**2.3. Quy định khi viết một chương trình.**

- Sau mỗi câu lệnh có dấu kết thúc ;

- Khai báo những thông tin cần khi viết chương trình, tức là khai báo các hàm thư viện

**III.Cấu trúc chương trình trong TURBO C/C++**

**3.1. Hằng,biến**

***- Khai báo hằng.***

***const <kiểu dữ liệu> <tên hằng>= <giá trị>;***

**3.2.Biến.**

***<kiểu dữ liệu> <tên biến>;***

**3.3.Các loại dữ liệu cơ sở.**

- ***Kiểu số nguyên:*** **int**

+ ***Khai báo: int <tên biến>;***

- ***Kiểu số thực:*** **float**

+ ***Khai báo: float <tên biến>;***

***Kiểu ký tự:*** **char**

+ ***Khai báo: char <tên biến>;***

|  |  |  |  |
| --- | --- | --- | --- |
| Xuống dòng mớiLF | \n | Xuống dưới FF | \f |

+ Hàm xử lý xâu tự:

***toASCII(c):***Chuyển c thành mã ASCII,

***tolower(c),toupper(c):***Chuyển c thành chữ thường, hoa

**getchar():** Dừng chương trình chờ nhập một ký tự

- ***Kiểu boolean:*** ***bool***

+ ***Khai báo: bool <tên biến>;***

+ Nhận giá trị TRUE(1), FALSE(0)

+ Các phép toán: ***AND(&&); OR (||); NOT(!)*** với các giá trị như bảng chân lý logic.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| U | V | U&&V | U||V | !U |
| 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 1 | 1 |

***- Các hằng tượng trưng:***

***+ Khai báo 1: #define <Tên hằng> <Giá trị>;***

***+ Khai báo 2: #define <Tên hằng> <Xâu ký tự>;***

**3.2. Hàm.**

***<Kiểu hàm> <tên hàm>( <Các tham số hình thức>);***

***{***

***Khai báo các biến cục bộ;***

***....***

***Các câu lệnh***

***...***

***}***

***- Hàm cout: Dùng để thể hiện thông tin lên màn hình,***

***cout***<<***”<tham số định dạng>”<< <các tham số>;***

***- Hàm getch():***Có tác dụng dừng màn hình đang hiện kết quả, trước đó phải viết câu hướng dẫn tệp đầu

**3.3. Mảng**

- Khai báo mảng: <***Kiểu mảng> <tên mảng>[kích thước];***

**3.4.Cấu trúc của một chương trình.**

- Cấu trúc chương trình của C++ gồm:

***+*** ***Khai báo các hàm thư viện chuẩn sẽ sử dụng***

***+ Định nghĩa các biến ngoài***

***+ Khai báo tượng trưng các hàm con***

***+ Hàm chính main()***

***{***

* ***Hàm 1***
* ***Hàm 2***
* ***...***
* ***}***

***+ Nọi dung của các hàm con***

**3.5.Biến,hằng, khai báo biến và các hàm nhập xuất:**

- ***Phép gán:*** ***<tên biến>=<toán hạng>;***

***- Hàm cin***: Dùng để nhập dữ liệu từ bàn phím với cú pháp sau:

***+ Cú pháp*** c**in*>> <biến>;***

***- Hàm cout***: Dùng để đưa thông tin ra màn hình

***+ Cú pháp: cout<< <biến> hoặc cout<< ” dòng thông báo”;hoặc***

***cout<< ” dòng thông báo”<< <biến>***

***printf(<biến>)***

**- Chú ý:** Khi dùng **2 hàm trên phải gọi hàm *đầu là #include <iostream.h>,***

***#include <stdio.h>***

**3.6.Quy ước khi biểu diễn biến theo khuôn dạng:**

***- Hàm getchar:***

+ Cú pháp: ***<tên biến>=getchar();***

**IV. Các phép toán.**

**4.1 Quan hệ**

- Gồm : >(lớn hơn),<( nhỏ hơn),==(bằng),!=(khác)

- Cho giá trị là TRUE(1) nếu đúng và FALSE(0) nếu sai

**4.2 Logic (BOOL)**

- Gồm : Hoặc(||),và (&&),phủ định(!)

