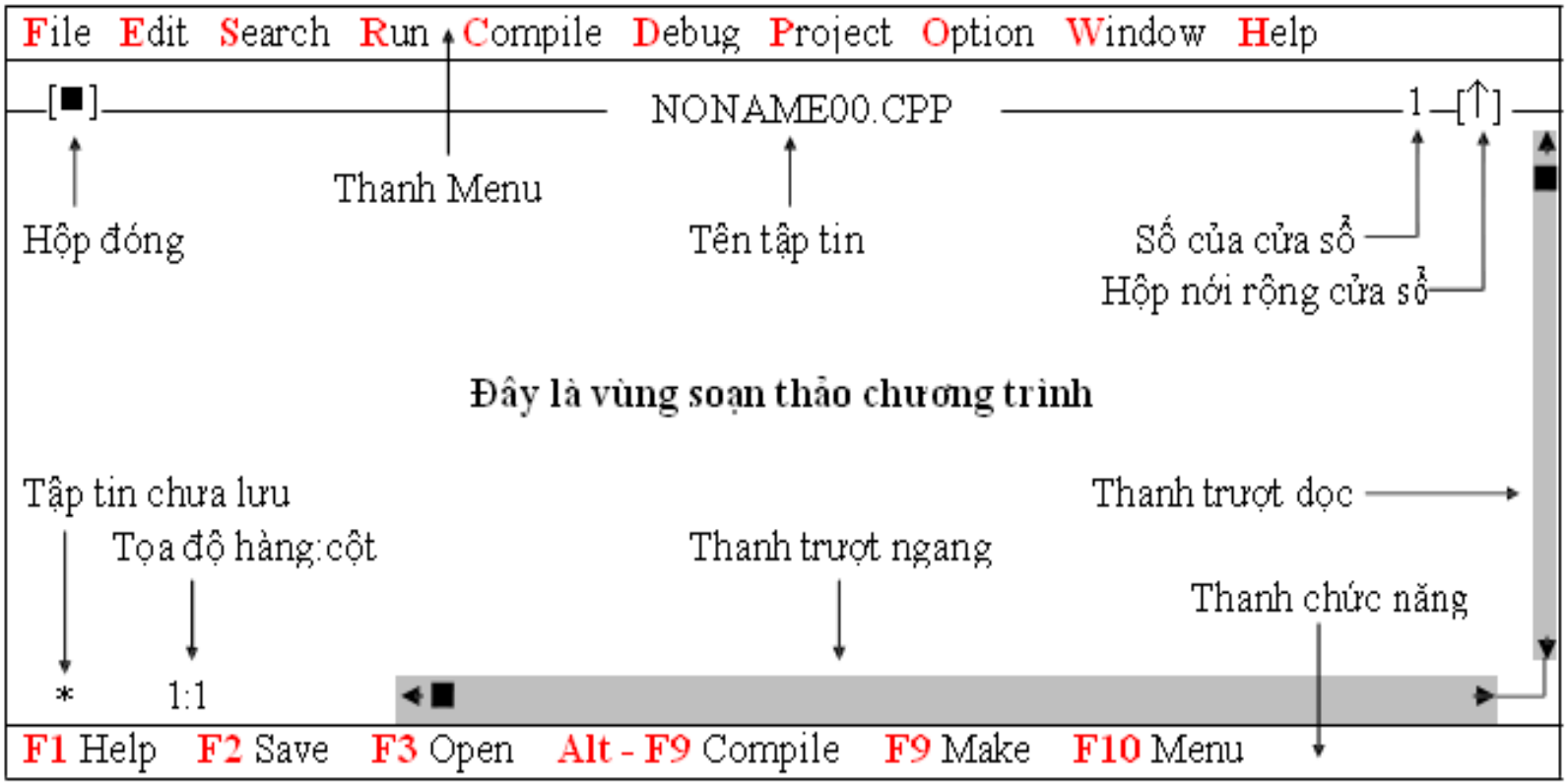
Nếu bạn muốn vừa khởi động BC vừa soạn thảo chương trình với một tập tin có tên do chúng ta đặt, thì gõ lệnh: BC [đường dẫn]<tên file cần soạn thảo>, nếu tên file cần soạn thảo đã có thì được nạp lên, nếu chưa có sẽ được tạo mới.

Khởi động tại Windows: Bạn vào menu Start, chọn Run, bạn gõ vào hộp Open 1 trong các dòng lệnh như nhập tại DOS. Hoặc bạn vào Window Explorer, chọn ổ đĩa chứa thư mục BORLANDC, vào thư mục BORLANDC, vào thư mục BIN, khởi động tập tin BC.EXE

2.2 Nội dung

2.2.1 Khởi động và thoát BorlandC

2.2.1.1 Khởi động



2.2.1 Khởi động và thoát BorlandC

2.2.1.2 Thoát

Ấn phím **F10** (kích hoạt Menu), chọn menu **File**, chọn **Quit**; Hoặc ấn tổ hợp phím **Alt – X**.

