TRƯỜNG CAO ĐẲNG NGHỀ ĐẮK LẮK
KHOA ĐIỆN TỬ TIN HỌC
----------oOo----------

GIÁO TRÌNH

LẬP TRÌNH CƠ BẢN

NGHỀ: CÔNG NGHỆ THÔNG TIN
TRÌNH ĐỘ: CAO ĐẲNG NGHỀ, TRUNG CẤP NGHỀ

Người biên soạn:
Chủ biên: Nguyễn Thị Thu Hà
Đồng chủ biên: Lê Văn Tùng

Lưu hành nội bộ - 2015

Lời nói đầu

Tin học là một ngành khoa học mũi nhọn phát triển hết sức nhanh chóng trong vài chục năm lại đây và ngày càng mở rộng lĩnh vực nghiên cứu, ứng dụng trong mọi mặt của đời sống xã hội.

Ngôn ngữ lập trình là một loại công cụ giúp con người thể hiện các vấn đề của thực tế lên máy tính một cách hữu hiệu. Với sự phát triển của tin học, các ngôn ngữ lập trình cũng dần tiến hoá để đáp ứng các thách thức mới của thực tế.

C là ngôn ngữ lập trình vạn năng. Ngoài việc C được dùng để viết hệ điều hành UNIX, người ta nhanh chóng nhận ra sức mạnh của C trong việc xử lý cho các vấn đề hiện đại của tin học. C không gắn với bất kỳ một hệ điều hành hay máy nào, và mặc dầu nó đã được gọi là " ngôn ngữ lập trình hệ thống" vì nó được dùng cho việc viết hệ điều hành, nó cũng tiện lợi cho cả việc viết các chương trình xử lý số, xử lý văn bản và cơ sở dữ liệu.

Vậy, rất mong được sự góp ý của bạn đọc để tài liệu này ngày càng được hoàn thiện hơn, chúng tôi xin chân thành cảm ơn.

Mục lục

CHƯƠNG TRÌNH MÔN HỌC LẬP TRÌNH CƠ BẢN

Mã số môn học : MH13
Thời gian môn học : 75h (Lý thuyết: 30h; Thực hành: 45h)

I. VỊ TRÍ, TÍNH CHẤT CỦA MÔN HỌC

- Vị trí của môn học : Môn học được bố trí sau khi học sinh học xong các môn học chung, trước các môn học/ mô-đun đào tạo chuyên môn nghề.
- Tính chất của môn học : Là môn học lý thuyết chuyên ngành bắt buộc

II. MỤC TIÊU MÔN HỌC

Sau khi học xong môn học này học sinh có khả năng :

- Hiểu được công dụng của ngôn ngữ lập trình C, hiểu cú pháp, cụng dụng của các câu lệnh dùng trong ngôn ngữ lập trình C.
- Phân tích được chương trình: xác định nhiệm vụ chương trình (phải làm gì).
- Vận dụng điều kiện, trợ giúp môi trường của ngôn ngữ lập trình C, chẳng hạn: các thao tác biên tập chương trình, các công cụ, điều khiển, thực đơn lệnh trợ giúp, gỡ rối, bẫy lỗi,v.v.
- Viết chương trình và thực hiện chương trình trong máy tính.

III. NỘI DUNG MÔN HỌC:

1. Nội dung tổng quát và phân phối thời gian:

Số TT	Tên chương mục	Thời gian			
		Tổng số	Lý thuyết	Thực hành	Kiểm tra*
I	Tổng quan về ngôn ngữ lập trình C	2	2		
II	Các thành phần cơ bản	10	5	5	
III	Các lệnh cấu trúc	18	7	11	*
IV	Hàm	12	4	8	*
V	Mảng	12	4	8	
VI	Con trỏ	12	4	8	*
VII	Chuỗi ký tự	9	4	5	*
	Cộng	75	30	45	

chương 1: TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH

Mục tiêu:

- Hiểu được lịch sử phát triển của ngôn ngữ C;
- Biết được những ứng dụng thực tế của ngôn ngữ lập trình C;
- Thao tác thành thạo trên môi trường phát triển C;
- Sử dụng được hệ thống trợ giúp của phần mềm C;
- Chủ động tìm hiểu các tính năng của ngôn ngữ C.

