

Môn PPLTHĐT

Hướng dẫn thực hành tuần 1

Mục tiêu

Giới thiệu sơ lược về nhập xuất trong C++, tìm hiểu một số khái niệm cơ bản như hàm toán tử và function template, thống nhất một số chuẩn và quy ước trong lập trình.

Nội dung

- Nhập xuất trong C++.
- Các chuẩn và quy ước lập trình.

Yêu cầu

Nắm vững phương pháp lập trình cấu trúc trong C và biết cách sử dụng môi trường lập trình Visual Studio 2005 trở lên.

1. Nhập xuất trong C++

Cũng như C, C++ không có các hàm nhập xuất được xây dựng sẵn trong ngôn ngữ. Tất cả việc nhập xuất trong C++ đều thông qua các thư viện nhập xuất (như “**stdio**” và “**iostream**”). Nếu như “**stdio**” là thư viện nhập xuất chuẩn thường được sử dụng trong C thì “**iostream**” là thư viện nhập xuất thường được sử dụng trong C++. Để sử dụng thư viện “**iostream**”, ta thêm dòng sau vào đầu chương trình: “**#include "iostream.h"**”.

Có rất nhiều đối tượng nhập xuất trong thư viện “**iostream**” để đáp ứng từng yêu cầu cụ thể của lập trình viên. Ở đây, chúng ta chỉ quan tâm đến 2 đối tượng nhập xuất cơ bản là “**cin**” và “**cout**”.

cin

Đối tượng nhập chuẩn (bàn phím).

Để đọc giá trị nhập vào từ bàn phím, ta dùng toán tử >> (extraction) hoặc hàm “**getline**”.

Ví dụ:

```
unsigned char s[20];           // Lay cac ky tu trong cin dua vao chuoai s
cin >> s;                     // cho den khi gap khoang trong.

int x;                         // Doc vao tu cin mot so nguyen
cin >> x;                     // dua vao bien x.

unsigned char t[256];          // Lay cac ky tu trong cin dua vao chuoai t
cin.getline(t, 256, '\n');     // cho den khi gap ky tu '\n' hoac den 255 ky tu.
```

cout

Đối tượng xuất chuẩn (màn hình).

Để xuất giá trị ra màn hình, ta dùng toán tử << (insertion).

Ví dụ:

```
cout << 'A';                  // Dua ky tu 'A' vao cout.
cout << "Hello world";        // Dua chuoai "Hello world" vao cout.
cout << "Hello" << "\n" << "world"; // Lan luoc dua chuoai "Hello", ky tu '\n', roi chuoai
                                // "world" vao cout.
cout << 5;                    // Dua so 5 vao cout.
```

2. Các chuẩn và quy ước lập trình

Quy ước đặt tên hằng

Trong C++, hằng số được khai báo bằng từ khóa “**#define**” hoặc “**const**”. Một số quy ước trong việc đặt tên hằng như sau:

- i) Tên hằng phải thể hiện được ý nghĩa của nó.

```
#define N 100 // Không rõ nghĩa.
#define NUMBER_OF_ELEMENTS 100 // Rõ nghĩa.
```

- ii) Tên hằng được viết hoa toàn bộ và các từ trong tên cách nhau bằng ký tự “_”.

```
#define NumberOfElements 100 // Sai.
#define NUMBEROFELEMENTS 100 // Sai.
#define NUMBER_OF_ELEMENTS 100 // Đúng.
```

Quy ước đặt tên biến

- i) Tên biến phải thể hiện được ý nghĩa của nó.

```
int t, m; // Không rõ nghĩa.
int iTuSo, iMauSo; // Rõ nghĩa.
```

- ii) Tên biến được viết hoa các ký tự đầu mỗi từ trong tên, các ký tự còn lại viết thường.

```
int ituso, imauso; // Sai.
int iTuso, iMauso; // Sai.
int iTuSo, iMauSo; // Đúng..
```