2.2.2 Các ví dụ đơn giản

2.2.2.1 Ví dụ 1

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	
4	void main(void)
5	{
6	printf("Bai hoc C dau tien.");
7	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ ***Kết quả in ra màn hình***

```
Bai hoc C dau tien.
```

Dòng thứ 1: bắt đầu bằng /* và kết thúc bằng */ cho biết hàng này là hàng diễn giải (chú thích). Khi dịch và chạy chương trình, dòng này không được dịch và cũng không thi hành lệnh gì cả. Mục đích của việc ghi chú này giúp chương trình rõ ràng hơn. Sau này bạn đọc lại chương trình biết chương trình làm gì.

2.2.2.1 Ví dụ 1

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	
4	void main(void)
5	{
6	printf("Bai hoc C dau tien.");
7	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ ***Kết quả in ra màn hình***

```
Bai hoc C dau tien.
```

Dòng thứ 2: chứa phát biểu tiền xử lý ***#include <stdio.h>***. Vì trong chương trình này ta sử dụng hàm thư viện của C là ***printf***, do đó bạn cần phải có khai báo của hàm thư viện này để báo cho trình biên dịch C biết. **Nếu không khai báo chương trình sẽ báo lỗi**.

Dòng thứ 3: hàng trắng viết ra với ý đồ làm cho bảng chương trình thoáng, dễ đọc.

2.2.2.1 Ví dụ 1

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	
4	void main(void)
5	{
6	printf("Bai hoc C dau tien.");
7	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

☞ ***Kết quả in ra màn hình***

```
Bai hoc C dau tien.
```

Dòng thứ 4: ***void main(void)*** là thành phần chính của mọi chương trình C (bạn có thể viết main() hoặc void main() hoặc main(void)). Tuy nhiên, bạn nên viết theo dạng void main(void) để chương trình rõ ràng hơn. Mọi chương trình C đều bắt đầu thi hành từ hàm main. Cặp dấu ngoặc () cho biết đây là khối hàm (function). Hàm void main(void) có từ khóa void đầu tiên cho biết hàm này không trả về giá trị, từ khóa void trong ngoặc đơn cho biết hàm này không nhận vào đối số

2.2.2.1 Ví dụ 1

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	
4	void main(void)
5	{
6	printf("Bai hoc C dau tien.");
7	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ ***Kết quả in ra màn hình***

```
Bai hoc C dau tien.
```

Dòng thứ 5 và 7: cặp dấu ngoặc móc {} giới hạn thân của hàm. Thân hàm bắt đầu bằng dấu { và kết thúc bằng dấu }.

Dòng thứ 6: ***printf("Bai hoc C dau tien.");***, chỉ thị cho máy in ra chuỗi ký tự nằm trong nháy kép (""). Hàng này được gọi là một câu lệnh, kết thúc một câu lệnh trong C phải là dấu chấm phẩy (;).

+ Chú ý:

- Các từ include, stdio.h, void, main, printf phải viết bằng chữ thường.

- Chuỗi trong nháy kép cần in ra "Bạn có thể viết chữ HOA, thường tùy, ý".

- Kết thúc câu lệnh phải có dấu chấm phẩy.

- Kết thúc tên hàm không có dấu chấm phẩy hoặc bất cứ dấu gì.

- Ghi chú phải đặt trong cặp /* …. */.

- Thân hàm phải được bao bởi cặp { }.

- Các câu lệnh trong thân hàm phải viết thụt vào.

- Bạn nhập đoạn chương trình trên vào máy. Dịch, chạy và quan sát kết quả.

Ctrl – F9: Dịch và chạy chương trình. Alt – F5: Xem màn hình kết quả.

- Sau khi bạn nhập xong đoạn chương trình vào máy. Bạn Ấn và giữ phím Ctrl, gõ F9 để dịch và chạy chương trình. Khi đó bạn thấy chương trình chớp rất nhanh và không thấy kết quả gì cả. Bạn Ấn và giữ phím Alt, gõ F5 để xem kết quả, khi xem xong, bạn ấn phím bất kỳ để quay về màn hình soạn thảo chương trình.

-Bây giờ bạn sửa lại dòng thứ 6 bằng câu lệnh

```
printf("Bai hoc C dau tien.\n");
```

sau đó dịch và chạy lại chương trình, quan sát kết quả.

Ở dòng bạn vừa sửa có thêm \n, \n là ký hiệu xuống dòng sử dụng trong lệnh printf. Sau đây

là một số ký hiệu khác.

+ *Các kí tự điều khiển*:

\n : Nhảy xuống dòng kế tiếp canh về cột đầu tiên.

\t : Canh cột tab ngang.