1. Giới thiệu của ngôn ngữ C:

1.1 Lịch sử phát triển:

C là ngôn ngữ lập trình cấp cao, được sử dụng rất phổ biến để lập trình hệ thống cùng với Assembler và phát triển các ứng dụng.

Vào những năm cuối thập kỷ 60 đầu thập kỷ 70 của thế kỷ XX, Dennish Ritchie (làm việc tại phòng thí nghiệm Bell) đã phát triển ngôn ngữ lập trình C dựa trên ngôn ngữ BCPL (do Martin Richards đưa ra vào năm 1967) và ngôn ngữ B (do Ken Thompson phát triển từ ngôn ngữ BCPL vào năm 1970 khi viết hệ điều hành UNIX đầu tiên trên máy PDP-7) và được cài đặt lần đầu tiên trên hệ điều hành UNIX của máy DEC PDP-11.

Năm 1978, Dennish Ritchie và B.W Kernighan đã cho xuất bản quyển "Ngôn ngữ lập trình C" và được phổ biến rộng rãi đến nay.

Lúc ban đầu, C được thiết kế nhằm lập trình trong môi trường của hệ điều hành Unix nhằm mục đích hỗ trợ cho các công việc lập trình phức tạp. Nhưng về sau, với những nhu cầu phát triển ngày một tăng của công việc lập trình, C đã vượt qua khuôn khổ của phòng thí nghiệm Bell và nhanh chóng hội nhập vào thế giới lập trình để rồi các công ty lập trình sử dụng một cách rộng rãi. Sau đó, các công ty sản xuất phần mềm lần lượt đưa ra các phiên bản hỗ trợ cho việc lập trình bằng ngôn ngữ C và chuẩn ANSI C cũng được khai sinh từ đó.

Ngôn ngữ lập trình C là một ngôn ngữ lập trình hệ thống rất mạnh và rất "mềm

dẻo", có một thư viện gồm rất nhiều các hàm (function) đã được tạo sẵn. Người lập trình có thể tận dụng các hàm này để giải quyết các bài toán mà không cần phải tạo mới. Hơn thế nữa, ngôn ngữ C hỗ trợ rất nhiều phép toán nên phù hợp cho việc giải quyết các bài toán kỹ thuật có nhiều công thức phức tạp. Ngoài ra, C cũng cho phép người lập trình tự định nghĩa thêm các kiểu dữ liệu trừu tượng khác. Tuy nhiên, điều mà người mới vừa học lập trình C thường gặp "rắc rối" là "hơi khó hiểu" do sự "mềm dẻo" của C. Dù vậy, C được phổ biến khá rộng rãi và đã trở thành một công cụ lập trình khá mạnh, được sử dụng như là một ngôn ngữ lập trình chủ yếu trong việc xây dựng những phần mềm hiện nay.

1.2 Sự cần thiết:

Ngôn ngữ C có những đặc điểm cơ bản sau:

o Tính cô đọng (compact): C chỉ có 32 từ khóa chuẩn và 40 toán tử chuẩn, nhưng hầu hết đều được biểu diễn bằng những chuỗi ký tự ngắn gọn.

o Tính cấu trúc (structured): C có một tập hợp những chỉ thị của lập trình như cấu trúc lựa chọn, lặp… Từ đó các chương trình viết bằng C được tổ chức rõ ràng, dễ hiểu.

o Tính tương thích (compatible): C có bộ tiền xử lý và một thư viện chuẩn vô cùng phong phú nên khi chuyển từ máy tính này sang máy tính khác các chương trình viết bằng C vẫn hoàn toàn tương thích.

o Tính linh động (flexible): C là một ngôn ngữ rất uyển chuyển và cú pháp, chấp nhận nhiều cách thể hiện, có thể thu gọn kích thước của các mã lệnh làm chương trình chạy nhanh hơn.

o Biên dịch (compile): C cho phép biên dịch nhiều tập tin chương trình riêng rẽ thành các tập tin đối tượng (object) và liên kết (link) các đối tượng đó lại với nhau thành một chương trình có thể thực thi được (executable) thống nhất.

