# Chương 2: Biến và Kiểu dữ liệu

## 2.1. Kiểu dữ liệu:

### a. Các loại dữ liệu thông dụng:

| Tên kiểu dữ liệu | Phạm vi | Ý nghĩa | Ví dụ |
|---|---|---|---|
| int | -32768 -> 32767 | Kiểu số nguyên | -2 |
| unsigned int | 0 -> 65535 | Số nguyên không âm | 5 |
| long | -2147483648->2147483647 | Số nguyên dài | 3445 |
| float | | Số thực | 4.5 |
| double | | Số thực | 7.8 |
| char | -128 -> 127 | Ký tự | 'A', '+' |
| unsigned char | 0 -> 255 | Ký tự | 'B','-' |
| bool | true, false | Kiểu logic | |

**Chú ý**:

- Kiểu ký tự cũng có thể là một dạng của kiểu số nguyên
- Kiểu bool Là kiểu thể hiện giá trị đúng sai
  + Có hai giá trị: true hoặc false
  + Các phép toán:
    && à phép AND (và)
    || à phép OR (hoặc)
    ! à phép NOT (phủ định)

### b. Kiểu dữ liệu do người dùng định nghĩa

- Từ khoá: typedef
- Cú pháp: typedef tên_kiểu_cũ tên_kiểu_mới
- Ý nghĩa: sau khi khai báo, ta có thể sử dụng tên_kiểu_mới như một kiểu dữ liệu bình thường khác.

Ví dụ:

- Định nghĩa lại tên kiểu dữ liệu số nguyên

```
typedef int so_nguyen;
```

Khi đó khi so_nguyen là một kiểu dữ liệu như kiểu int

## 2.2. Biến: Định nghĩa - Các thành phần của biến - Khai báo biến

- Mọi biến cần phải khai báo trước khi sử dụng.
- Việc khai báo biến được thực hiện theo cú pháp sau:

```
Kiểu_dữ_liệu   tên_biến;
Kiểu tên_biến = giá_trị_ban_đầu_của_biến;
Kiểu tên_biến1, tên-biến2,...;
```

Ví dụ:

```
int i, j;    //i và j là 2 kiểu số nguyên
char a;   //a là kiểu ký tự
float f;  // f là kiểu số thực
```

- Lưu ý:

+ Một tên hợp lệ là một dãy của các
  - Chữ (hoa và thường)
  - Số (Một tên không được bắt đầu bởi số)
  - Dấu nối dưới (Không nên bắt đầu bằng dấu nối dưới)

+ Các tên có phân biệt chữ hoa và chữ thường

Ví dụ: Bien1 khác với BIEN1

+Tên biến **không được trùng** với tên của **các lệnh**, các **từ khoá** trong C++ (ví dụ không được đặt tên biến là int, return, cout,...)

## 2.3. Biểu thức và toán tử: Biểu thức - Toán tử và độ ưu tiên của toán tử

| Tên toán tử | Ý nghĩa | Ví dụ |
|---|---|---|
| + | cộng | a+b |
| - | trừ | a-b |
| * | nhân | a*b |

| | | |
|---|---|---|
| / | chia | a/b |
| % | chia lấy phần dư | a%b |
| > | | a>b |
| < | | a<b |
| <= | nhỏ hơn hoặc bằng | a<=b |
| >= | lớn hơn hoặc bằng | a>=b |
| != | so sánh khác | a!=b |
| == | so sánh bằng | a==b |
| = | Lệnh gán | a = b |
| += | Viết tắt của lệnh a=a+b | a+=b |
| -= | Viết tắt của lệnh a=a-b | a-=b |
| *= | Viết tắt của lệnh a=a*b | a*=b |
| /= | Viết tắt của lệnh a=a/b | a/=b |
| %= | Viết tắt của lệnh a=a%b | a%=b |
| ++ | Viết tắt của lệnh<br>a=a+1 | a++<br>++a |
| -- | Viết tắt của lệnh<br>a=a-1 | a--<br>--a |

## 2.4. **Câu lệnh gán: Cú pháp - Thi hành câu lệnh gán**

Cú pháp: tên_biến = biểu thức (hoặc biến);

Ý nghĩa:

> Giá trị của biến bên trái dấu "=" sẽ bằng giá trị của biểu thức *sau khi tính toán xong* (hoặc giá trị của biến) bên phải dấu "="

Ví dụ:

```
int i, j ;       //khai báo biến i  và j là kiểu số nguyên
i  = 3;    // i sẽ có giá trị là 3;
j = i +7;  // j sẽ có giá trị là 10  ( 3 + 7)
```