\r : Nhảy về đầu hàng, không xuống hàng.

\a : Tiếng kêu bip.

+ *Các kí tự đặc biệt*:

\\ : In ra dấu \

\" : In ra dấu "

\' : In ra dấu '

- Mỗi khi chạy chương trình bạn thấy rất bất tiện trong việc xem kết quả phải ấn tổ hợp

phím Alt – F5. Để khắc phục tình trạng này bạn sửa lại chương trình như sau:

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	#include <conio.h>
4	
5	void main(void)
6	{
7	printf("\t\tBai hoc C \r dau tien.\n");
8	getch();
9	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Kết quả in ra màn hình

dau tien. Bai hoc C

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh in ra cau bai hoc C dau tien */
2	#include <stdio.h>
3	#include <conio.h>
4	
5	void main(void)
6	{
7	printf("\t\tBai hoc C \rdau tien.\n");
8	getch();
9	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Dòng thứ 3: chứa phát biểu tiền xử lý ***#include <conio.h>***. Vì trong chương trình này ta sử dụng hàm thư viện của C là ***getch***, do đó bạn cần phải có khai báo của hàm thư viện này để báo cho trình biên dịch C biết. **Nếu không khai báo chương trình sẽ báo lỗi**.

Dòng thứ 8: ***getch();***, chờ nhận 1 ký tự bất kỳ từ bàn phím, nhưng không in ra màn hình. Vì thế ta sử dụng hàm này để khi chạy chương trình xong sẽ dừng lại ở màn hình kết quả, sau đó ta ấn phím bất kỳ sẽ quay lại màn hình soạn thảo.

Ví dụ 2

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap va in ra man hinh gia tri bien*/
2	#include <stdio.h>
3	#include <conio.h>
4	
5	void main(void)
6	{
7	int i;
8	printf("Nhap vao mot so: ");
9	scanf("%d", &i);
10	printf("So ban vua nhap la: %d.\n", i);
11	getch();
12	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

- ***Kết quả in ra màn hình***

```
Nhap vao mot so: 15
So ban vua nhap la: 15.
```

Ví dụ 2

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap va in ra man hinh gia tri bien*/
2	#include <stdio.h>
3	#include <conio.h>
4	
5	void main(void)
6	{
7	int i;
8	printf("Nhap vao mot so: ");
9	scanf("%d", &i);
10	printf("So ban vua nhap la: %d.\n", i);
11	getch();
12	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

Dòng thứ 7: ***int i;*** là lệnh khai báo, mẫu tự i gọi là tên biến. Biến là một vị trí trong bộ nhớ dùng lưu trữ giá trị nào đó mà chương trình sẽ lấy để sử dụng. Mỗi biến phải thuộc một kiểu dữ liệu. Trong trường hợp này ta sử dụng biến i kiểu số nguyên (integer) viết tắt là int.

```
Dòng | File  Edit  Search  Run  Compile  Debug  Project  Option  Window  Help
   1 | /* Chuong trinh nhap va in ra man hinh gia tri bien*/
   2 | #include <stdio.h>
   3 | #include <conio.h>
   4 |
   5 | void main(void)
   6 | {
   7 |     int i;
   8 |     printf("Nhap vao mot so: ");
   9 |     scanf("%d", &i);
  10 |     printf("So ban vua nhap la: %d.\n", i);
  11 |     getch();
  12 | }
     | F1 Help  Alt-F8 Next Msg  Alt-F7 Prev Msg  Alt - F9 Compile  F9 Make  F10 Menu
```

Dòng thứ9: ***scanf("%d", &i)***. Sử dụng hàm scanf để nhận từ người sử dụng một trị nào đó. Hàm scanf trên có 2 đối mục. Đối mục ***"%d"*** được gọi là chuỗi định dạng, cho biết loại dữ kiện mà người sử dụng sẽ nhập vào. Chẳng hạn, ở đây phải nhập vào là số nguyên. Đối mục thứ 2

&i có dấu & đi đầu gọi là address operator, dấu & phối hợp với tên biến cho hàm scanf biến đem

trị gõ từ bàn phím lưu vào biến i.

```
Dòng | File  Edit  Search  Run  Compile  Debug  Project  Option  Window  Help
   1 | /* Chuong trinh nhap va in ra man hinh gia tri bien*/
   2 | #include <stdio.h>
   3 | #include <conio.h>
   4 |
   5 | void main(void)
   6 | {
   7 |     int i;
   8 |     printf("Nhap vao mot so: ");
   9 |     scanf("%d", &i);
  10 |     printf("So ban vua nhap la: %d.\n", i);
  11 |     getch();
  12 | }
     | F1 Help  Alt-F8 Next Msg  Alt-F7 Prev Msg  Alt - F9 Compile  F9 Make  F10 Menu
```

Dòng thứ 10: ***printf("So ban vua nhap la: %d.\n", i);***. Hàm này có 2 đối mục. Đối mục thứ nhất là một chuỗi định dạng có chứa chuỗi văn bản ***So ban vua nhap la:*** và ***%d*** (ký hiệu khai báo chuyển đổi dạng thức) cho biết số nguyên sẽ được in ra. Đối mục thứ 2 là i cho biết giá trị lấy từ biến i để in ra màn hình.