- Kết quả là giá trị ở các bảng chân lý

**4.3. Các phép toán số học.**

- Gồm: +, -, \*, /, % - cộng, trừ, nhân, chia, lấy phần dư

- Cách viết gọn:

+ a+=b tương đương a=a+b

+ a-=b tương đương a=a-b

+ a\*=b tương đương a=a\*b

+ a/=b tương đương a=a/b

+ a%=b tương đương a=a%b

**4.4. Tăng giảm một đơn vị.**

- ++ tăng lên 1, (++a) tương đương a+=1

- -- Giảm đi 1 (--b) tương đương b-=1

***4.5. Ép kiểu:***

int i,m,n;

float x;

x=m/n /\* Kết quả là phần nguyên\*/

x=m/ float(n) /\* n bị ép thành số thực nên Kết quả là số thực\*/

**4.6. Biểu thức.**

- Tổ hợp toán tử và toán hạng gọi là biểu thức.

***4.7 Chuyển kiểu bắt buộc trong biểu thức.***

- Cú pháp: **(*kiểu) <Toán hạng>.***

- Ví dụ: (float)(x/3+10\*y) kết quả là kiểu float và không phụ thuộc vào kết quả

**4.8. Các hàm số học dành cho số nguyên và số thực:**

|  |  |  |
| --- | --- | --- |
| **Hàm** | **Kiểu nhận được** | **Kết quả** |
| **abs(i)** | **int** | **Cho trị tuyệt đối của i** |
| **fabs(d)** | **double** | **Cho trị tuyệt đối của số thực d** |
| **sin(d),cos,cosh,tan,**  **exp,log,sqrt,floor,ceil** | **double** | **Cho trị gia trị sin,cos,cosh,tan,e mũ,loga,căn,cắt tròn số,hàm làm tròn số** |
| **pow(d1,d2),fmod(d1,d2)** | **double** | **d1 mũ d2, lấy d1/d2** |

**4.9. Kích thước của đối tượng.**

- Cú pháp: ***sizeof(toán hạng)***

**V. Các câu lệnh điều kiện.**

**5.1. Khái niệm.**

- Là các câu lệnh được thực hiện khi một điều kiện nào đó được thoả mãn

**5.2. Câu lệnh IF ...ELSSE.**

- Cú pháp: ***if*** ***(biểu thức)***

***{***

***Nhóm lệnh 1;***

***}***

***else***

***{***

***Nhóm lệnh 2;***

***}***

**5.3. Câu lệnh switch: Chọn một trong các phương án**

- Cú pháp: **switch(Biểu thức)**

**{**

**case <hằng 1>: <Dãy lệnh 1>;**

**case <hằng 2>: <Dãy lệnh 2>;**

**….**

**case <hằng k>: <Dãy lệnh k>;**

**defaut: <dãy lệnh k+1>;**

**}**

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

- Chú ý: Sau mỗi case phải có lệnh ***break*** để thoát ra khi thoả mãn.

**VI.Câu lệnh lặp.**

**6.1. Câu lệnh for (vòng lặp xác định).**

- Cú pháp:

***for(<biểu thức 1>;<biểu thức 2>;<biểu thức 3>)***

***{***

***<các lệnh>;***

***}***

***-*** <Biểu thức 1>: Dùng khởi tạo biến điều khiển vòng lặp.

- <Biểu thức 2>: Dùng để kiểm tra điều kiện thực hiện vòng lặp.

- <Biểu thức 3>: Dùng để thay đổi giá trị của biến điều khiển.

**6.2. Câu lệnh while.**

- Cú pháp:

***while(<biểu thức>)***

***{***

***< Các lệnh>***

***}***

**6.3. Câu lệnh do… while.**

- Cú pháp: **do**

**{**

**<Các lệnh>**

**}**

**w*hile(<biểu thức***

- Giải thích: Câu lệnh do while khác câu lệnh while ở chỗ while kiểm tra điều kiện rồi tiến hành thực hiện các lệnh trong { }, còn do… while lặp rồi mới kiểm tra điều kiện nếu đúng chạy tiếp.

- Chú ý : Câu lệnh do while thì phải có dấu kết thúc **;**ở sau

**6.4. Câu lệnh break.**

- Cú pháp: ***break;***

- Giải thích: Cho phép thoát khỏi các vòng lặp và chuyển quyền điều khiển cho lệnh nằm ngay sau vòng lặp,hoặc là kết thúc câu lệnh điều khiển đang thực hiện.

**6.5. Câu lệnh continue**

- Cú pháp: **continue;**

**6.6. Lệnh :goto**

-Lệnh goto sẽ chuyển điều khiển đến vị trị có nhãn trong chương trình.

**VII. Hàm và đệ quy.**

**7.1. Khái niệm hàm.**

***- Cú pháp:***

***<kiểu giá trị hàm> <ten hàm>(<các th.s hình thức và kiểu của chúng>); (1)***

Hoặc ***void <ten hàm>(<các th.s hình thức và kiểu của chúng>); (2)***

- Khi tính xong giá trị của hàm ta phải trả lại giá trị cho tên hàm bằng lệnh ***return <giá trị>***

- Nếu hàm không cần trả lại giá trị thì đầu tên hàm phải có từ khoá void.

