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:

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

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

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 chuoi "Hello world" vao cout.

cout << "Hello" << '\n' << "world";  // Lan luoc dua chuoi "Hello", ky tu '\n', roi chuoi

// "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 // Khong ro nghia. #define NUMBER_OF_ELEMENTS 100 // Ro nghia.
```

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 // Dung.
```

Const int x = 8;

Quy ước đặt tên biến

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

```
int t, m; // Khong ro nghia. int iTuSo, iMauSo; // Ro nghia.
```

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; // Dung..
```

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	arrSoNguyen[50];
	HocSinh	arrDanhSach[50];

Kiểu dữ liệu chuỗi		
char *, char [] – str	char	*strChuoi;
	char	strChuoi[50];

Kiểu dữ liệu con trỏ		
* - p	int	*pConTro;
	HocSinh	*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";</pre>
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.

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

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

// Dung.
if (a > b)
    cout << "a lon hon b";

for (int i = 0; i < n; i++)
    x = x + 5;
k = k * x;
```

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

```
// Sai.
int c = a;
a = b;
b = c;
k = k * a;
x = b + c;

// Dung.
int c = a;
a = b;
b = c;
k = k * a;
x = b + c;
```

Quy ước cách khoảng

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

```
// Sai.

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;
}
```

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

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 Calculate Values (int a, int b, int c); // Sai.
void Calculate Values (int a, int b, int c); // Dung.
for (int i = 0; i < n; i++) // Sai.</li>
```

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 = (1 + r) / 2;
    if (1 < m)
            msort(a, n, 1, m, b);
    if (m + 1 < r)
            msort(a, n, m + 1, r, b);
    for (i = 1; i \le m; i++)
            b[i] = a[i];
    for (i = m + 1; i \le r; i++)
            b[i] = a[m + 1 + r - i];
    for (i = 1, j = 1, k = r; i \le r; i++)
            if(b[i] < b[k])
                    a[i] = b[j++];
            else
                     a[i] = b[k--];
```

```
// Vi du chu thich ro rang, day du.
    // 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;
        // Lay vi tri giua cua a
        m = (1 + r) / 2;
        // Thuc hien merge sort tren a tu vi tri l den m
        if(1 < m)
                msort(a, n, 1, m, b);
        // Thuc hien merge sort tren a tu vi tri m + 1 den r neu m < r
        if (m + 1 < r)
                msort(a, n, m + 1, r, b);
        // Do cac phan tu cua a tu vi tri l den m co thu tu vao b
        for (i = 1; i \le m; i++)
               b[i] = a[i];
        // Do cac phan tu cua a tu vi tri m + 1 den r co thu tu vao b theo thu tu
        // nguoc
        for (i = m + 1; i \le r; i++)
                b[i] = a[m + 1 + r - i];
        // Tron b tu vi tri l den m co thu tu va b tu vi tri r den m + 1 co thu tu vao a
        for (i = 1, j = 1, k = r; i \le r; i++)
                if(b[i] < b[k])
                        a[i] = b[j++];
                else
                        a[i] = b[k--];
    }
ii) Dùng dấu "//" thay cho "/*" "*/" khi viết chú thích.
     // Sai.
                                         // Dung.
     /* void Swap(int &a, int &b)
                                         //void Swap(int &a, int &b)
     {
                                         //{
                                         //
             int c = a;
                                                 int c = a;
                                         //
             a = b;
                                                 a = b;
                                         //
                                                 b = c:
             b = c:
     } */
                                         //}
```