PHẦN 2

GIỚI THIỆU VỀ MỘT NGÔN NGỮ LẬP TRÌNH - NGÔN NGỮ LẬP TRÌNH C

Chương 1

GIỚI THIỆU VỀ NGÔN NGỮ C & MÔI TRƯỜNG TURBO C 3.0

Học xong chương này, sinh viên sẽ nắm được các vấn đề sau:

- Tổng quan về ngôn ngữ lập trình C.
- Môi trường làm việc và cách sử dụng Turbo C 3.0.

I. TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH C

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.

Ngôn ngữ C có những đặc điểm cơ bản sau:

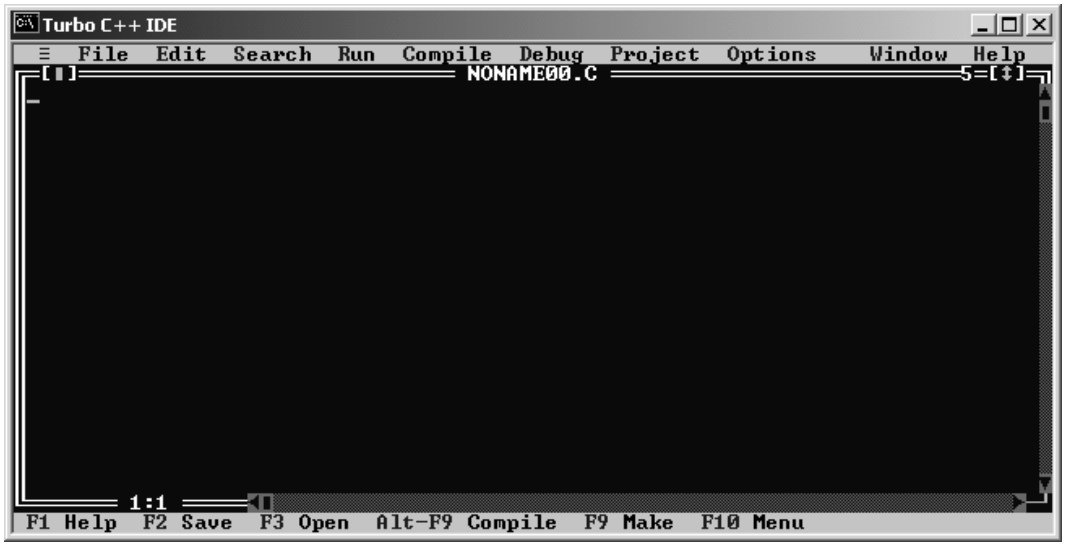
- *Tính cô đọng (compact)*: C chỉ có 32 từ khóa chuẩn và 40 toán tử chuẩn, nhưng hầu hết đều được biểu diễn bằng những chuỗi ký tự ngắn gọn.
- *Tính cấu trúc (structured)*: C có một tập hợp những chỉ thị của lập trình như cấu trúc lựa chọn, lặp… Từ đó các chương trình viết bằng C được tổ chức rõ ràng, dễ hiểu.
- *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.
- *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.
- *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.

II. MÔI TRƯỜNG LẬP TRÌNH TURBO C

Turbo C là môi trường hỗ trợ lập trình C do hãng Borland cung cấp. Môi trường này cung cấp các chức năng như: soạn thảo chương trình, dịch, thực thi chương trình… Phiên bản được sử dụng ở đây là Turbo C 3.0.

II.1. Gọi Turbo C

Chạy Turbo C cũng giống như chạy các chương trình khác trong môi trường DOS hay Windows, màn hình sẽ xuất hiện menu của Turbo C có dạng như sau:



Dòng trên cùng gọi là thanh menu (menu bar). Mỗi mục trên thanh menu lại có thể có nhiều mục con nằm trong một menu kéo xuống.

Dòng dưới cùng ghi chức năng của một số phím đặc biệt. Chẳng hạn khi gõ phím F1 thì ta có được một hệ thống trợ giúp mà ta có thể tham khảo nhiều thông tin bổ ích.

Muốn vào thanh menu ngang ta gõ phím F10. Sau đó dùng các phím mũi tên qua trái hoặc phải để di chuyển vùng sáng tới mục cần chọn rồi gõ phím Enter. Trong menu kéo xuống ta lại dùng các phím mũi tên lên xuống để di chuyển vùng sáng tới mục cần chọn rồi gõ Enter.

Ta cũng có thể chọn một mục trên thanh menu bằng cách giữ phím Alt và gõ vào một ký tự đại diện của mục đó (ký tự có màu sắc khác với các ký tự khác). Chẳng hạn để chọn mục File ta gõ Alt-F (F là ký tự đại diện của File)

II.2. Soạn thảo chương trình mới

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

Trên màn hình sẽ xuất hiện một vùng trống để cho ta soạn thảo nội dung của chương trình. Trong quá trình soạn thảo chương trình ta có thể sử dụng các phím sau:

Các phím xem thông tin trợ giúp:

- F1: Xem toàn bộ thông tin trong phần trợ giúp.

- Ctrl-F1: Trợ giúp theo ngữ cảnh (tức là khi con trỏ đang ở trong một từ nào đo, chẳng hạn int mà bạn gõ phím Ctrl-F1 thì bạn sẽ có được các thông tin về kiểu dữ liệu int)

Các phím di chuyển con trỏ trong vùng soạn thảo chương trình:

Phím	Ý nghĩa	Phím tắt (tổ hợp phím)
Enter	Đưa con trỏ xuống dòng	
Mũi tên đi lên	Đưa con trỏ lên hàng trước	Ctrl-E
Mũi tên đi xuống	Đưa con trỏ xuống hàng sau	Ctrl-X
Mũi tên sang trái	Đưa con trỏ sang trái một ký tự	Ctrl-S
Mũi tên sang phải	Đưa con trỏ sang phải một ký tự	Ctrl-D
End	Đưa con trỏ đến cuối dòng	
Home	Đưa con trỏ đến đầu dòng	
PgUp	Đưa con trỏ lên trang trước	Ctrl-R
PgDn	Đưa con trỏ xuống trang sau	Ctrl-C
	Đưa con trỏ sang từ bên trái	Ctrl-A
	Đưa con trỏ sang từ bên phải	Ctrl-F

Các phím xoá ký tự/ dòng:

Phím	Ý nghĩa	Phím tắt
Delete	Xoá ký tự tại ví trí con trỏ	Ctrl-G
BackSpace	Di chuyển sang trái đồng thời xoá ký tự đứng trước con trỏ	Ctrl-H
	Xoá một dòng chứa con trỏ	Ctrl-Y
	Xóa từ vị trí con trỏ đến cuối dòng	Ctrl-Q-Y
	Xóa ký tự bên phải con trỏ	Ctrl-T