2.2.2.3 Ví dụ 3

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap vao 2 so a, b in ra tong*/
2	#include <stdio.h>
3	#include <conio.h>
4	
5	void main(void)
6	{
7	int a, b;
8	printf("Nhap vao so a: ");
9	scanf("%d", &a);
10	printf("Nhap vao so b: ");
11	scanf("%d", &b);
12	printf("Tong cua 2 so %d va %d la %d.\n", a, b, a+b);
13	getch();
14	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ ***Kết quả in ra màn hình***

```
Nhap vao so a: 4
Nhap vao so b: 14
Tong cua 2 so 4 va 14 la 18.
```

- Dòng thứ12: ***printf("Tong cua 2 so %d va %d la %d.\n", a, b, a+b);***

2.2.2.4 Ví dụ 4

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap vao ban kinh hinh tron. Tinh dien tich */
2	#include <stdio.h>
3	#include <conio.h>
4	
5	#define PI 3.14
6	
7	void main(void)
8	{
9	float fR;
10	printf("Nhap vao ban kinh hinh tron: ");
11	scanf("%f", &fR);
12	printf("Dien tich hinh tron: %.2f.\n", 2*PI*fR);
13	getch();
14	}
	F1 Help **Alt-F8** Next Msg **Alt-F7** Prev Msg **Alt - F9** Compile **F9** Make **F10** Menu

☞ ***Kết quả in ra màn hình***

```
Nhap vao ban kinh hinh tron: 1
Dien tich hinh tron: 6.28
```

2.2.2.4 Ví dụ 4

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap vao ban kinh hinh tron. Tinh dien tich */
2	#include <stdio.h>
3	#include <conio.h>
4	
5	#define PI 3.14
6	
7	void main(void)
8	{
9	float fR;
10	printf("Nhap vao ban kinh hinh tron: ");
11	scanf("%f", &fR);
12	printf("Dien tich hinh tron: %.2f.\n", 2*PI*fR);
13	getch();
14	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Dòng thứ 5: ***#define PI 3.14***, dùng chỉ thị define để định nghĩa hằng số PI có giá trị
3.14. *Trước define phải có dấu # và cuối dòng không có dấu chấm phẩy.*

2.2.2.4 Ví dụ 4

Dòng	File Edit Search Run Compile Debug Project Option Window Help
1	/* Chuong trinh nhap vao ban kinh hinh tron. Tinh dien tich */
2	#include <stdio.h>
3	#include <conio.h>
4	
5	#define PI 3.14
6	
7	void main(void)
8	{
9	float fR;
10	printf("Nhap vao ban kinh hinh tron: ");
11	scanf("%f", &fR);
12	printf("Dien tich hinh tron: %.2f.\n", 2*PI*fR);
13	getch();
14	}
	F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt - F9 Compile F9 Make F10 Menu

Dòng thứ 12: ***printf("chu vi hinh tron: %.2f.\n", 2*PI*fR);***. Hàm này có 2 đối mục. Đối mục thứ nhất là một chuỗi định dạng có chứa chuỗi văn bản ***Chu vi hinh tron:*** và ***%.2f*** (ký hiệu khai báo chuyển đổi dạng thức) cho biết dạng số chấm động sẽ được in ra, trong đó ***.2*** nghĩa là in ra với 2 số lẻ. Đối mục thứ 2 là biểu thức hằng ***2*PI*fR***;

Bài 3 :
CÁC THÀNH PHẦN TRONG NGÔN NGỮ C

3.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Khái niệm từ khóa
- Các kiểu dữ liệu
- Cách ghi chú
- Đặt tên biến
- Khai báo biến.
- Phạm vi sử dụng biến.

3.2 Nội dung

3.2.1 Từ khóa

Từ khóa là từ có ý nghĩa xác định dùng để khai báo dữ liệu, viết câu lệnh… Trong C có các từ khóa sau:

asm	const	else	for	interrupt	return	sizeof	void
break	continue	enum	goto	long	short	switch	volatile
case	default	extern	huge	near	static	typedef	while
cdecl	do	far	if	pascal	struct	union	
char	double	float	int	register	signed	unsigned	

☞ Các từ khóa phải viết bằng ***chữ thường***

3.2 Nội dung

3.2.2 Tên

Khái niệm tên rất quan trọng trong quá trình lập trình, nó không những thể hiện rõ ý nghĩa trong chương trình mà còn dùng để xác định các đại lượng khác nhau khi thực hiện chương trình. Tên thường được đặt cho hằng, biến, mảng, con trỏ, nhãn… Chiều dài tối đa của tên là 32 ký tự.