2. Các thao tác cơ bản:

2.1 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)

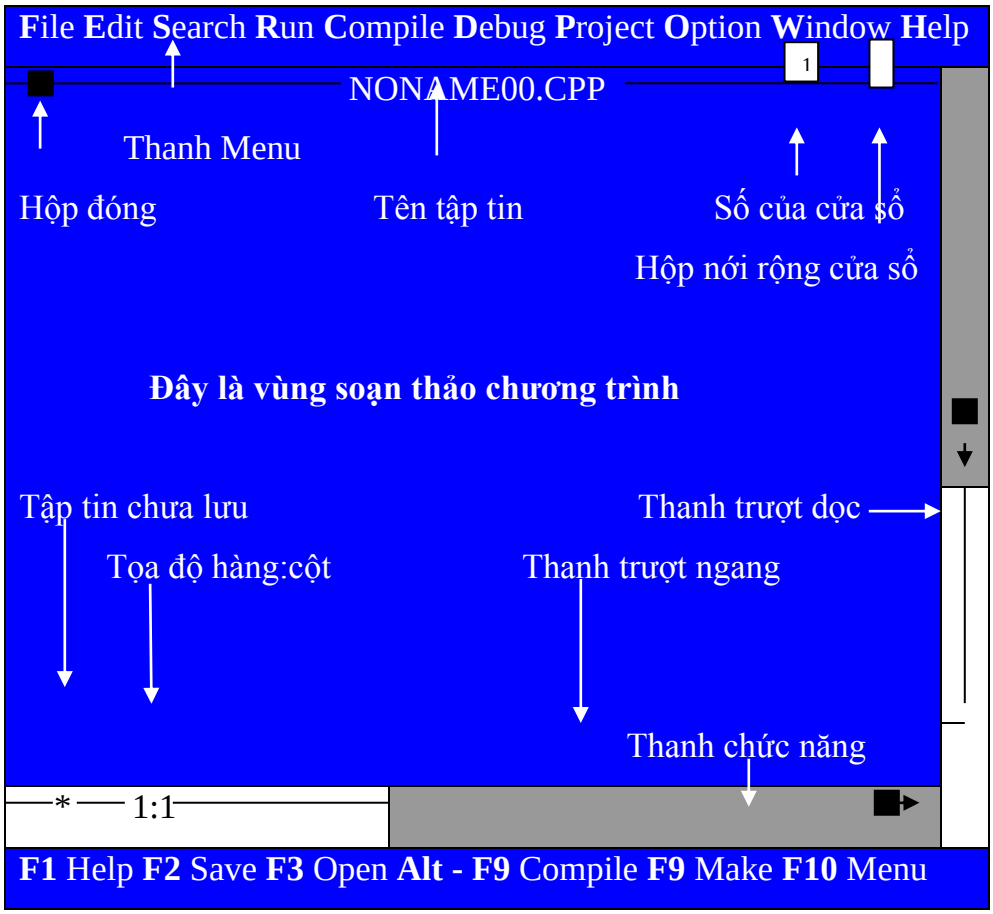
Nếu bạn muốn vừa khởi động BC vừa soạn thảo chương trình với một tập tin có tên do chúng ta đặt, thì gõ lệnh: **BC [đường dẫn]<tên file cần soạn thảo>**, nếu tên file cần soạn thảo đã có thì được nạp lên, nếu chưa có sẽ được tạo mới.

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.

Ví dụ: Bạn gõ D:\BORLANDC\BIN\BC E:\BAITAP_BC\VIDU1.CPP

Câu lệnh trên có nghĩa khởi động BC và nạp tập tin VIDU1.CPP chứa trong thư mụcBAITAP_BC trong ổ đĩa E. Nếu tập tin này không có sẽ được tạo mới.

2.2 Màn hình sau khi khởi động thành công



2.3 Thoát khỏi Turbo C và trở về DOS (hay Windows)

Dùng File/Exit hoặc Alt-X.

2.4. Tạo mới, ghi một chương trình C

a, Soạn thảo chương trình mới

Muốn soạn thảo một chương trình mới ta chọn mục New trong menu File (File ->New)

b,Ghi chương trình đang soạn thảo vào đĩa

Sử dụng File/Save hoặc gõ phím F2. Có hai trường hợp xảy ra:

- Nếu chương trình chưa được ghi lần nào thì một hội thoại sẽ xuất hiện cho phép bạn xác định tên tập tin (FileName). Tên tập tin phải tuân thủ quy cách đặt tên của DOS và không cần có phần mở rộng (sẽ tự động có phần mở rộng là .C hoặc .CPP sẽ nói thêm trong phần Option). Sau đó gõ phím Enter.
- Nếu chương trình đã được ghi một lần rồi thì nó sẽ ghi những thay đổi bổ sung lên tập tin chương trình cũ.