Các phím chèn ký tự/ dòng:

Insert	Thay đổi viết xen hay viết chồng
Ctrl-N	Xen một dòng trống vào trước vị trí con trỏ

Sử dụng khối :

Khối là một đoạn văn bản chương trình hình chữ nhật được xác định bởi đầu khối là góc trên bên trái và cuối khối là góc dưới bên phải của hình chữ nhật. Khi một khối đã được xác định (trên màn hình khối có màu sắc khác chỗ bình thường) thì ta có

thể chép khối, di chuyển khối, xoá khối... Sử dụng khối cho phép chúng ta soạn thảo chương trình một cách nhanh chóng. sau đây là các thao tác trên khối:

Phím tắt	Ý nghĩa
Ctrl-K-B	Đánh dấu đầu khối
Ctrl-K-K	Đánh dấu cuối khối
Ctrl-K-C	Chép khối vào sau vị trí con trỏ
Ctrl-K-V	Chuyển khối tới sau vị trí con trỏ
Ctrl-K-Y	Xoá khối
Ctrl-K-W	Ghi khối vào đĩa như một tập tin
Ctrl-K-R	Đọc khối (tập tin) từ đĩa vào sau vị trí con trỏ
Ctrl-K-H	Tắt/mở khối
Ctrl-K-T	Đánh dấu từ chứa chon trỏ
Ctrl-K-P	In một khối

Các phím, phím tắt thực hiện các thao tác khác:

Phím	Ý nghĩa	Phím tắt
F10	Kích hoạt menu chính	Ctrl-K-D, Ctrl-K-Q
F2	Lưu chương trình đang soạn vào đĩa	Ctrl-K-S
F3	Tạo tập tin mới	
Tab	Di chuyển con trỏ một khoảng đồng thời đẩy dòng văn bản	Ctrl-I
ESC	Hủy bỏ thao tác lệnh	Ctrl-U
	Đóng tập tin hiện tại	Alt-F3
	Hiện hộp thoại tìm kiếm	Ctrl-Q-F
	Hiện hộp thoại tìm kiếm và thay thế	Ctrl-Q-A
	Tìm kiếm tiếp tục	Ctrl-L

Ví dụ: Bạn hãy gõ đoạn chương trình sau:

```
#include <stdio.h>
#include<conio.h>
int  main ()
{
      char ten[50];
      printf("Xin cho biet ten cua ban !");
      scanf("%s",ten);
      printf("Xin chao ban %s",ten);
      getch();
      return 0;
}
```

II.3. 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.*

- *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.*
- *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

II.4. Thực hiện chương trình

Để thực hiện chương trình hãy dùng Ctrl-F9 (giữ phím Ctrl và gõ phím F9).

Ví dụ: Thực hiện chương trình vừa soạn thảo xong và quan sát trên màn hình để thấy kết quả của việc thực thi chương trình sau đó gõ phím bất kỳ để trở lại với Turbo.

II.5. Mở một chương trình đã có trên đĩa

Với một chương trình đã có trên đĩa, ta có thể mở nó ra để thực hiện hoặc sửa chữa bổ sung. Để mở một chương trình ta dùng File/Open hoặc gõ phím F3. Sau đó gõ tên tập tin vào hộp File Name hoặc lựa chọn tập tin trong danh sách các tập tin rồi gõ Enter.

Ví dụ: Mở tập tin CHAO.C sau đó bổ sung để có chương trình mới như sau:

```
#include <stdio.h>
#include<conio.h>
int main ()
{
      char ten[50];
      printf("Xin cho biet ten cua ban !");
      scanf("%s",ten);
      printf("Xin chao ban %s\n ",ten);
      printf("Chao mung ban den voi Ngon ngu lap trinh C");
      getch();
      return 0;
}
```

Ghi lại chương trình này (F2) và cho thực hiện (Ctrl-F9). Hãy so sánh xem có gì khác trước?

II.6. Thoát khỏi Turbo C và trở về DOS (hay Windows)

Dùng File/Exit hoặc Alt-X.

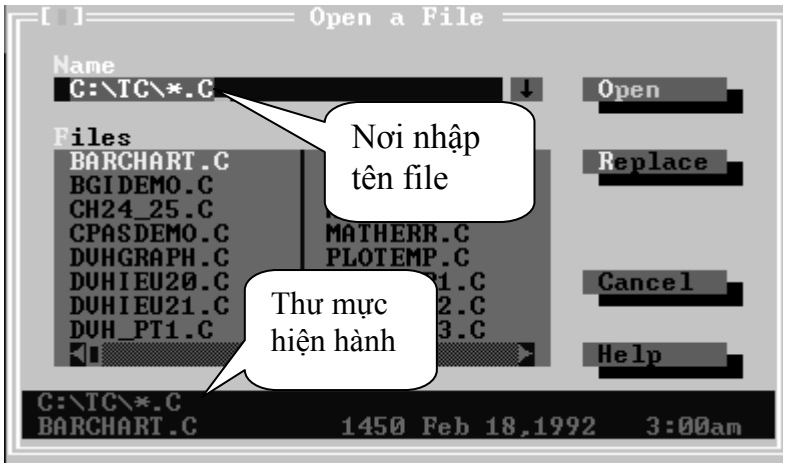
II.7. Sử dụng một số lệnh trên thanh menu

II.7.1. Các lệnh trên menu File (Alt -F)

- Lệnh New : Dùng để tạo mới một chương trình. Tên ngầm định của chương trình là NONAMEXX.C (XX là 2 số từ 00 đến 99).

- Lệnh Open : Dùng để mở một chương trình đã có sẵn trên đĩa để sửa chữa, bổ sung hoặc để thực hiện chương trình