Tên biến hợp lệ là một chuỗi ký tự liên tục gồm: ***Ký tự chữ, số và dấu gạch dưới***. Ký tự đầu

của tên phải là ***chữ hoặc dấu gạch dưới***. Khi đặt tên không được đặt trùng với các từ khóa.

3.2 Nội dung

3.2.2 Tên

Ví dụ 1 :

Các tên đúng: delta, a_1, Num_ODD, Case

Các tên sai:

3a_1 (ký tự đầu là số)

num-odd (sử dụng dấu gạch ngang)

int (đặt tên trùng với từ khóa)

del ta (có khoảng trắng)

f(x) (có dấu ngoặc tròn)

Lưu ý: Trong C, tên phân biệt chữ hoa, chữ thường

Ví dụ 2 : number khác Number case khác Case

(case là từ khóa, do đó bạn đặt tên là Case vẫn đúng)

3.2.5 Khai báo biến

3.2.5.1 Tên biến

Cách đặt tên biến như mục 2.

3.2.5.2 Khai báo biến

Cú pháp

Kiểu dữ liệu *Danh sách tên biến*;

- Kiểu dữ liệu: 1 trong các kiểu ở mục 3

Danh sách tên biến: gồm các tên biến có cùng kiểu dữ liệu, mỗi tên biến cách nhau dấu phẩy (,), cuối cùng là dấu chấm phẩy (;).

- Khi khai báo biến nên đặt tên biến theo **quy tắc Hungarian Notation**

Ví dụ 4 :

int ituoi; //khai báo biến ituoi có kiểu int

float fTrongluong; //khai báo biến fTrongluong có kiểu long

char ckitu1, ckitu2;//khai báo biến ckitu1, ckitu2 có kiểu char

Các biến khai báo trên theo quy tắc Hungarian Notation. Nghĩa là biến ituoi là kiểu int, bạn thêm chữ i (kí tự đầu của kiểu) vào đầu tên biến tuoi để trong quá trình lập trình hoặc sau này xem lại, sửa chữa… bạn dễ dàng nhận ra biến ituoi có kiểu int mà không cần phải di chuyển đến phần khai báo mới biết kiể.u của biến này. Tương tự cho biến fTrongluong, bạn nhìn vào là biết ngay biến này có kiểu float.

Có thể kết hợp việc khai báo với toán tử gán để biến nhận ngay giá trị cùng lúc với khai báo.

Ví dụ 5 :

Khai báo trước, gán giá trị sau:

```
void main()
{
int a, b, c;
a = 1; b = 2; c = 5;
…
}
```

Vừa khai báo vừa gán giá trị:

```
void main()
{
int a = 1, b = 2, c = 5;
```

3.2.5.4 Phạm vi của biến

Khi lập trình, bạn phải nắm rõ phạm vi của biến. Nếu khai báo và sử dụng không đúng, không rõ ràng sẽ dẫn đến sai sót khó kiểm soát được, vì vậy bạn cần phải xác định đúng vị trí, phạm vi sử dụng biến trước khi sử dụng biến.

- Khai báo biến ngoài (biến toàn cục): Vị trí biến đặt bên ngoài tất cả các hàm, cấu trúc... Các biến này có ảnh hưởng đến toàn bộ chương trình. Chu trình sống của nó là bắt đầu chạy chương trình đến lúc kết thúc chương trình.

- Khai báo biến trong (biến cục bộ): Vị trí biến đặt bên trong hàm, cấu trúc…. Chỉ ảnh hưởng nội bộ bên trong hàm, cấu trúc đó…. Chu trình sống của nó bắt đầu từ lúc hàm, cấu trúc được gọi thực hiện đến lúc thực hiện xong.

Bài 4 :
NHẬP / XUẤT DỮ LIỆU

4.1 Mục tiêu

Sau khi hoàn tất bài này học viên sẽ hiểu và vận dụng các kiến thức kĩ năng cơ bản sau:

- Ý nghĩa, cách sử dụng hàm printf, scanf

- Sử dụng khuôn dạng, ký tự đặc biệt, ký tự điều khiển trong printf, scanf.

4.2 Nội dung

4.2.1 Hàm printf

Kết xuất dữ liệu được định dạng.

Cú pháp

printf ("chuỗi định dạng"[, đối mục 1, đối mục 2,…]);

-*Khi sử dụng hàm phải khai báo tiền xử lý*

#include <stdio.h>

- printf: tên hàm, ***phải viết bằng chữ thường.***

- đối mục 1,…: là các mục dữ kiện cần in ra màn hình. Các đối mục này có thể là biến, hằng hoặc biểu thức phải được định trị trước khi in ra.