**7.2.Tham số thực sự và tham số hình thức**

- Các tham số dùng khi khai báo gọi là tham số hình thức.

- Tham số được cung cấp cho hàm khi sử dụng gọi là tham số thực sự

- Tham số thực sự có thể là biểu thức.

- Phải có tương ứng 1 – 1 giữa tham số thực sự và tham số hình thức.

**7.3.Lệnh return.**

- Hàm **return** có thể trả giá trị của cả một biểu thức ví dụ

**7.4.Hàm không có giá tri.**

- Hàm không có giá trị phải được khai báo theo dạng (2) và không chứa lệnh return ở trong.

**7.4. Truyền tham số giá trị cho hàm**

**-** Truyền tham số cho hàm trong C/C++ được thực hiện duy nhất là ***truyền tham trị*** điều đó có nghĩa là tham số thực sự trước và sau khi gọi hàm là không đổi và đó là kiểu duy nhất.

**7.5. Biến toàn cục và biến địa phương.**

- Biến toàn cục là biến sử dụng mọi nơi trong chương trình. Nó được khai báo bên ngoài cả chương trình chính.

- Biến địa phương là biến được khai báo trong các hàm khi hàm kết thúc thì các biến này cũng mất luôn.

**7.6. Tầm hoạt động của biến**

- ***Biến toàn cục:*** Hoạt động trên toàn bộ chương trình ngay sau khi khai báo nó

***- Biến địa phương:*** Chỉ hoạt động trong hàm chưa nó. Cách cấp phát cho biến địa phương là cấp phát tự động, mỗi lần gọi hàm là một lần cấp phát.

- ***Biến địa phương tĩnh (static):****Là* biến do yêu cầu được cấp phát ở một vị trí cố định, loại biến này có thể tồn tại ngay cả khi hàm đã kết thúc hoạt động

- Các biến địa phương tĩnh được khởi tạo =0 khi bắt đầu lần gọi đầu

***static int m=3;*** /\* m là biến địa phương static \*/

**7.7. Tính đệ quy của hàm.**

- Đệ quy là một đối tượng T được thực hiện thông qua T1 giống hệt T nhưng có kích thước bé hơn.

- Hàm có thể có lời gọi chính nó đó là tính đệ quy của hàm.

- Ví dụ: n!=n\*(n-1)!

**VIII. Dữ liệu kiểu mảng.**

**8.1. Mảng 1 chiều (nhắc lại).**

- Cú pháp:***<Kiểu phần tử> <Tên mảng>[<Số ph. tử của mảng];***

**8.2. Khởi tạo giá trị mảng.**

- Mảng có thể khởi tạo giá trị ngay sau khi khai báo:

int a[3]={3,4,5}. Điều này nghĩa là a[0]=3,a[1]=4,a[2]=5,

**8.3.Mảng nhiều chiều**

- Cú pháp:***<kiểu mảng> <tên mang>[k.t.hàng][k.t.cột];***

**9.1. Khái niệm xâu.**

- C/C++ sử dụng mảng các xâu ký tự để chứa xâu ký tự.

- Xâu ký tự được lưu trữ trong mảng ký tự song nó được đánh dấu sự kết thúc xâu bằng một phần tử cuối cùng mang giá trị đặc biệt: Ký tự ***null*** là ký tự đầu tiên có mã ký tự 0 trong ASCII được ký hiệu trong C/C++ là ***’\0’***

- Khai báo: ***char <ten xâu>[<độ dài];***

- Lệnh ***gets(<tên xâu):*** Cho phép nhập <tên xâu> vào từ bàn phím

**9.2. Truy cập vào một phần tử của xâu.**

-Cú pháp: ***<tên xâu>[<chỉ số>]***

- Hàm xác định độ dài của xâu là: ***strlen(<xâu>)***.Hàm này có trong thư viện ***<string.h****>*.

**9.3. In một xâu ra màn hình.**

- Cú pháp: ***cout<< <tên xâu>;***

*-* Ví dụ: cout<<” Thi du”; cout<<st;

**9.4. Nhập một xâu từ bàn phím**

- Để đơn giản ta ký hiệu ***str*** là một xâu ký tự

- ***Cú pháp:*** + ***cin>> str; (1)*** lấy từ tệp ***iostream.h***

+ ***cin.get(str,max)***; ***(2)***lấy từ tệp ***iostream.h***

+ ***gets(str);*** ***(3)*** lấy từ tệp ***stdio.h***

**9.4. Ghép xâu ký tự.**

- Có thể dùng hàm ***strcat(str1,str2)*** trong thư viện ***string.h***

**9.5. Copy xâu ký tự.**

- Có thể dùng hàm **strcpy(str1,str2)** trong thư viện chuẩn <string.h>.