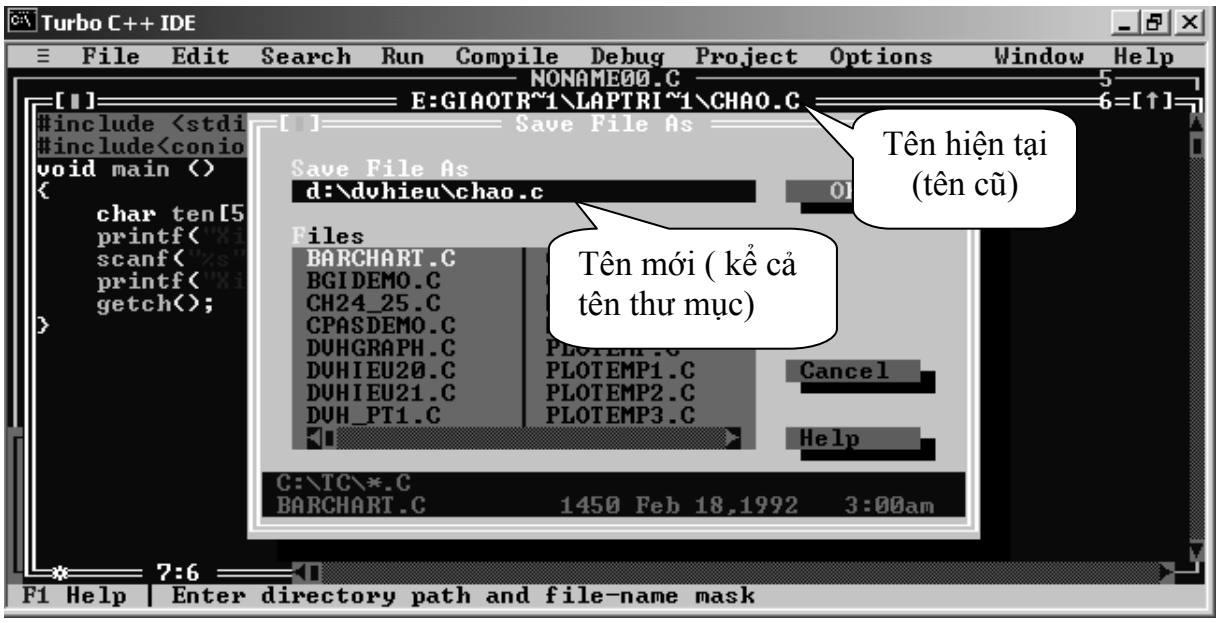
đó. Khi tập tin được mở thì văn bản chương trình được trình bày trong vùng soạn thảo; hộp thoại Open như sau:



Trong trường hợp ta nhập vào tên tập tin chưa tồn tại thì chương trình được tạo mới và sau này khi ta lưu trữ, chương trình được lưu với tên đó.

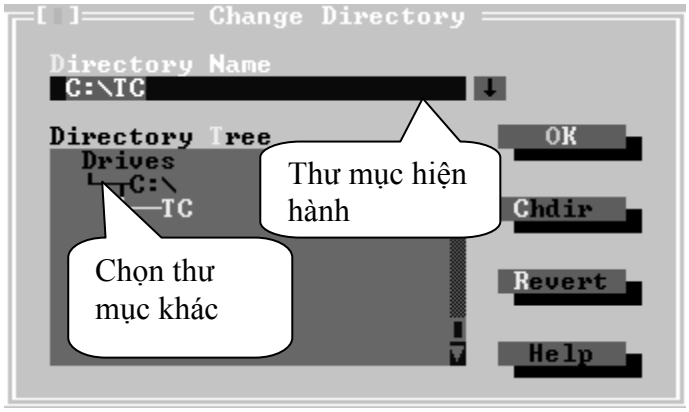
- Lệnh Save : Dùng để lưu chương trình đang soạn thảo vào đĩa.

- Lệnh Save as... : Dùng để lưu chương trình đang soạn thảo với tên khác, hộp thoại lưu tập tin đang soạn với tên khác như sau:



- Lệnh : Save All: Trong lúc làm việc với Turbo C, ta có thể mở một lúc nhiều chương trình để sửa chữa, bổ sung. Lệnh Save All dùng để lưu lại mọi thay đổi trên tất cả các chương trình đang mở ấy..

- Lệnh Change Dir ... : Dùng để đổi thư mục hiện hành



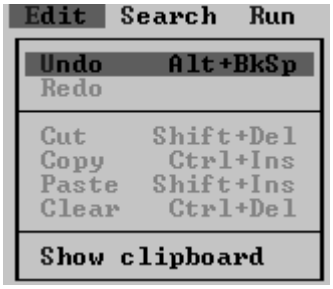
- Lệnh Print : Dùng để in chương trình đang soạn thảo ra máy in.

- Lệnh Printer Setup ...: Dùng để thiết đặt một số thông số cho máy in.

- Lệnh Dos Shell : Dùng để thoát tạm thời về Dos, để trở lại Turbo C ta đánh EXIT.

- Lệnh Exit : Dùng để thoát khỏi C.

II.7.2. Các lệnh trên menu Edit (Alt -E)



- Lệnh Undo : Dùng để hủy bỏ thao tác soạn thảo cuối cùng trên cửa số soạn thảo.

- Lệnh Redo : Dùng để phục hồi lại thao tác đã bị Undo cuối cùng.

- Lệnh Cut : Dùng để xóa một phần văn bản đã được đánh dấu khối, phần dữ liệu bị xóa sẽ được lưu vào một vùng nhớ *đặc biệt* gọi là Clipboard.

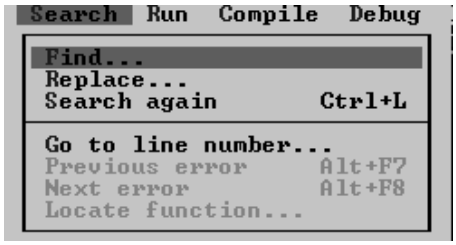
- Lệnh Copy : Dùng để chép phần chương trình đã được đánh dấu khối vào Clipboard.

- Lệnh Paste : Dùng để dán phần chương trình đang được lưu trong Clipboard vào của số đang soạn thảo, bắt đầu tại vị trí của con trỏ.

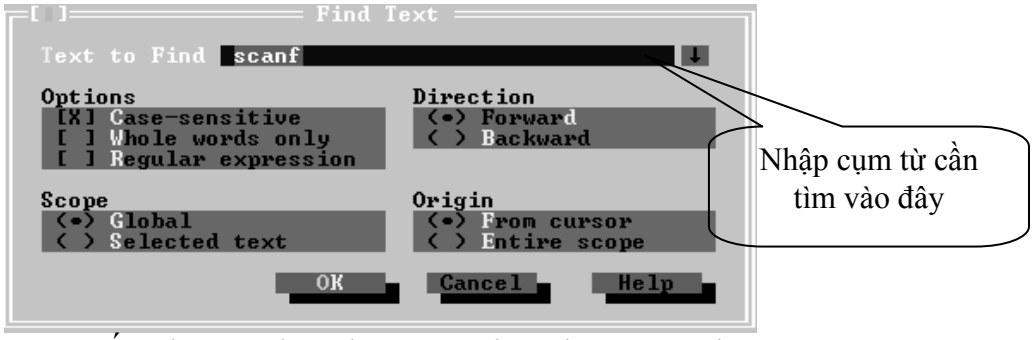
- Lệnh Clear : Dùng để xóa phần dữ liệu đã được đánh dấu khối, dữ liệu bị xóa không được lưu vào Clipboard.

- Lệnh Show clipboard : Dùng để hiển thị phần chương trình đang được lưu trong Clipboard trong một cửa sổ mới.