4.2 Nội dung

4.2.1 Hàm printf

- chuỗi định dạng: được đặt trong cặp nháy kép (" "), gồm 3 loại:

+ Đối với chuỗi kí tự ghi như thế nào in ra giống như vậy.

+ Đối với những kí tự chuyển đổi dạng thức cho phép kết xuất giá trị của các đối mục

ra màn hình tạm gọi là mã định dạng. Sau đây là các dấu mô tả định dạng:

%c : Ký tự đơn

%s : Chuỗi

%d : Số nguyên thập phân có dấu

%f : Số chấm động (ký hiệu thập phân)

4.2 Nội dung

4.2.1 Hàm printf

%e : Số chấm động (ký hiệu có số mũ)

%g : Số chấm động (%f hay %g)

%x : Số nguyên thập phân không dấu

%u : Số nguyên hex không dấu

%o : Số nguyên bát phân không dấu

1: Tiền tố dùng kèm với %d, %u, %x, %o để chỉ số nguyên dài (ví dụ %ld)

4.2 Nội dung

4.2.1 Hàm printf

+ Các ký tự điều khiển và ký tự đặc biệt

\n : Nhảy xuống dòng kế tiếp canh về cột đầu tiên.

\t : Canh cột tab ngang.

\r : Nhảy về đầu hàng, không xuống hàng.

\a : Tiếng kêu bip.

\\ : In ra dấu \

\" : In ra dấu "

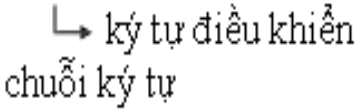
\' : In ra dấu '

%%: In ra dấu %

4.2 Nội dung

4.2.1 Hàm printf

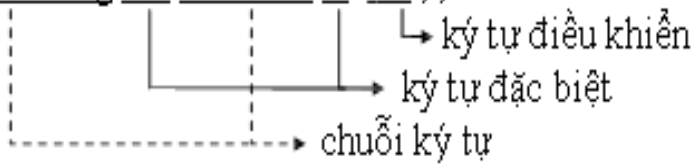
Vi du1: printf("Bai hoc C dau tien. \n");



☞ ***Kết quả in ra màn hình***

```
Bai hoc C dau tien.
```

Vi du2: printf("Ma dinh dang \\\" in ra dau \" .\n");



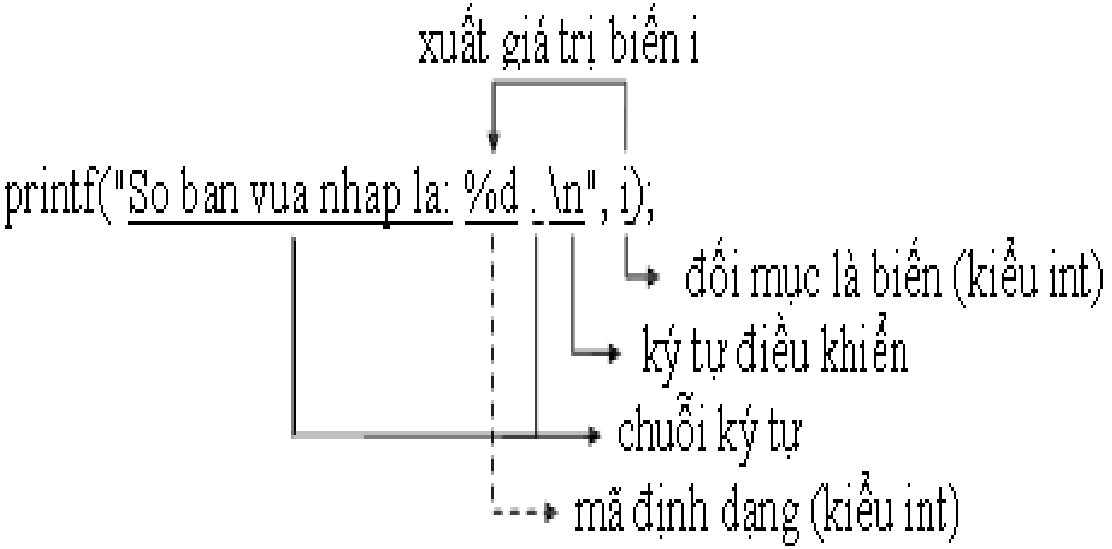
☞ ***Kết quả in ra màn hình***

```
Ma dinh dang \" in ra dau ".
```

