Bài 2: Các thành phần cơ bản của C/C++

1. Hằng, Biến, Biểu thức

1.1. *Hằng*

Là đại lượng có giá trị không đổi trong chương trình.

Hằng số nguyên: có thể biểu diễn bằng số hệ Hexa hoặc biểu diễn thông thường.

Ví dụ:

```
const int n = 0x12A;            // khai báo hằng số n = 12A hệ Hexa (cơ số 16)
const int m = 123;        // khai báo hằng số m = 123 (hệ thập phân);
```

Hằng số thực: có hai cách biểu diễn, biểu diễn thông thường (dấu chấm tĩnh) hoặc biểu diễn khoa học (dấu chấm động)

Ví dụ:

```
const float pi= 3.1415;                 // khai báo hằng số pi biểu diễn dạng thông thường
const float x = 1e3;                    // khai báo hằng số thực x = 1×10^3 (dạng khoa học)
```

Hằng ký tự: kiểu ký có tên là *char*, trong C/C++ ký tự thực chất là số nguyên có giá trị trong bảng mã ASCII. Ký tự có thể biểu diễn bằng hai cách, đặt trong dấu ‘’ hoặc biểu diễn bằng giá trị mã ASCII. Ví dụ chữ cái ‘A’ có thể biểu diễn ‘\65’.

Ví dụ:

```
const char ch = ‘A’; //khai báo hằng ký tự ch có giá trị là  ‘A’.
```

Một số ký tự đặc biệt

Kí hiệu	Ý nghĩa
\n	Kí tự xuống dòng
\t	Kí tự TAB
\0	Kí tự NULL
\’	Dấu nháy đơn
\”	Dấu nháy kép
\\	Dấu sổ chéo ngược (Backslash)

Hằng chuỗi ký tự: được đặt trong cặp “” (nháy đôi). Thực chất đó là mảng các kí tự và có kí tự kết thúc chuỗi là kí tự NULL, kí hiệu ‘\0’ nằm cuối dãy.

Khai báo hằng: thường được đặt trong phần khai báo toàn cục ở đầu chương trình, ngay sau các khai báo chỉ thị tiền xử lý.

Có 2 cách khai báo hằng:

+ Dùng chỉ thị tiền xử lý:

```
#define  <tên hằng>  <chuỗi thay thế>;
```

+ Dùng từ khoá const:

```
const   <kiểu>   <tên hằng>=<giá trị>;
```

Ví dụ:

```
const  int MAX = 10;
```

hay

```
#define MAX 10
```

1.2. *Biến*

Là đại lượng có giá trị có thể thay đổi trong chương trình bằng toán tử gán (kí hiệu =).

Trong C/C++ cho phép khai báo biến ở khắp mọi nơi trong chương trình, miễn sao đảm bảo nguyên tắc: khai báo trước khi sử dụng.

Cách khai báo biến:

```
<kiểu> <danh sách biến>;
```

Ví dụ:

```
char ch;
int a,b;
double x,y,z;
```

Có thể khai báo đồng thời khởi đầu giá trị

Ví dụ :

```
int tong = 0, x = 5, y = 7;
```

1.3. *Biểu thức*

Là công thức tính toán bao gồm các toán hạng và toán tử tương ứng. Các toán hạng có thể là một biến, hằng, lời gọi hàm. Trong biểu thức sử dụng cặp ngoặc đơn () để ưu tiên thứ tự tính toán. Thứ tự ưu tiên tính toán trong một biểu thức không có dấu ngoặc tương tự trong toán học.

Có các loại biểu thức thông dụng sau:

+ Biểu thức gán.

+ Biểu thức số học.

+ Biểu thức logic.

Đặc biệt, biểu thức logic trong C/C++ được xem là có giá trị nguyên: bằng 0 tương ứng với FALSE ; khác 0 tương ứng với TRUE.

Kiểu của biểu thức phụ thuộc vào kiểu của giá trị trả về. Nếu khi gán giá trị của biểu thức cho một biến mà kiểu giá trị trả về của biểu thức khác kiểu của biến có thể chuyển đổi kiểu của giá trị biểu thức về cùng kiểu với kiểu của biến. Kiểu dữ liệu có phạm vi giá trị biểu diễn nhỏ hơn có thể chuyển đổi được sang kiểu có phạm vị giá trị biểu diễn lớn hơn.