II.7.3. Các lệnh trên menu Search (Alt -S)



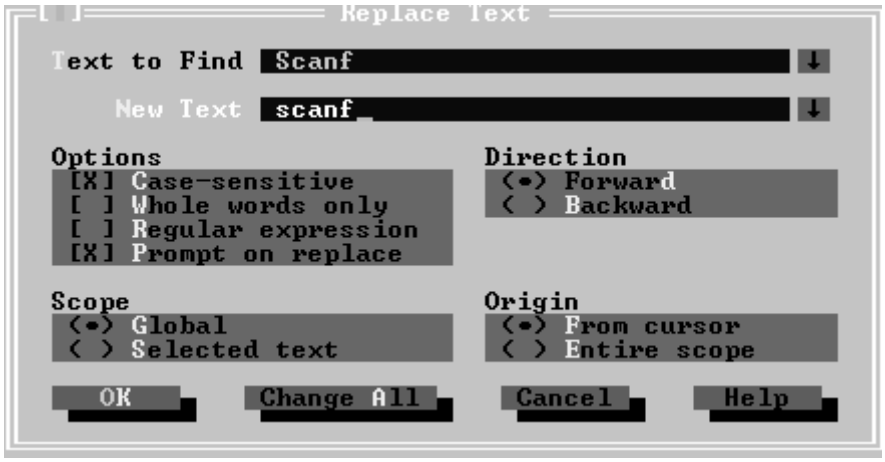
- Lệnh Find ...: Dùng để tìm kiếm một cụm từ trong văn bản chương trình. Nếu tìm thấy thì con trỏ sẽ di chuyển đến đoạn văn bản trùng với cụm từ cần tìm; hộp thoại Find như sau:



Ý nghĩa các lựa chọn trong hộp thoại trên như sau:

- Case sentitive : Phân biệt chữ IN HOA với chữ in thường trong khi so sánh cụm từ cần tìm với văn bản chương trình.
- Whole word only: Một đoạn văn bản chương trình trùng với toàn bộ cụm từ cần tìm thì mới được xem là tìm thấy.
- Regular expression: Tìm theo biểu thức
- Global:Tìm trên tất cả tập tin.
- Forward : Tìm đến cuối tập tin.
- Selected text: Chỉ tìm trong khối văn bản đã được đánh dấu.
- Backward: Tìm đến đầu tập tin.
- From cursor : Bắt đầu từ vị trí con nháy.
- Entire scope: Bắt đầu tại vị trí đầu tiên của khối hoặc tập tin.

- Lệnh Replace...: Dùng để tìm kiếm một đoạn văn bản nào đó, và tự động thay bằng một đoạn văn bản khác, hộp thoại replace như sau:

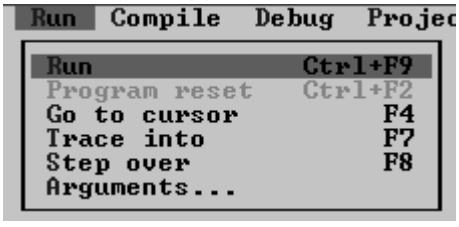


Tìm các cụm từ Scanf và thay thế bằng scanf

- Lệnh Search again : Dùng để thực hiện lại việc tìm kiếm.

- Các lệnh còn lại trên menu Search, các bạn sẽ tìm hiểu thêm khi thực hành trực tiếp trên máy tính.

II.7.4. Các lệnh trên menu Run (Alt -R)



- Lệnh Run : Dùng để thực thi hay "chạy" một chương trình.

- Lệnh Step over : Dùng để "chạy" chương trình từng bước.

- Lệnh Trace into : Dùng để chạy chương trình từng bước. *Khác với lệnh Step over ở chỗ:* Lệnh Step over không cho chúng ta xem từng bước "chạy" trong chương trình con, còn lệnh Trace into cho chúng ta xem từng bước trong chương trình con.

- Các lệnh còn lại, các bạn sẽ tìm hiểu thêm khi thực hành trên máy.

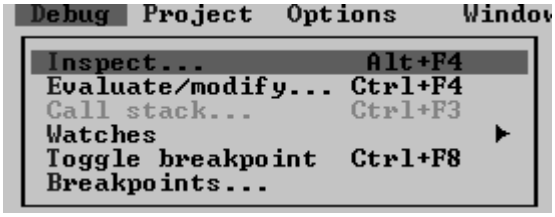
II.7.5. Các lệnh trên menu Compile (Alt C)

- Lệnh Complie: Biên dịch một chương trình.

- Lệnh Make , Build, … : Các lệnh này bạn sẽ tìm hiểu thêm khi thực hành trực tiếp trên máy tính.

- Lệnh Information : Dùng để hiện thị các thông tin về chương trình, Mode, môi trường .

II.7.6. Các lệnh trên menu Debug (Alt-D)



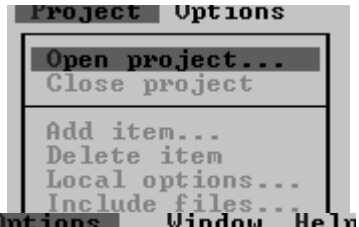
Trên menu Debug bao gồm một số lệnh giúp người lập trình "gỡ rối" chương trình . Người lập trình sử dụng chức năng "gỡ rối" khi gặp một số "lỗi" về thuật toán, sử dụng biến nhớ…

- Lệnh Breakpoints: Dùng để đặt "điểm dừng" trong chương trình. Khi chương trình thực thi đến "điểm dừng thì nó sẽ dừng lại" .

- Lệnh Watch : Dùng để mở một cửa sổ hiển thị kết quả trung gian của một biến nhớ nào đó khi chạy chương trình từng bước.

- Lệnh Evaluate/Modify: Bạn sẽ tìm hiểu khi thực hành trực tiếp trên máy.

II.7.7. Các lệnh trên menu Project (Alt- P)



Trên menu Project bao gồm các lệnh liên quan đến dự án như : đóng, mở, thêm , xóa các mục,…

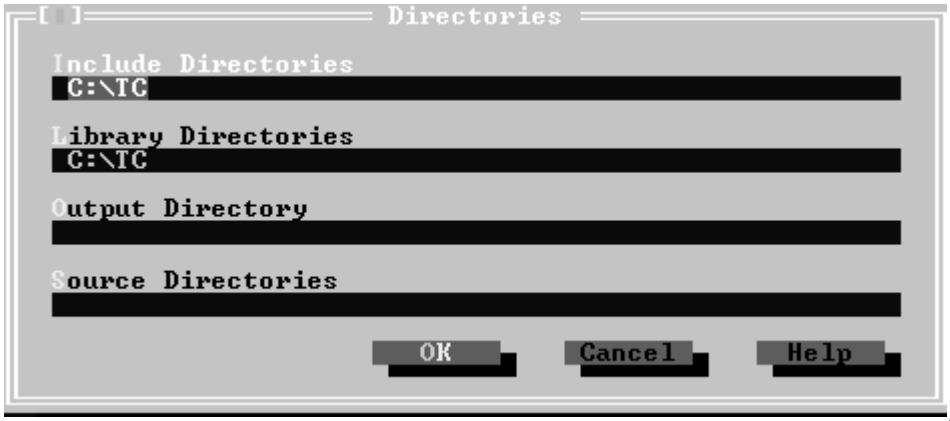
II.7.8. Các lệnh trên menu Option (Alt -O)

Trên menu Option bao gồm các lệnh giúp người lập trình thiết đặt một số tự chọn khi chạy chương trình. Thông thường, người lập trình không cần phải thiết đặt lại các tự chọn.