Chú ý: Để đề phòng mất điện trong khi soạn thảo chương trinh thỉnh thoảng bạn nên gõ phím F2.

Quy tắc đặt tên tập tin của DOS: Tên của tập tin gồm 2 phần: Phần tên và phần mở rộng.

o Phần tên của tập tin phải bắt đầu là 1 ký tự từ a..z (không phân biệt hoa thường), theo sau có thể là các ký tự từ a..z, các ký số từ 0..9 hay dấu gạch dưới (_), phần này dài tối đa là 8 ký tự.

o Phần mở rộng: phần này dài tối đa 3 ký tự.

Ví dụ: Ghi chương trình vừa soạn thảo trên lên đĩa với tên là CHAO.C

3. **Sử dụng trợ giúp:** Ví dụ đơn giản

Dòng	**F**ile **E**dit **S**earch **R**un **C**ompile **D**ebug **P**roject **O**ption **W**indow **H**elp
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 ra màn hình là:

Bai hoc C dau tien.

- giải thích:

➢ 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.

➢ 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.

➢ 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.

➢ 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.

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 '

4. Bài tập

Xác định Input, Process, Output của 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

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

Chương 2 : CÁC THÀNH PHẦN CƠ BẢN

Mục tiêu:

- Hiểu được các khái niệm: tập kí tự, từ khóa, kiểu dữ liệu, biến, hằng, biểu thức, câu lệnh, khối lệnh trong ngôn ngữ C;
- Hiểu được cấu trúc, phương thức thực thi của một chương trình C;
- Viết được một số đoạn chương trình cụ thể để thực hiện các yêu cầu đơn giản;
- Nghiêm túc, tỉ mỉ trong việc tiếp nhận các kiến thức và áp dụng vào thực hành.

1.Từ khóa và ký hiệu:

1.1 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 break case cdecl char const for goto huge if int interrupt long near pascal register return short static struct signed sizeof switch typedef union unsigned void volatile while

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

1.2 Hệ thống các ký hiệu trong C:

Bộ chữ viết trong ngôn ngữ C bao gồm những ký tự, ký hiệu sau: (phân biệt chữ in hoa và in thường):

 26 chữ cái latinh lớn A,B,C...Z

 26 chữ cái latinh nhỏ a,b,c ...z.

 10 chữ số thập phân 0,1,2...9.

 Các ký hiệu toán học: +, -, *, /, =, <, >, (,)

f Các ký hiệu đặc biệt: :. , ; " ' _ @ # $! ^ [ ] { } ...

 Dấu cách hay khoảng trống.

1.3.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.

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)

2. Các Kiểu dữ liệu sơ cấp

Có 4 kiểu dữ liệu cơ bản trong C là: char, int, float, double.

TT	Kiểu dữ liệu (Type)	Kích thước (Length)	Miền giá trị (Range)
1	unsigned char	1 byte	0 đến 255
2	char	1 byte	– 128 đến 127
3	enum	2 bytes	– 32,768 đến 32,767
4	unsigned int	2 bytes	0 đến 65,535
5	short int	2 bytes	– 32,768 đến 32,767
6	int	2 bytes	– 32,768 đến 32,767

7	unsigned long	4 bytes	0 đến 4,294,967,295
8	long	4 bytes	– 2,147,483,648 đến 2,147,483,647
9	float	4 bytes	3.4 * 10–38 đến 3.4 * 1038
10	double	8 bytes	1.7 * 10–308 đến 1.7 * 10308
11	long double	10 bytes	3.4 * 10–4932 đến 1.1 * 104932

*. *Ghi chú*

Trong khi lập trình cần phải ghi chú để giải thích các biến, hằng, thao tác xử lý giúp cho chương trình rõ ràng dễ hiểu, dễ nhớ, dễ sửa chữa và để người khác đọc vào dễ hiểu.