4.2 Nội dung

4.2.1 Hàm printf

Vi du3: giả sử biến i có giá trị = 5



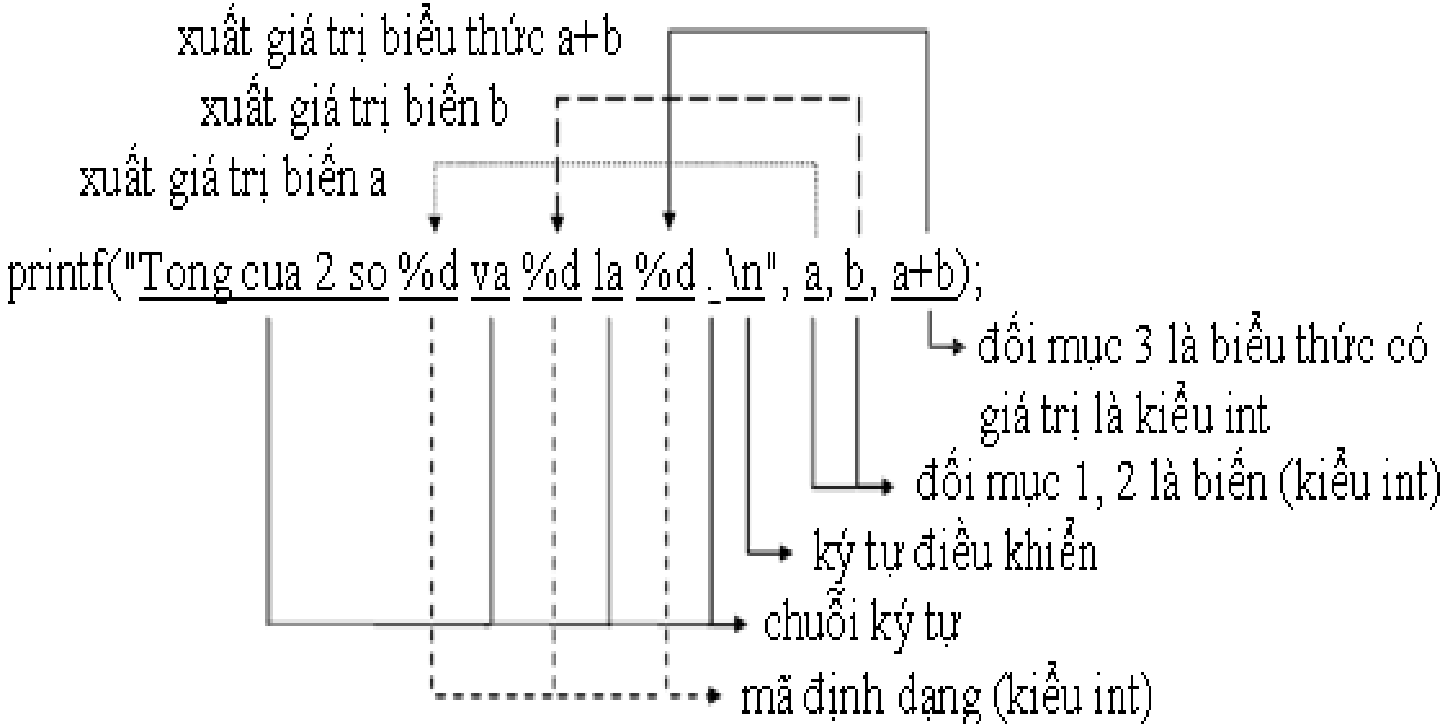
☞ ***Kết quả in ra màn hình***

```
So ban vua nhap la: 5.
```

4.2 Nội dung

4.2.1 Hàm printf

Ví dụ4: giả sử biến a có giá trị = 7 và b có giá trị = 4



☞ ***Kết quả in ra màn hình***

```
Tong cua 2 so 7 va 4 la 11.
```

4.2 Nội dung

4.2.1 Hàm printf

Ví dụ5: sửa lại ví dụ 4

```
printf("Tong cua 2 so %5d va %3d la %1d . \n", a, b, a+b);
```

Bề rộng trường

- ***Kết quả in ra màn hình***

```
Tong cua 2 so    7 va   4 la 11.
_
```

2 kí tự (mặc dù định dạng là 1)

3 kí tự

5 kí tự

4.2 Nội dung

4.2.1 Hàm printf

Vi du6: sửa lại ví dụ 5

```
printf("Tong cua 2 so %-5d va %-3d la %-1d . \n", a, b, a+b);
```

☞ ***Kết quả in ra màn hình***

```
Tong cua 2 so 7    va 4   la 11.
_
```

2 kí tự (mặc dù định dạng là 1)
3 kí tự
5 kí tự

☞ **Dấu trừ trước cb ể rộng trường sẽ kéo kết quả sang trái**

Vi du7: sửa lại ví dụ 4

```
printf("Tong cua 2 so %02d va %02d la %04d . \n", a, b, a+b);
```

☞ ***Kết quả in ra màn hình***

```
Tong cua 2 so 07 va 04 la 0011.
_
```

thêm 2 số 0 trước -> đủ 4 kí tự
thêm 1 số 0 trước -> đủ 2 kí tự
thêm 1 số 0 trước -> đủ 2 kí tự

4.2 Nội dung

4.2.1 Hàm printf

Ví dụ: giả sử int a = 6, b = 1234, c = 62

```
printf("%7d%7d%7d.\n", a, b, c);
printf("%7d%7d%7d.\n", 165, 2, 965);
```

☞ ***Kết quả in ra màn hình***

`   6  1234    62` `165     2   965`	Số canh về bên phải bề rộng trường.

```
printf("%-7d%-7d%-7d.\n", a, b, c);
printf("%-7d%-7d%-7d.\n", 165, 2, 965);
```

☞ ***Kết quả in ra màn hình***

`6      1234   62`
`165    2      965`
`_`
	Số canh về bên trái bề rộng trường.