- Lệnh Compiler ...: Dùng để thiết đặt lại một số thông số khi biên dịch chương trình như hình sau

Phần trình bày dưới đây thuộc về 3 mục: Directories, Enviroment và Save; các phần khác sinh viên tự tìm hiểu.

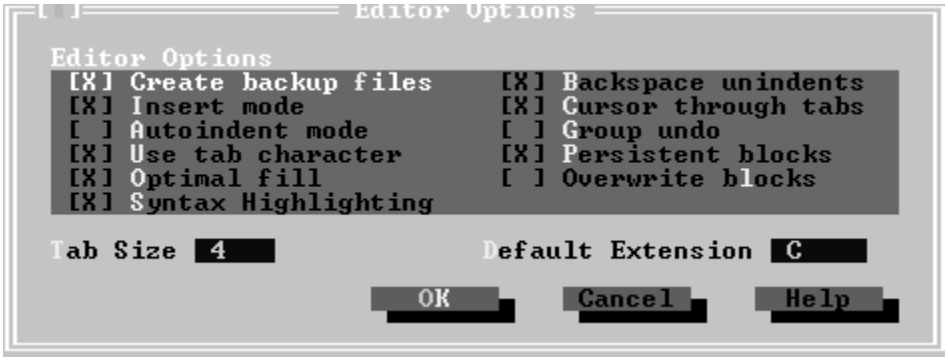
- Lệnh Directories...: Dùng để đặt lại đường dẫn tìm đến các tập tin cần thiết khi biên dịch chương trình như hình sau:



- Include directory: Thư mục chứa các tập tin mà chúng ta muốn đưa vào chương trình (các tập tin .h trong dòng #include).
- Library directory : Thư mục chứa các tập tin thư viện (các tập tin .Lib)
- Output directory: Thư mục chứa các tập tin "đối tượng " (có phần mở rộng là .OBJ), tập tin thực thi (.exe) khi biên dịch chương trình.
- Source directory: Thư mục chứa các tập tin "nguồn" (có phần mở rộng là .obj, .lib).

- Lệnh Environment: dùng để thiết lập môi trường làm việc như:

- Reference…: Các tham chiếu.
- Editor: Môi trường soạn thảo gồm: tạo tập tin dự phòng khi có sự chỉnh sửa (create backup file), chế độ viết đè (insert mode), tự động thụt đầu dòng (indent), đổi màu từ khóa (Syntax highlighting)… Đặc biệt, trong phần này là thiết lập phần mở rộng mặc định (Default Extension) của tập tin chương trình là C hay CPP (C Plus Plus: C++).



- Mouse...: Đặt chuột.
- Colors…: Đặt màu.

II.7.9. Các lệnh trên menu Window (Alt- W)

Trên menu Window bao gồm các lệnh thao tác đến cửa sổ như:

- Lệnh Cascade : Dùng để sắp xếp các cửa sổ.
- Lệnh Close all : Dùng để đóng tất cả các cửa sổ.
- Lệnh Zoom: Dùng để phóng to/ thu nhỏ cửa sổ.
- Các lệnh Tile, Refresh display, Size/ Move, Next, Previous, Close, List...: Các bạn sẽ tìm hiểu thêm khi thực hành trực tiếp trên máy tính.

II.7.10. Các lệnh trên menu Help (Alt- H)

Trên menu Help bao gồm các lệnh gọi trợ giúp khi người lập trình cần giúp đỡ một số vấn đề nào đó như: Cú pháp câu lệnh, cách sử dụng các hàm có sẵn…

- Lệnh Contents: Hiện thị toàn bộ nội dung của phần help.

- Lệnh Index : Hiển thị bảng tìm kiếm theo chỉ mục.
- Các lệnh còn lại, bạn sẽ tìm hiểu khi thực hành trên máy.

Chương 2

CÁC THÀNH PHẦN CƠ BẢN CỦA NGÔN NGỮ C

Học xong chương này, sinh viên sẽ nắm được các vấn đề sau:

- Bộ chữ viết trong C.
- Các từ khóa.
- Danh biểu.
- Các kiểu dữ liệu
- Biến và các biểu thức trong C.
- Cấu trúc của một chương trình viết bằng ngôn ngữ lập trình C

I. BỘ CHỮ VIẾT 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: +, -, *, /, =, <, >, (,)
- Các ký hiệu đặc biệt: :. , ; " ' _ @ # $! ^ [ ] { } ...
- Dấu cách hay khoảng trống.

Ii. Các từ khoá trong c

Từ khóa là các từ dành riêng (reserved words) của C mà người lập trình có thể sử dụng nó trong chương trình tùy theo ý nghĩa của từng từ. Ta không được dùng từ khóa để đặt cho các tên của riêng mình. Các từ khóa của Turbo C 3.0 bao gồm:

asm	auto	break	case	cdecl	char
class	const	continue	_cs	default	delete
do	double	_ds	else	enum	_es
extern	_export	far	_fastcall	float	for
friend	goto	huge	if	inline	int
interrupt	_loadds	long	near	new	operator
pascal	private	protected	public	register	return
_saveregs	_seg	short	signed	sizeof	_ss
static	struct	switch	template	this	typedef
union	unsigned	virtual	void	volatile	while

Iii. Cặp dấu ghi chú thích

Khi viết chương trình đôi lúc ta cần phải có vài lời ghi chú về 1 đoạn chương trình nào đó để dễ nhớ và dễ điều chỉnh sau này; nhất là phần nội dung ghi chú phải

không thuộc về chương trình (khi biên dịch phần này bị bỏ qua). Trong ngôn ngữ lập trình C, nội dung chú thích phải được viết trong cặp dấu /* và */.

Ví dụ :

```
#include <stdio.h>
#include<conio.h>
int  main ()
{
    char ten[50]; /* khai bao bien ten kieu char 50 ky tu */
         /*Xuat chuoi ra man hinh*/
    printf("Xin cho biet ten cua ban !");
    scanf("%s",ten);  /*Doc vao 1 chuoi la ten cua ban*/
    printf("Xin chao ban %s\n ",ten);
    printf("Chao mung ban den voi Ngon ngu lap trinh C");
    /*Dung chuong trinh, cho go phim*/
    getch();
    return 0;
}
```

IV. CÁC KIỂU DỮ LIỆU SƠ CẤP CHUẨN TRONG C

Các kiểu dữ liệu sơ cấp chuẩn trong C có thể được chia làm 2 dạng : kiểu số nguyên, kiểu số thực.