Sơ đồ chuyển đổi kiểu ngầm định: `char → int → long → float → double → long double`

2. Các loại phép toán trong trong biểu thức

2.1. *Toán tử số học*

Kí hiệu	Ý nghĩa	Số ngôi	Toán hạng
+	Cộng	2	int, float, double, char
-	Trừ	2	- nt -
*	Nhân	2	- nt -
/	Chia	2	- nt -
%	Modulo	2	int, char (không có kiểu số thực)

2.2. Toán tử so sánh

Kí hiệu	Ý nghĩa	Số ngôi	Toán hạng
<	Nhỏ hơn	2	int, float, double, char
<=	Nhỏ hơn hoặc bằng	2	- nt -
>	Lớn hơn	2	- nt -

Kí hiệu	Ý nghĩa	Số ngôi	Toán hạng
>=	Lớn hơn hoặc bằng	2	- nt -
==	Bằng	2	- nt -
!=	khác	2	- nt -

2.3 Toán tử logic

Kí hiệu	Ý nghĩa	Số ngôi	Toán hạng
!	Phủ định (NOT)	2	int, float, double, char, bool
&&	Và (AND)	2	- nt -
||	Hoặc (OR)	2	- nt -

2.4 Toán tử gán

Ký hiệu: =

Cách viết: `<biến> = <biểu thức>`

Ví dụ:

```
int a, b;
a = 2;
b = 3;
a=a+b;
```

Trong C++ cho phép viết gọn các biểu thức gán bằng các toán tử gán như sau:

Dạng thường	Dạng thu gọn	Ý nghĩa
i=i+<bt>	i+=<bt>	Tự cộng
i=i-<bt>	i-=<bt>	Tự trừ
i=i*<bt>	i*=<bt>	Tự nhân
i=i/<bt>	i/=<bt>	Tự chia
i=i%<bt>	i%=<bt>	Tự MOD

2.5. Toán tử điều kiện

Là toán tử 3 ngôi, và có dạng: <BT1> ? <BT2> : <BT3>

Trong đó:

- <BT1> thường là một biểu thức so sánh hay một biểu thức logic
- <BT2>, và <BT3> là một biểu thức thông thường nào đó.

Kiểu của biểu thức điều kiện phụ thuộc vào kiểu của <BT2>, <BT3>.

Ví dụ 1: Xác định số nhỏ nhất trong 2 số x, y. Tacó thể viết: `x<y?x:y`

Ví dụ 2: Xác định số lớn nhất trong 3 số x, y, z. Ta có thể viết: `(x>y)?(x>z?x:z):(y>z?y:z)`

2.6. Toán tử tăng giảm một đơn vị

Biểu thức có dạng

<biến>++

<biến>--

++<biến>

--<biến>

Ví dụ:

```
int n,j;
n=5;
j=2;
```

`j=++n;` cho kết quả: `n=6, j=8`

```
n=5;
j=2;
```

`j=n++;` cho kết quả: `j=7, n=6.`

Sự khác nhau giữa hai dạng như sau:

- Dạng tiền tố: trị của <biến> được thay đổi trước khi tham gia biểu thức chung;
- Dạng hậu tố: biểu thức chung sử dụng trị cũ của <biến>, sau đó <biến> mới được thay đổi trị.

2.7. Toán tử thao tác trên bit

Chỉ thực hiện trên kiểu int

>> dịch phài

<< dịch trái

Ví dụ : `4>>1 = 2`

`4<<1 = 8`

Các phép toán trên từng bit

~ : lấy phần bù (đảo bit)

& : AND bit (hội)

| : OR bit (tuyển)

^ : XOR bit (tuyển loại)

2.8. Toán tử lấy địa chỉ của biến

&<biến>

2.9. Độ ưu tiên và thứ tự kết hợp các toán tử

Mức	Toán tử	Trật tự
1	() [] ->	→
2	! ~ ++ -- - * & (type) sizeof()	←
3	* /	→
4	+	→
5	<< >>	→
6	< <= > >=	→
7	== !=	→
8	&	→
9	^	→
10	|	→
11	&&	→
12	||	→
13	? :	←
14	= += -= *= /= %= . . .	←

3. Một số ví dụ

Ví dụ 1: Một hình tròn có diện tích 28.2735 cm. Hãy tính bán kính của hình tròn

```
#include <stdio.h>
#include <math.h>
const float pi = 3.141592564;
int main(){
      float s = 28.2735;
      float r = sqrt(s/pi);
      printf("DT hinh tron la %0.5f , ban kinh se la: %0.5f",s,r);
      return 0;
}
```

Ví dụ 2: In ra màn hình ký tự có mã ASCII bằng 100.

```
#include <stdio.h>
char a=100;
int main(){
      printf("chu cai co ma ASCII %d la %c",a,a);
      return 0;
}
```

Ví dụ 3: Chữ cái có mã ASCII lớn hơn chữ ‘B’ 13 đơn vị là chữ gì?

```
#include <stdio.h>
int main(){
      char ch='B';
      ch+=13;
      printf("chu cai %c co ma ASCII lon hon chu B 13 don vi",ch);
      return 0;
}
```

Ví dụ 4: Hai chữ cái ‘R’ vã ‘T’ chữ cái nào có mã ASCII là số lẻ.

```
#include <stdio.h>
char c1='S',c2='T';
```

```
int main(){
      char kq =(c1%2==1)?c1:c2;
      printf("Chu cai %c ma ascii le %d", kq,kq);
}
```

Ví dụ 5: Tính giá trị 2^{12}.

```
#include <stdio.h>
int main(){
      int  gt = 2;
      printf("2 ^ 12 = %d", gt << 11);
      return 0;
}
```