**9.5. So sánh xâu ký tự.**

- Hai xâu có độ dài khác nhau giống nhau đến độ dài xâu ngắn hơn được coi là xâu bé hơn.

- Hai xâu bằng nhau giống nhau nếu nội dung và độ dài của chúng giống nhau.

- Xâu trước sau tuân thủ thứ tự từ điển.

- Hàm so sánh: ***strcmp(str1,str2)***: Cho kết quả 0 nếu bằng, <0 nếu str1<str2,>0 nếu str1>str2.

**X. Kiểu dữ liệu có cấu trúc.**

- Kiểu dữ liệu có cấu trúc là kiểu dữ liệu mà mỗi phần tử có nhiều thành phần và các thành phần này có kiểu khác nhau. Trong C/C++ có định nghĩa kiểu **struct** viết tắt củachữ structure, gióng như bản ghi trong pascal.

**10.1. Mô tả và Khai báo.**

**- Cú pháp:**

***struct <Tên kiểu cấu trúc>***

***{***

***<kiểu dữ liệu 1> <trường 1>;***

***. . . . . . . . . .***

***<kiểu dữ liệu k> <trường k>;***

***} <biến cấu trúc>;***

Hoặc có thể khai báo trực tiếp như sau:

***struct***

***{***

***<kiểu dữ liệu 1> <trường 1>;***

***. . . . . . . . . .***

***<kiểu dữ liệu k> <trường k>;***

***} <biến cấu trúc>;***

Cũng có thể khai báo trực tiếp như sau:

***struct <Tên kiểu cấu trúc>***

***{***

***<kiểu dữ liệu 1> <trường 1>;***

***. . . . . . . . . .***

***<kiểu dữ liệu k> <trường k>;***

***}***

***struct <Tên kiểu cấu trúc> <biến>;***

**10.2. Truy nhập vào từng phần tử của struct**

- Truy nhập vào từng trường của struct theo cú pháp sau:

***<Tên biến cấu trúc>.<trường i>;***

**10.3.Kết hợp với mảng**

- Kết hợp với mảng làm cho chương trình sáng sủa hơn và tiện lợi hơn với cú pháp như sau:

***struct***

***{***

***<kiểu dữ liệu 1> <trường 1>;***

***. . . . . . . . . .***

***<kiểu dữ liệu k> <trường k>;***

***} <biến mảng>[độ dài];***

**XI. Dữ liệu kiểu tệp**

- Khi giải các bài toán ta cần lưu giữ lại các kết quả hoặc nhập những thông tin ban đầu, để viẹc làm đó có hiệu quả người ta thực hiện thông qua các tệp tin.

- Mỗi một phần tử cả tệp tin được bố trí như một cửa sổ, kết thúc tệp được ký hiệu là EOF.

- Trong C/C++ có một hàm chuẩn feof có kiểu boolean với tham số là biến tệp để cho biết cửa sổ đã đặt vào vị kết thúc tệp đó chưa(có giá trị true nếu kết thúc).

- Truy nhập vào tệp có 2 phương pháp: tuần tự và trực tiếp

- Việc xử lý tệp gồm 4 bước:

1. Khai báo biến tệp

2. Mở hoặc ghi tệp

3. Ghi hoặc đọc tệp

4. Đóng tệp

**11.1. Khai báo tệp.**

***- Cú pháp: FILE \*<tên con trỏ tệp>;***

***- Giải thích: <Tên con trỏ tệp>*** do người sử dụng đặt theo ý muốn miễn là hợp lệ. Ta hay dùng fprt (file pointer)

**11.2. Mở tệp ghi hoặc đọc.**

***- Cú pháp: <biến tệp>=fopen(<tên tệp trên đĩa>,<kiểu xử lý tệp>);***

**11.3. Các quy ước xử lý tệp**

- ”w” : Mở tệp văn bản mới để **ghi**.

- ”r” : Mở tệp đã tồn tại để **đọc**.