Trong C có các ghi chú sau: // hoặc /* nội dung ghi chú */

Ví dụ 3 :

```
void main()
{
int a, b; //khai bao bien t kieu int
a = 1; //gan 1 cho a
b =3; //gan 3 cho b
/* thuat toan tim so lon nhat la
neu a lon hon b thi a lon nhat
nguoc lai b lon nhat */
if (a > b) printf("max: %d", a);
else printf("max: %d", b);
}
```

Khi biên dịch chương trình, C gặp cặp dấu ghi chú sẽ không dịch ra ngôn ngữ máy.

Tóm lại, đối với ghi chú dạng // dùng để ghi chú một hàng và dạng /* …. */ có thể ghi chú một hàng hoặc nhiều hàng.

3. biến, hằng, biểu thức:

3.1. Biến:

Biến là một đại lượng được người lập trình định nghĩa và được đặt tên thông

qua việc khai báo biến. Biến dùng để chứa dữ liệu trong quá trình thực hiện chương trình và giá trị của biến có thể bị thay đổi trong quá trình này. Cách đặt tên biến giống như cách đặt tên đã nói trong phần trên.

Mỗi biến thuộc về một kiểu dữ liệu xác định và có giá trị thuộc kiểu đó.

VI.1.1. Cú pháp khai báo biến:

<Kiểu dữ liệu> Danh sách các tên biến cách nhau bởi dấu phẩy;

Ví dụ:

```
int   a, b, c;        /*Ba biến a, b,c có kiểu int*/
long int   chu_vi;    /*Biến chu_vi có kiểu long*/
float    nua_chu_vi;   /*Biến nua_chu_vi có kiểu float*/
double  dien_tich;    /*Biến dien_tich có kiểu double*/
```

Lưu ý: Để kết thúc 1 lệnh phải có dấu chấm phẩy (;) ở cuối lệnh.

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.**

3.2. Hằng:

Là đại lượng không đổi trong suốt quá trình thực thi của chương trình.

Hằng có thể là một chuỗi ký tự, một ký tự, một con số xác định. Chúng có thể được biểu diễn hay định dạng (Format) với nhiều dạng thức khác nhau.

V.2.1 Hằng số thực

Số thực bao gồm các giá trị kiểu float, double, long double được thể hiện theo 2 cách sau:

- Cách 1: Sử dụng cách viết thông thường mà chúng ta đã sử dụng trong các môn Toán, Lý, …Điều cần lưu ý là sử dụng dấu thập phân là dấu chấm (.);

Ví dụ: 123.34 -223.333 3.00 -56.0

- Cách 2: Sử dụng cách viết theo số mũ hay số khoa học. Một số thực được tách làm 2 phần, cách nhau bằng ký tự e hay E

Phần giá trị: là một số nguyên hay số thực được viết theo cách 1.

Phần mũ: là một số nguyên

Giá trị của số thực là: Phần giá trị nhân với 10 mũ phần mũ.

Ví dụ: 1234.56e-3 = 1.23456 (là số 1234.56 * 10-3

)

-123.45E4 = -1234500 (là -123.45 *104

)

V.2.2 Hằng số nguyên

Số nguyên gồm các kiểu int (2 bytes) , long (4 bytes) được thể hiện theo những cách sau.

- Hằng số nguyên 2 bytes (int) hệ thập phân: Là kiểu số mà chúng ta sử dụng thông thường, hệ thập phân sử dụng các ký số từ 0 đến 9 để biểu diễn một giá trị nguyên.

Ví dụ: 123 (một trăm hai mươi ba), -242 (trừ hai trăm bốn mươi hai).

- Hằng số nguyên 2 byte (int) hệ bát phân: Là kiểu số nguyên sử dụng 8 ký số từ 0 đến 7 để biểu diễn một số nguyên.

Cách biểu diễn: 0<các ký số từ 0 đến 7>

Ví dụ : 0345 (số 345 trong hệ bát phân)

-020 (số -20 trong hệ bát phân)

3.3. Biểu thức:

Biểu thức là một sự kết hợp giữa các toán tử (operator) và các toán hạng (operand) theo đúng một trật tự nhất định.

Mỗi toán hạng có thể là một hằng, một biến hoặc một biểu thức khác.

Trong trường hợp, biểu thức có nhiều toán tử, ta dùng cặp dấu ngoặc đơn () để chỉ định toán tử nào được thực hiện trước.