- iii) Tên biến có phần tiếp đầu ngữ (prefix) thể hiện kiểu dữ liệu của biến (phong cách Hungarian):

Kiểu dữ liệu số		
char – c	char	cKyTu;
short – s	short	sSoNguyenNgan;
int – i	int	iSoNguyen;
long – l	long	lSoNguyenDai;
float – f	float	fSoThuc;
double – d	double	dSoThucDai;
	int	nSo;

Kiểu dữ liệu luận lý		
bool – b	bool	bLuanLy;

Kiểu dữ liệu mảng		
[] – arr	int HocSinh	arrSoNguyen[50]; arrDanhSach[50];
Kiểu dữ liệu chuỗi		
char *, char [] – str	char char	*strChuoi; strChuoi[50];
Kiểu dữ liệu con trỏ		
* - p	int HocSinh	*pConTro; *pDanhSach;

Quy ước đặt tên kiểu dữ liệu tự định nghĩa

- i) Tên kiểu dữ liệu tự định nghĩa (struct, class) thường là danh từ và phải thể hiện được ý nghĩa của kiểu dữ liệu đó.

```
struct TinhPhanSo           // Sai.
struct PhanSo              // Dung.

struct TinhDiemHocSinh     // Sai.
class HocSinh              // Dung.
```

- ii) Tên kiểu dữ liệu tự định nghĩa được viết hoa các ký tự đầu mỗi từ trong tên, các ký tự còn lại viết thường.

```
struct phanso              // Sai.
struct PHANSO             // Sai.
struct PhanSo             // Sai.
struct PhanSo             // Dung.
```

Quy ước đặt tên hàm

- i) Tên hàm thường là động từ và phải thể hiện hành động cần thực hiện.

```
int DataFile(char *strFileName) // Sai.
int LoadDataFile(char *strFileName) // Dung.

int BadValue(long lValue) // Sai.
int CheckForBadValue(long lValue) // Dung.
```

- ii) Tên hàm được viết hoa các ký tự đầu mỗi từ trong tên, các ký tự còn lại viết thường.

```
int checkforbadvalue(long lValue) // Sai.
```

```
int CheckforBadvalue(long lValue)    // Sai.
int CheckForBadValue(long lValue)    // Dung.
```

Quy ước viết câu lệnh

- i) Viết mỗi câu lệnh riêng trên một dòng.

```
// Sai.
x = 3; y = 5;
```

```
// Dung.
x = 3;
y = 5;
```

```
// Sai.
if (a > b) cout << "a lon hon b";
else cout << "a nho hon b";
```

```
// Dung.
if (a > b)
    cout << "a lon hon b";
else
    out << "a nho hon b";
```

```
// Sai.
for (int i = 0; i < n; i++) x = x + 5;
```

```
// Dung.
for (int i = 0; i < n; i++)
    x = x + 5;
```

- ii) Viết các dấu “{ “}” riêng trên một dòng.

```
// Sai.
void Swap(int &a, int &b) {
    int c = a;
    a = b;
    b = c;
}
```

```
void Swap(int &a, int &b)
{
    int c = a;
    a = b;
    b = c;
}
```

```
// Dung.
void Swap(int &a, int &b)
{
    int c = a;
    a = b;
    b = c;
}
```

iii) Viết các câu lệnh if, while, for riêng trên một đoạn.

<pre>// Sai. if (a > b) cout << "a lon hon b"; for (int i = 0; i < n; i++) x = x + 5; k = k * x;</pre>	<pre>// Dung. if (a > b) cout << "a lon hon b"; for (int i = 0; i < n; i++) x = x + 5; k = k * x;</pre>
--	--

iv) Viết các câu lệnh cùng thực hiện một công việc riêng trên một đoạn.

<pre>// Sai. int c = a; a = b; b = c; k = k * a; x = b + c;</pre>	<pre>// Dung. int c = a; a = b; b = c; k = k * a; x = b + c;</pre>
---	---

Quy ước cách khoảng

i) Viết cách vào một khoảng tab đối với các câu lệnh nằm giữa dấu “{” “}”.