IV.1. Kiểu số nguyên

Kiểu số nguyên là kiểu dữ liệu dùng để lưu các giá trị nguyên hay còn gọi là kiểu đếm được. Kiểu số nguyên trong C được chia thành các kiểu dữ liệu con, mỗi kiểu có một miền giá trị khác nhau

IV.1.1. Kiểu số nguyên 1 byte (8 bits)

Kiểu số nguyên một byte gồm có 2 kiểu sau:

STT	Kiểu dữ liệu	Miền giá trị (Domain)
1	unsigned char	Từ **0** đến **255** (*tương đương 256 ký tự trong bảng mã ASCII*)
2	char	Từ **-128** đến **127**

Kiểu unsigned char: lưu các số nguyên dương từ 0 đến 255.

=> Để khai báo một biến là kiểu ký tự thì ta khai báo biến kiểu unsigned char. Mỗi số trong miền giá trị của kiểu unsigned char tương ứng với một ký tự trong bảng mã ASCII .

Kiểu char: lưu các số nguyên từ -128 đến 127. Kiểu char sử dụng bit trái nhất để làm bit dấu.

=> Nếu gán giá trị > 127 cho biến kiểu char thì giá trị của biến này có thể là số âm (?).

IV.1.2. Kiểu số nguyên 2 bytes (16 bits)

Kiểu số nguyên 2 bytes gồm có 4 kiểu sau:

STT	Kiểu dữ liệu	Miền giá trị (Domain)
1	enum	Từ -32,768 đến 32,767
2	unsigned int	Từ **0** đến **65,535**

3	short int	Từ -32,768 đến 32,767
4	int	Từ -32,768 đến 32,767

Kiểu enum, short int, int : Lưu các số nguyên từ -32768 đến 32767. Sử dụng bit bên trái nhất để làm bit dấu.

=> Nếu gán giá trị >32767 cho biến có 1 trong 3 kiểu trên thì giá trị của biến này có thể là số âm.

Kiểu unsigned int: Kiểu unsigned int lưu các số nguyên dương từ 0 đến 65535.

IV.1.3. Kiểu số nguyên 4 byte (32 bits)

Kiểu số nguyên 4 bytes hay còn gọi là số nguyên dài (long) gồm có 2 kiểu sau:

STT	Kiểu dữ liệu	Miền giá trị (Domain)
1	unsigned long	Từ **0** đến **4,294,967,295**
2	long	Từ **-2,147,483,648** đến **2,147,483,647**

Kiểu long : Lưu các số nguyên từ -2147483658 đến 2147483647. Sử dụng bit bên trái nhất để làm bit dấu.

=> Nếu gán giá trị >2147483647 cho biến có kiểu long thì giá trị của biến này có thể là số âm.

Kiểu unsigned long: Kiểu unsigned long lưu các số nguyên dương từ 0 đến 4294967295

IV.2. Kiểu số thực

Kiểu số thực dùng để lưu các số thực hay các số có dấu chấm thập phân gồm có 3 kiểu sau:

STT	Kiểu dữ liệu	Kích thước (Size)	Miền giá trị (Domain)
1	float	4 bytes	Từ **$3.4 * 10^{-38}$** đến **$3.4 * 10^{38}$**
2	double	8 bytes	Từ **$1.7 * 10^{-308}$** đến **$1.7 * 10^{308}$**
3	long double	10 bytes	Từ **$3.4 * 10^{-4932}$** đến **$1.1 * 10^{4932}$**

Mỗi kiểu số thực ở trên đều có miền giá trị và độ chính xác (số số lẻ) khác nhau. Tùy vào nhu cầu sử dụng mà ta có thể khai báo biến thuộc 1 trong 3 kiểu trên.

Ngoài ra ta còn có kiểu dữ liệu **void**, kiểu này mang ý nghĩa là kiểu rỗng không chứa giá trị gì cả.

V. Tên và hằng trong C

V.1 Tên (danh biểu)

Tên hay còn gọi là danh biểu (identifier) được dùng để đặt cho chương trình, hằng, kiểu, biến, chương trình con... Tên có hai loại là tên chuẩn và tên do người lập trình đặt.

Tên chuẩn là tên do C đặt sẵn như tên kiểu: int, char, float,…; tên hàm: sin, cos...

Tên do người lập trình tự đặt để dùng trong chương trình của mình. Sử dụng bộ chữ cái, chữ số và dấu gạch dưới (_) để đặt tên, nhưng phải tuân thủ quy tắc:

- Bắt đầu bằng một chữ cái hoặc dấu gạch dưới.
- Không có khoảng trống ở giữa tên.

- Không được trùng với từ khóa.
- Độ dài tối đa của tên là không giới hạn, tuy nhiên chỉ có 31 ký tự đầu tiên là có ý nghĩa.
- Không cấm việc đặt tên trùng với tên chuẩn nhưng khi đó ý nghĩa của tên chuẩn không còn giá trị nữa.

Ví dụ: tên do người lập trình đặt: Chieu_dai, Chieu_Rong, Chu_Vi, Dien_Tich
Tên không hợp lệ: Do Dai, 12A2,…

V.2. Hằng (Constant)

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 $-123.45 * 10^{4}$)

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)

Cách tính giá trị thập phân của số bát phân như sau:

Số bát phân : $0d_nd_{n-1}d_{n-2}\ldots d_1d_0$ (d_i có giá trị từ 0 đến 7)

=> Giá trị thập phân= $\sum_{i=0}^{n} d_i * 8^i$

0345=229 , 020=16

- *Hằng số nguyên 2 byte (int) hệ thập lục phân*: Là kiểu số nguyên sử dụng 10 ký số từ 0 đến 9 và 6 ký tự A, B, C, D, E ,F để biểu diễn một số nguyên.

Ký tự	giá trị
A	10
B	11
C	12
D	13
E	14
F	15

Cách biểu diễn: 0x<các ký số từ 0 đến 9 và 6 ký tự từ A đến F>

Ví dụ:

0x345 (số 345 trong hệ 16)
0x20 (số 20 trong hệ 16)
0x2A9 (số 2A9 trong hệ 16)

Cách tính giá trị thập phân của số thập lục phân như sau:

Số thập lục phân : $0xd_nd_{n-1}d_{n-2}\ldots d_1d_0$ (d_i từ 0 đến 9 hoặc A đến F)

=> Giá trị thập phân= $\sum_{i=0}^{n} d_i * 16^i$

0x345=827 , 0x20=32 , 0x2A9= 681

- *Hằng số nguyên 4 byte (long)*: Số long (số nguyên dài) được biểu diễn như số int trong hệ thập phân và kèm theo ký tự *l* hoặc L. Một số nguyên nằm ngoài miền giá trị của số int (2 bytes) là số long (4 bytes).