- ”a” : Mở tệp đã tồn tại và **ghi thêm** dữ liệu vào cuối

**11.4. Đóng tệp**

***- Cú pháp:*** ***fclose(<bien tep>)***;

- Cấu trúc chung của chương trình xử lý tệp có dạng:

#include <stdio.h>

main()

{

FILE \*fprt;

fprt=fopen(”vidu.dat”,”w”);

if (fprt==null)

cout<<” Khong mo duoc tep”)<<endl;

else

{

. . .

fclose(fprt);

}

}

**11.5 Ghi dữ liệu vào tệp nhị phân**

***- Hàm ghi dữ liệu vào tệp:***

***fwrite(&<biến>,sizeof(<biến>),<số khối>,<bien tep>)***

+ &<biến> là địa chỉ của khối dữ liệu

+ sizeof(<biến>): Kích thước <biến>

+ <số khối>: Số khối dữ liệu cần ghi vào tệp

+ Con trỏ tệp

**11.6 Đọc dữ liệu từ tệp nhị phân**

***- Hàm đọc dữ liệu từ tệp nhị phân đã có:***

***fread(&<biến>,sizeof(<biến>),<số khối>,<bien tep>)***

+ &<biến> là địa chỉ của biến cần ghi dữ liệu từ biến tệp ra(con trỏ tệp)

+ sizeof(<biến>): Kích thước <biến>

+ <số khối>: Số khối dữ liệu cần ghi vào tệp

+ <bien tep>:Con trỏ tệp

+ Khi ghi giá trị ra biến thì phải luôn kiểm tra xem cửa sổ tệp đã chỉ đến hết tệp chưa bằng hàm EOF bằng 1 trong các cách sau:

***while (fread(&i,sizeof(int),1,f), !feof(f))***

**11.7 Ghi Đọc dữ liệu cấu trúc**

- Các ví dụ trên xử lý các tệp có các phần tử có dữ liệu đơn giản chuẩn. Tệp thuộc kiểu struct cũng có kiểu nhị phân.

- Chương trình sau ho phép ta nhập dữ liệu từ bàn phím

**11.8. Truy cập trực tiếp tệp dữ liệu nhị phân.**

- C/C++ chỉ định nghĩa một kiểu tệp: Tệp truy cập tuần tự

- Truy nhên do độ dài của các phần tử là như nhau nên có thể tính toán toạ độ của chúng và điều này cho phép ta truy cập trực tiếp đén từng phần tử của tệp.

- Hàm truy cập trực tiếp là:**fseek**

**-Cúpháp:fseek(fptr,No\*<kich\_thuoc-phan\_tu>,<vi tri>)**

**- Giải thích:**

+ **fptr**: Con trỏ tệp

+ **No:** Vị trí số thứ tự của phần tử trong tệp

+ **<kich\_thuoc-phan\_tu>:** thường được thực hiện bằng sizeof(<ten kpt>) và cho ta toạ độ của phần tử.

+ Tham số cuối cùng:**<vi tri>=**

**SEEK\_SET:** Di chuyển vị trí truy cập bắt đầu từ đầu tệp

**SEEK\_CUR:** Di chuyển tính từ vị trí hiện tại của của sổ hiện tại của tệp

**SEEK\_END:** Di chuyển tính từ toạ độ cuối tệp.

***fseek(f,sizeof(int)\*(No-1 ),SEEK\_SET);***

**12. Tệp văn bản**

- Các phần tử của tệp văn bản là các ký tự và được tổ chức thành từng dòng.

- Mỗi dòng có ký hiệu kết thúc dòng(End of line)

- Các cặp ký tự điều khiển: CR(nhảy về đầu dòng mã số ASCII=13,’\r’); LF(nhảy xuống dòng tiếp theo mã số ASCII=10,’\n’)

**12.1. Mở tệp văn bản mới để cất dữ liệu**

**- Gồm các hàm sau:**

**1. putc(ch,fprt):** Ghi một ký tự vào tệp văn bản

**2. fputs(str,fprt)**: Ghi xâu ký tự vào tệp văn bản

**3. fprintf(fprt,…)**: Đổ kết qả ra tẹp văn bản có con trỏ

fprt

- Mỗi lần ghi xong máy sẽ tự động chuyển cửa sổ đến vị trí tiếp theo.

**- Vi du 54** Đọc một dòng các chữ từ tệp văn bản

#include <stdio.h>

#include <iostream.h>

#include <conio.h>

#include <string.h>

#include <ctype.h>

main()

{

FILE \*f;

char c;

/\* Cach nay doc doc ten tep vao tu ban phim

char tentep[11];

cout<<"Ten tep :"<<endl;

gets(tentep);

if((f=fopen(tentep,"r"))==NULL);

printf(" Tep co loi, hoac chua ton tai \n");

else \*/

if ((f=fopen("d:\\tc\\vidu.txt","r"))==NULL)

cout<<" Tep co loi, hoac chua ton tai "<<endl;

else

do

putchar(c=getc(f));

while(c!='\n');

fclose(f);

getch();

return 0;

}