<pre>// Sai. void Swap(int &a, int &b) { int c = a; a = b; b = c; }</pre>	<pre>// Dung. void Swap(int &a, int &b) { int c = a; a = b; b = c; }</pre>
---	--

ii) Viết cách vào một khoảng tab đối với câu lệnh ngay sau if, else, while, for.

<pre>// Sai. if (a > b) cout << "a lon hon b"; else cout << "a nho hon b"; for (int i = 0; i < n; i++) x = x + 5;</pre>	<pre>// Dung. if (a > b) cout << "a lon hon b"; else cout << "a nho hon b"; for (int i = 0; i < n; i++) x = x + 5;</pre>
--	---

iii) Viết cách một khoảng trắng xung quanh các toán tử 2 ngôi.

```
x=x+5*a-c;      // Sai.
x = x + 5 * a - c; // Dung.
```

```
if (a>=b) // Sai.
if (a >= b) // Dung.
```

iv) Viết cách một khoảng trắng sau các dấu “,” “.”.

```
void CalculateValues(int a,int b,int c); // Sai.
void CalculateValues(int a, int b, int c); // Dung.
```

```
for (int i = 0;i < n;i++) // Sai.
for (int i = 0; i < n; i++) // Dung.
```

Quy ước viết chú thích

Trong C++, chúng ta dùng dấu “//” hoặc “/*” “*/” để viết chú thích cho chương trình. Một số quy ước khi viết chú thích như sau:

i) Chú thích phải rõ ràng, dễ hiểu và diễn giải được ý nghĩa của đoạn lệnh.

```
// Vi du chu thich so sai.
// Merge sort, gan : n * log(2)n, can mang phu b
void msort(int a[], int n, int l, int r, int b[])
{
    int    m, i, j, k;

    m = (l + r) / 2;

    if (l < m)
        msort(a, n, l, m, b);

    if (m + 1 < r)
        msort(a, n, m + 1, r, b);

    for (i = l; i <= m; i++)
        b[i] = a[i];

    for (i = m + 1; i <= r; i++)
        b[i] = a[m + 1 + r - i];

    for (i = l, j = l, k = r; i <= r; i++)
        if (b[j] < b[k])
            a[i] = b[j++];
        else
            a[i] = b[k--];
}
```

```

}

// Ví dụ chu thích rõ ràng, đầy đủ.
// Merge sort, gan :  $n * \log(2)n$ , cần mảng phụ b
void msort(int a[], int n, int l, int r, int b[])
{
    int m, i, j, k;

    // Lấy vị trí giữa của a
    m = (l + r) / 2;

    // Thực hiện merge sort trên a từ vị trí l đến m
    if (l < m)
        msort(a, n, l, m, b);

    // Thực hiện merge sort trên a từ vị trí m + 1 đến r nếu m < r
    if (m + 1 < r)
        msort(a, n, m + 1, r, b);

    // Do các phần tử của a từ vị trí l đến m có thu vào b
    for (i = l; i <= m; i++)
        b[i] = a[i];

    // Do các phần tử của a từ vị trí m + 1 đến r có thu vào b theo thứ tự
    // ngược
    for (i = m + 1; i <= r; i++)
        b[i] = a[m + 1 + r - i];

    // Trộn b từ vị trí l đến m có thu vào a và b từ vị trí r đến m + 1 có thu vào a
    for (i = l, j = l, k = r; i <= r; i++)
        if (b[j] < b[k])
            a[i] = b[j++];
        else
            a[i] = b[k--];
}

```

ii) Dùng dấu “//” thay cho “/*” “*/” khi viết chú thích.

<pre> // Sai. /* void Swap(int &a, int &b) { int c = a; a = b; b = c; } */ </pre>	<pre> // Đúng. //void Swap(int &a, int &b) //{ // int c = a; // a = b; // b = c; //} </pre>
---	--