Ví dụ: 45345L hay 45345l hay 45345

- *Các hằng số còn lại*: Viết như cách viết thông thường (không có dấu phân cách giữa 3 số)

Ví dụ:

12 (mười hai)
12.45 (mười hai chấm 45)
1345.67 (một ba trăm bốn mươi lăm chấm sáu mươi bảy)

V.2.3. Hằng ký tự

Hằng ký tự là một ký tự riêng biệt được viết trong cặp dấu nháy đơn (‘). Mỗi một ký tự tương ứng với một giá trị trong bảng mã ASCII. Hằng ký tự cũng được xem như trị số nguyên.

Ví dụ: ‘a’, ‘A’, ‘0’, ‘9’

Chúng ta có thể thực hiện các phép toán số học trên 2 ký tự (thực chất là thực hiện phép toán trên giá trị ASCII của chúng)

V.2.4. Hằng chuỗi ký tự

Hằng chuỗi ký tự là một chuỗi hay một xâu ký tự được đặt trong cặp dấu nháy kép (“).

Ví dụ: “Ngon ngu lap trinh C”, “Khoa CNTT-DHCT”, “NVLinh-DVHieu”

Chú ý:

1. Một chuỗi không có nội dung “” được gọi là chuỗi rỗng.

2. Khi lưu trữ trong bộ nhớ, một chuỗi được kết thúc bằng ký tự NULL (‘\0’: mã Ascii là 0).

3. Để biểu diễn ký tự đặc biệt bên trong chuỗi ta phải thêm dấu \ phía trước.

Ví dụ: “I’m a student” phải viết “I\’m a student”
“Day la ky tu “dac biet”” phải viết “Day la ky tu \”dac biet\”“

VI. BIẾN VÀ BIỂU THỨC

VI.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.

VI.1.2. Vị trí khai báo biến trong C

Trong ngôn ngữ lập trình C, ta phải khai báo biến đúng vị trí. Nếu khai báo (đặt các biến) không đúng vị trí sẽ dẫn đến những sai sót ngoài ý muốn mà người lập trình không lường trước (hiệu ứng lề). Chúng ta có 2 cách đặt vị trí của biến như sau:

a) *Khai báo biến ngoài*: Các biến này được đặt bên ngoài tất cả các hàm và nó có tác dụng hay ảnh hưởng đến toàn bộ chương trình (còn gọi là biến toàn cục).

Ví dụ:

```
int      i;    /*Bien ben ngoai */
float         pi;    /*Bien ben ngoai*/
int  main()
{ … }
```

b) *Khai báo biến trong*: Các biến được đặt ở bên trong hàm, chương trình chính hay một khối lệnh. Các biến này chỉ có tác dụng hay ảnh hưởng đến hàm, chương trình hay khối lệnh chứa nó. Khi khai báo biến, phải đặt các ***biến này ở đầu của khối lệnh***, trước các lệnh gán, …

Ví dụ 1:

```
#include <stdio.h>
#include<conio.h>
int bienngoai;           /*khai bao bien ngoai*/
int  main ()
{   int j,i; /*khai bao bien ben trong chuong trinh chinh*/
  clrscr();
  i=1; j=2;
  bienngoai=3;
```

```
    printf("\n Gia7 tri cua i la %d",i);
          /*%d là số nguyên, sẽ biết sau */
    printf("\n Gia tri cua j la %d",j);
    printf("\n Gia tri cua bienngoai la %d",bienngoai);
    getch();
    return 0;
}
```

Ví dụ 2:

```
#include <stdio.h>
#include<conio.h>
int  main ()
{  int i, j;                          /*Bien ben trong*/
   clrscr();
   i=4; j=5;
   printf("\n Gia tri cua i la %d",i);
   printf("\n Gia tri cua j la %d",j);
   if(j>i)
   {
         int  hieu=j-i;                /*Bien ben trong */
         printf("\n Hieu so cua j tru i la %d",hieu);
   }
   else
   {
         int hieu=i-j                  ; /*Bien ben trong*/
         printf("\n Gia tri cua i tru j la %d",hieu);
   }
   getch();
   return 0;
}
```

VI.2. 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.

VI.2.1 Các toán tử số học

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 + -

VI.2.2 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

Bảng chân trị cho các toán tử Logic:

P	q	p&&q	p||q	!p
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Các toán tử quan hệ và Logic đều có độ ưu tiên thấp hơn các toán tử số học. Do đó một biểu thức như: 10 > 1+ 12 sẽ được xem là 10 > (1 + 12) và kết quả là sai (0).

Ta có thể kết hợp vài toán tử lại với nhau thành biểu thức như sau:

10>5&&!(10<9)||3<=4 Kết quả là đúng

Thứ tự ưu tiên của các toán tử quan hệ là Logic

Cao nhất: !

> >= < <=

== !=

&&

Thấp nhất: ||

VI.2.3 Các toán tử Bitwise:

Các toán tử Bitwise ý nói đến kiểm tra, gán hay sự thay đổi các Bit thật sự trong 1 Byte của Word, mà trong C chuẩn là các kiểu dữ liệu và biến char, int. Ta không thể sử dụng các toán tử Bitwise với dữ liệu thuộc các kiểu float, double, long double, void hay các kiểu phức tạp khác.

Toán tử	Ý nghĩa
&	AND
|	OR
^	XOR
~	NOT
>>	Dịch phải
<<	Dịch trái

Bảng chân trị của toán tử ^ (XOR)

p	q	p^q
0	0	0
0	1	1
1	0	1
1	1	0

VI.2.4 Toán tử ? cùng với :

C có một toán tử rất mạnh và thích hợp để thay thế cho các câu lệnh của If-Then-Else. Cú pháp của việc sử dụng toán tử ? là:

E1 ? E2 : E3

Trong đó E1, E2, E3 là các biểu thức.

Ý nghĩa: Trước tiên E1 được ước lượng, nếu đúng E2 được ước lượng và nó trở thành giá trị của biểu thức; nếu E1 sai, E2 được ước lượng và trở thành giá trị của biểu thức.