Ví dụ: Biểu thức nghiệm của phương trình bậc hai:

(-b + sqrt(Delta))/(2*a)

Trong đó 2 là hằng; a, b, Delta là biến.

a. các phép toán:

Trong ngôn ngữ C, các toán tử +, -, *, / làm việc tương tự như khi chúng làm việc trong các ngôn ngữ khác. Ta có thể áp dụng chúng cho đa số kiểu dữ liệu có sẵn được cho phép bởi C. Khi ta áp dụng phép / cho một số nguyên hay một ký tự, bất kỳ phần dư nào cũng bị cắt bỏ. Chẳng hạn, 5/2 bằng 2 trong phép chia nguyên.

Toán tử Ý nghĩa

+ Cộng

- Trừ

* Nhân

/ Chia

% Chia lấy phần dư

-- Giảm 1 đơn vị

++ Tăng 1 đơn vị

Tăng và giảm (++ & --)

Toán tử ++ thêm 1 vào toán hạng của nó và – trừ bớt 1. Nói cách khác:

x = x + 1 giống như ++x

x = x – 1 giống như x—

Cả 2 toán tử tăng và giảm đều có thể tiền tố (đặt trước) hay hậu tố (đặt sau) toán hạng. Ví dụ: x = x + 1 có thể viết x++ (hay ++x)

Tuy nhiên giữa tiền tố và hậu tố có sự khác biệt khi sử dụng trong 1 biểu thức. Khi 1 toán tử tăng hay giảm đứng trước toán hạng của nó, C thực hiện việc tăng hay giảm trước khi lấy giá trị dùng trong biểu thức. Nếu toán tử đi sau toán hạng, C lấy giá trị toán hạng trước khi tăng hay giảm nó. Tóm lại:

x = 10

y = ++x //y = 11

Tuy nhiên:

x = 10

x = x++ //y = 10

Thứ tự ưu tiên của các toán tử số học:

++ -- sau đó là * / % rồi mới đến + -

b. Các toán tử quan hệ và các toán tử Logic

Ý tưởng chính của toán tử quan hệ và toán tử Logic là đúng hoặc sai. Trong C mọi giá trị khác 0 được gọi là đúng, còn sai là 0. Các biểu thức sử dụng các toán tử quan hệ và Logic trả về 0 nếu sai và trả về 1 nếu đúng.

Toán tử Ý nghĩa

Các toán tử quan hệ

> Lớn hơn

>= Lớn hơn hoặc bằng

< Nhỏ hơn

<= Nhỏ hơn hoặc bằng

== Bằng

!= Khác

Các toán tử Logic

&& AND

|| OR

! NOT

4. Cấu trúc chương trình

4.1.Tiền xử lý và biên dịch

Tiền xử lý và biên dịch Trong C, việc dịch (translation) một tập tin nguồn được tiến hành trên hai bước hoàn toàn độc lập với nhau:

- Tiền xử lý.
- Biên dịch.

Hai bước này trong phần lớn thời gian được nối tiếp với nhau một cách tự động

theo cách thức mà ta có ấn tượng rằng nó đã được thực hiện như là một xử lý duy nhất. Nói chung, ta thường nói đến việc tồn tại của một bộ tiền xử lý (preprocessor?) nhằm chỉ rõ chương trình thực hiện việc xử lý trước. Ngược lại, các thuật ngữ trình biên dịch hay sự biên dịch vẫn còn nhập nhằng bởi vì nó chỉ ra khi thì toàn bộ hai giai đoạn, khi thì lại là giai đoạn thứ hai.
Bước tiền xử lý tương ứng với việc cập nhật trong văn bản của chương trình nguồn, chủ yếu dựa trên việc diễn giải các mã lệnh rất đặc biệt gọi là các chỉ thị dẫn hướng của bộ tiền xử lý (destination directive of preprocessor); các chỉ thị này được nhận biết bởi chúng bắt đầu bằng ký hiệu (symbol) #.
Hai chỉ thị quan trọng nhất là:
- Chỉ thị sự gộp vào của các tập tin nguồn khác: #include
- Chỉ thị việc định nghĩa các macros hoặc ký hiệu: #define
Chỉ thị đầu tiên được sử dụng trước hết là nhằm gộp vào nội dung của các tập tin cần có (header file), không thể thiếu trong việc sử dụng một cách tốt nhất các hàm của thư viện chuẩn, phổ biến nhất là:
#include <stdio.h>