4.2 Nội dung

4.2.1 Hàm printf

Ví dụ9: giả sử float a = 6.4, b = 1234.56, c = 62.3

```
printf("%7.2d%7.2d%7.2d.\n", a, b, c);
```

số số lẻ

☞ ***Kết quả in ra màn hình***

6.40 1234.56 62.30	Số canh về bên phải bề rộng trường.

7 kí tự

☞ **Bề rộng trường bao gồm: phần nguyên, phần lẻ và dấu chấm động**

Ví dụ10: giả sử float a = 6.4, b = 1234.55, c = 62.34

```
printf("%10.1d%10.1d%10.1d.\n", a, b, c);
printf("%10.1d%10.1d%10.1d.\n", 165, 2, 965);
```

☞ ***Kết quả in ra màn hình***

6.4 1234.6 62.3 165.0 2.0 965.0	Số canh về bên phải bề rộng trường.

4.2 Nội dung

4.2.2 Hàm scanf

Định dạng khi nhập liệu.

Cú pháp

scanf ("chuỗi định dạng"[, đối mục 1, đối mục 2,…]);

-*Khi sử dụng hàm phải khai báo tiền xử lý*

#include <stdio.h>

- scanf: tên hàm, ***phải viết bằng chữ thường.***

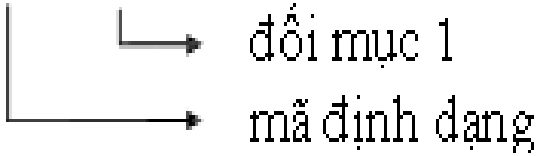
- khung định dạng: được đặt trong cặp nháy kép (" ") là hình ảnh dạng dữ liệu nhập vào.

- đối mục 1,…: là danh sách các đối mục cách nhau bởi dấu phẩy, mỗi đối mục sẽ tiếp nhận giá trị nhập vào.

4.2 Nội dung

4.2.2 Hàm scanf

Vi du11: scanf("%d", &i);



☞ Nhập vào 12abc, biến i chỉ nhận giá trị 12. Nhập 3.4 chỉ nhận giá trị 3.

4.2 Nội dung

4.2.2 Hàm scanf

Vi du12: scanf("%d%d", &a, &b);

☞ Nhập vào 2 số a, b phải cách nhau bằng **khoảng trắng** hoặc **enter**.

Vi du13: scanf("%d/%d/%d", &ngay, &thang, &nam);

☞ Nhập vào ngày, tháng, năm theo dạng ngay/thang/nam (20/12/2002)

Vi du14: scanf("%d%*c%d%*c%d", &ngay, &thang, &nam);

☞ Nhập vào ngày, tháng, năm với dấu phân cách /, -,…; ngoại trừ số.

Vi du15: scanf("%2d%2d%4d", &ngay, &thang, &nam);

☞ Nhập vào ngày, tháng, năm theo dạng dd/mm/yyyy.

4.3 Bài tập

1. Viết chương trình đổi một số nguyên hệ 10 sang hệ 2.

2. Viết chương trình đổi một số nguyên hệ 10 sang hệ 16.

3. Viết chương trình đọc và 2 số nguyên và in ra kết quả của phép (+), phép trừ (-), phép nhân (*), phép chia (/). Nhận xét kết quả chia 2 số nguyên.

4. Viết chương trình nhập vào bán kính hình cầu, tính và in ra diện tích, thể tích của

hình cầu đó.

Hướng dẫn: $S = 4\pi R^2$ và $V = (4/3)\pi R^3$.

4.3 Bài tập

5. Viết chương trình nhập vào một số a bất kỳ và in ra giá trị bình phương (a2), lập phương (a3) của a và giá trị a4.

6. Viết chương trình đọc từ bàn phím 3 số nguyên biểu diễn ngày, tháng, năm và xuất

ra màn hình dưới dạng "ngay/thang/nam" (chỉ lấy 2 số cuối của năm).

7. Viết chương trình nhập vào số giây từ 0 đến 86399, đổi số giây nhập vào thành dạng

"gio:phut:giay", mỗi thành phần là một số nguyên có 2 chữ số.

Ví dụ: 02:11:05