Ví dụ:

```
X = 10
Y = X > 9 ? 100 : 200
```

Thì Y được gán giá trị 100, nếu X nhỏ hơn 9 thì Y sẽ nhận giá trị là 200. Đoạn mã này tương đương cấu trúc if như sau:

```
X = 10
if (X < 9) Y = 100
else Y = 200
```

VII.2.5 Toán tử con trỏ & và *

Một con trỏ là địa chỉ trong bộ nhớ của một biến. *Một biến con trỏ* là một biến được khai báo riêng để chứa một con trỏ đến một đối tượng của kiểu đã chỉ ra nó. Ta sẽ tìm hiểu kỹ hơn về con trỏ trong chương về con trỏ. Ở đây, chúng ta sẽ đề cập ngắn gọn đến hai toán tử được sử dụng để thao tác với các con trỏ.

Toán tử thứ nhất là &, là một toán tử quy ước trả về địa chỉ bộ nhớ của hệ số của nó.

Ví dụ: m = &count

Đặt vào biến m địa chỉ bộ nhớ của biến count.

Chẳng hạn, biến count ở vị trí bộ nhớ 2000, giả sử count có giá trị là 100. Sau câu lệnh trên m sẽ nhận giá trị 2000.

Toán tử thứ hai là *, là một bổ sung cho &; đây là một toán tử quy ước trả về giá trị của biến được cấp phát tại địa chỉ theo sau đó.

Ví dụ: q = *m

Sẽ đặt giá trị của count vào q. Bây giờ q sẽ có giá trị là 100 vì 100 được lưu trữ tại địa chỉ 2000.

VI.2.6 Toán tử dấu phẩy ,

Toán tử dấu , được sử dụng để kết hợp các biểu thức lại với nhau. Bên trái của toán tử dấu , luôn được xem là kiểu void. Điều đó có nghĩa là biểu thức bên phải trở thành giá trị của tổng các biểu thức được phân cách bởi dấu phẩy.

Ví dụ: x = (y=3,y+1);

Trước hết gán 3 cho y rồi gán 4 cho x. Cặp dấu ngoặc đơn là cần thiết vì toán tử dấu , có độ ưu tiên thấp hơn toán tử gán.

VI.2.7 Xem các dấu ngoặc đơn và cặp dấu ngoặc vuông là toán tử

Trong C, cặp dấu ngoặc đơn là toán tử để tăng độ ưu tiên của các biểu thức bên trong nó.

Các cặp dấu ngoặc vuông thực hiện thao tác truy xuất phần tử trong mảng.

VI.2.8 Tổng kết về độ ưu tiên

Cao nhất	() []
	! ~ ++ -- (Kiểu) * &
	* / %
	+ -
	<< >>
	< <= > >=
	&
	^

	|
	&&
	||
	?:
	= += -= *= /=
Thấp nhất	,

VI.2.9 Cách viết tắt trong C

Có nhiều phép gán khác nhau, đôi khi ta có thể sử dụng viết tắt trong C nữa. Chẳng hạn:

x = x + 10 được viết thành x +=10

Toán tử += báo cho chương trình dịch biết để tăng giá trị của x lên 10.

Cách viết này làm việc trên tất cả các toán tử nhị phân (phép toán hai ngôi) của C. Tổng quát:

(Biến) = (Biến) (Toán tử) (Biểu thức)

có thể được viết:

(Biến) (Toán tử)= (Biểu thức)

VII. CẤu trúc củaA một chương trình C

VII.1. 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>

Chỉ thị thứ hai rất hay được sử dụng trong các tập tin thư viện (header file) đã được định nghĩa trước đó và thường được khai thác bởi các lập trình viên trong việc định nghĩa các ký hiệu như là:

```
#define NB_COUPS_MAX 100
#define SIZE 25
```

VII.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
}

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).

VII.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.

VII.4 Cú pháp khai báo các phần bên trong môt chương trình C

VII.4.1. Chỉ thị #include để sử dụng tập tin thư viện

Cú pháp:

#include <Tên tập tin> // Tên tập tin được đạt trong dấu <>

hay **#include "Tên đường dẫn"**

Menu Option của Turbo C có mục INCLUDE DIRECTORIES, mục này dùng để chỉ định các tập tin thư viện được lưu trữ trong thư mục nào.

Nếu ta dùng #include<Tên tập tin> thì Turbo C sẽ tìm tập tin thư viện trong thư mục đã được xác định trong INCLUDE DIRECTORIES.

Ví dụ: include <stdio.h>

Nếu ta dùng #include"Tên đường dẫn" thì ta phải chỉ rõ tên ở đâu, tên thư mục và tập tin thư viện.

Ví dụ: #include"C:\\TC\\math.h"

Trong trường hợp tập tin thư viện nằm trong thư mục hiện hành thì ta chỉ cần đưa tên tập tin thư viện. Ví dụ: #include"math.h".

Ví dụ:

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

VII.4.2. Chỉ thị #define để định nghĩa hằng số

Cú pháp:

#define <Tên hằng> <Giá trị>

Ví dụ:

#define MAXINT 32767

VII.4.3. Khai báo các prototype của hàm

Cú pháp:

<Kiểu kết quả trả về> Tên hàm (danh sách đối số)

Ví dụ:

```
long giaithua( int  n); //Hàm tính giai thừa của số nguyên n
double  x_mu_y(float  x, float y);       /*Hàm tính x mũ y*/
```

VII.4.4. Cấu trúc của hàm "bình thường"

Cú pháp:

<Kiểu kết quả trả về> Tên hàm (các đối số)
{
Các khai báo và các câu lệnh định nghĩa hàm
return kết quả;
}

Ví dụ:

```
int  tong(int x, int y)       /*Hàm tính tổng 2 số nguyên*/
{
  return (x+y);
}
float  tong(float x, float y)  /*Hàm tính tổng 2 số thực*/
{
  return (x+y);
}
```

VII.4.5. Cấu trúc của hàm main

Hàm main chính là chương trình chính, gồm các lệnh xử lý, các lời gọi các hàm khác.

Cú pháp:

<Kết quả trả về> main(đối số)
{
Các khai báo và các câu lệnh định nghĩa hàm
return <kết quả>;
}

Ví dụ 1:

```
int  main()
{
  printf("Day la chuong trinh chinh");
  getch();
```

```
    return 0;
}
```

Ví dụ 2:

```
int  main()
{
   int a=5, b=6,c;
   float   x=3.5, y=4.5,z;
   printf("Day la chuong trinh chinh");
   c=tong(a,b);
   printf("\n Tong cua %d va %d la %d",a,b,c);
   z=tong(x,y);
   printf("\n Tong cua %f và %f là %f", x,y,z);
   getch();
   return 0;
}
```

VIII. BÀI TẬP

Bài 1: Biểu diễn các hằng số nguyên 2 byte sau đây dưới dạng số nhị phân, bát phân, thập lục phân

a)12 b) 255 c) 31000 d) 32767 e) -32768

Bài 2: Biểu diễn các hằng ký tự sau đây dưới dạng số nhị phân, bát phân.

a) 'A' b) 'a' c) 'Z' d) 'z'