**Lưu ý:**

So sánh phép toán **a++** với **++a** (tương tự cho **a--**và **-- a**)

Ví dụ:

```
/* khai báo biến a và b là kiểu số nguyên và cùng có giá  trị ban đầu là 5
*/
int a = 5;
int b = 5;
int i,j;  //khai báo biến i và j
```

`i = ++a;` Ý nghĩa:

- Thực hiện phép toán **a=a+1 trước**, tức giá trị a tăng lên 1 đơn vị, hay a sẽ có giá trị là 6
- Sau **đó i sẽ được gán giá trị 6 của a**
- Giá trị của a sau lệnh này sẽ có giá trị là 6

`j = b++;` Ý nghĩa:

- Thực hiện phép **gán j=b trước**, do giá trị của b hiện giờ là 5, **nên j sẽ có có trị là 5**
- Sau đó phép toán b=b+1 sẽ được thục hiện, tức giá trị của b sẽ được tăng lên 1 đơn vị, khi đó b sẽ có giá trị là 6

## 2.5. **Chuyển đổi kiểu (ép kiểu)**

Ép kiểu là một thủ thuật khá hay dùng để biến đổi kiểu của một biến, hay một biểu thức sang một kiểu dữ liệu phù hợp với nhu cầu tính toán của chương trình.

**Cú pháp**: (tên kiểu mới) tên_biến(hoặc biểu thức);

Lưu ý: giá trị của phép ép kiểu sẽ chuyển giá trị sang kiểu dữ liệu mới, nhưng **kiểu dữ liệu của những biến tham gia vào phép ép kiểu vẫn không bị thay đổi kiểu dữ liệu**

Ví dụ 1:

1.34 : là một số thực, thuộc kiểu số thực

(int) 1.34 : giá trị được ép kiểu sang kiểu số nguyên, khi đó kết quả sẽ là 1

Ví dụ 2:

```
float i = 2.34;   /* i được khai báo là kiểu số thực, với giá trị ban đầu là 2.34 */
int j = (int) i ; // ép kiểu khi đó j sẽ có giá trị là 2, tuy nhiên giá trị của i vẫn là 2.34
```

**Những lưu ý quan trọng**:

- Tất cả lệnh đều kết thúc bằng dấu chấm phẩy ;
- Cách chú thích:
  - Chú thích 1 dòng: dùng ký hiệu //
  - Chú thích nhiều dòng: dùng ký hiệu mở /* và ký kiệu đóng */
    - Ví dụ:

```
/* đây là dòng chú thích dòng 1
Đây là dòng chú thích dòng 2 */
```

- Một số ký tự đặc biệt:

**'\n'** là mã xuống dòng

**'\t'** là mã dấu Tab

**' \" '** là mã dấu nháy kép "

**'\\'** là mã dấu \

- Chuỗi: bắt đầu và kết thúc bằng dấu **"** và **"**
  - Ví dụ:

"Đây là một chuỗi"

"Đây là một chuỗi có xuống dòng \n"

## 2.6 **Lệnh xuất/nhập chuẩn:**

Một số lệnh cơ bản cần nắm

- std::cout (xuất dữ liệu ra màn hình)

  Ví dụ:

  std::cout<< a ; // xuất giá trị của a ra màn hình

  std::cout<<a<<b; // xuất giá trị của a và b ra màn hình

- std::cin (nhập giá trị vào vào biến)

  Ví dụ:

  std::cin>>a; /*May tính sẽ yêu cầu người dùng nhập một giá trị từ bàn phím, sau khi nhập từ phím 1 giá trị và bấm **enter**, thì a sẽ mang giá trị mà người đó vừa nhập*/

- Để việc lập trình thuận tiện, thay vì phải đánh chữ std trước một số lệnh, ta khai báo tên vùng std ở phía trước hàm main (bằng cú pháp **using namespace std**), khi đó ta không cần phải đánh lại std trước các lệnh đó

  Ví dụ:

```
//Chương trình nhập xuất giá trị của biến
#include <iostream>     //Khai báo thư viện hàm cout, cin
#include <conio.h>      //khai báo thư viện hàm getch()

using namespace std;   //khai báo tên vùng lệnh std

int main()
{
        int i; //Khai báo biến i là kiểu số nguyên
        int j = 10;   /*khai báo biến j là kiểu số nguyên với giá trị ban đầu là
10 */

        cin>>i;   //Nhập giá trị và biến i
        cout<<i<<j;  //Xuất giá trị của biến i và j ra màn hình
```

```
    getch();  //Dừng màn hình
    return 0;
}
```