4.2. Cấu trúc một chương trình C

Một chương trình C bao gồm các phần như: Các chỉ thị tiền xử lý, khai báo biến ngoài, các hàm tự tạo, chương trình chính (hàm main).
Cấu trúc có thể như sau:
Các chỉ thị tiền xử lý (Preprocessor directives)
#include <Tên tập tin thư viện>
#define ….
Định nghĩa kiểu dữ liệu (phần này không bắt buộc): dùng để đặt tên lại cho một kiểu dữ liệu nào đó để gợi nhớ hay đặt 1 kiểu dữ liệu cho riêng mình dựa trên các kiểu dữ liệu đã có.
Cú pháp: typedef <Tên kiểu cũ> <Tên kiểu mới>
Ví dụ: typedef int SoNguyen; // Kiểu SoNguyen là kiểu int
Khai báo các prototype (tên hàm, các tham số, kiểu kết quả trả về,… của các hàm sẽ cài đặt trong phần sau, phần này không bắt buộc): phần này chỉ là các khai báo đầu hàm, không phải là phần định nghĩa hàm.
Khai báo các biến ngoài (các biến toàn cục) phần này không bắt buộc: phần này khai báo các biến toàn cục được sử dụng trong cả chương trình.
Chương trình chính phần này bắt buộc phải có
<Kiểu dữ liệu trả về> main()
{
Các khai báo cục bộ trong hàm main: Các khai báo này chỉ tồn tại trong hàm mà thôi, có thể là khai báo biến hay khai báo kiểu.
Các câu lệnh dùng để định nghĩa hàm main
return <kết quả trả về>; // Hàm phải trả về kết quả
}
Lưu ý: Một số tập tin header thường dùng:
Cài đặt các hàm
<Kiểu dữ liệu trả về> function1(các tham số)
{

Các khai báo cục bộ trong hàm.
Các câu lệnh dùng để định nghĩa hàm
return <kết quả trả về>;
}
…

Một chương trình C bắt đầu thực thi từ hàm main (thông thường là từ câu lệnh đầu tiên đến câu lệnh cuối cùng).

4.3 Các tập tin thư viện thông dụng

Đây là các tập tin chứa các hàm thông dụng khi lập trinh C, muốn sử dụng các hàm trong các tập tin header này thì phải khai báo #include <Tên tập tin> ở phần đầu của chương trình

1) stdio.h: Tập tin định nghĩa các hàm vào/ra chuẩn (standard input/output). Gồm các hàm in dữ liệu (printf()), nhập giá trị cho biến (scanf()), nhận ký tự từ bàn phím (getc()), in ký tự ra màn hình (putc()), nhận một dãy ký tự từ bàm phím (gets()), in chuỗi ký tự ra màn hình (puts()), xóa vùng đệm bàn phím (fflush()), fopen(), fclose(), fread(), fwrite(), getchar(), putchar(), getw(), putw()…

2) conio.h : Tập tin định nghĩa các hàm vào ra trong chế độ DOS (DOS console). Gồm các hàm clrscr(), getch(), getche(), getpass(), cgets(), cputs(), putch(), clreol(),…

3) math.h: Tập tin định nghĩa các hàm tính toán gồm các hàm abs(), sqrt(), log(). log10(), sin(), cos(), tan(), acos(), asin(), atan(), pow(), exp(),…

4) alloc.h: Tập tin định nghĩa các hàm liên quan đến việc quản lý bộ nhớ. Gồm các hàm calloc(), realloc(), malloc(), free(), farmalloc(), farcalloc(), farfree(), …

5) io.h: Tập tin định nghĩa các hàm vào ra cấp thấp. Gồm các hàm open(), _open(), read(), _read(), close(), _close(), creat(), _creat(), creatnew(), eof(), filelength(), lock(),…

6) graphics.h: Tập tin định nghĩacác hàm liên quan đến đồ họa. Gồm initgraph(), line(), circle(), putpixel(), getpixel(), setcolor(), …

Còn nhiều tập tin khác nữa.

5. Câu lệnh:

5.1 khái niệm:

Là một tác vụ, biểu thức, hàm, cấu trúc điều khiển…

Ví dụ 1:

```
x = x + 2;

printf("Day la mot lenh\n");
```