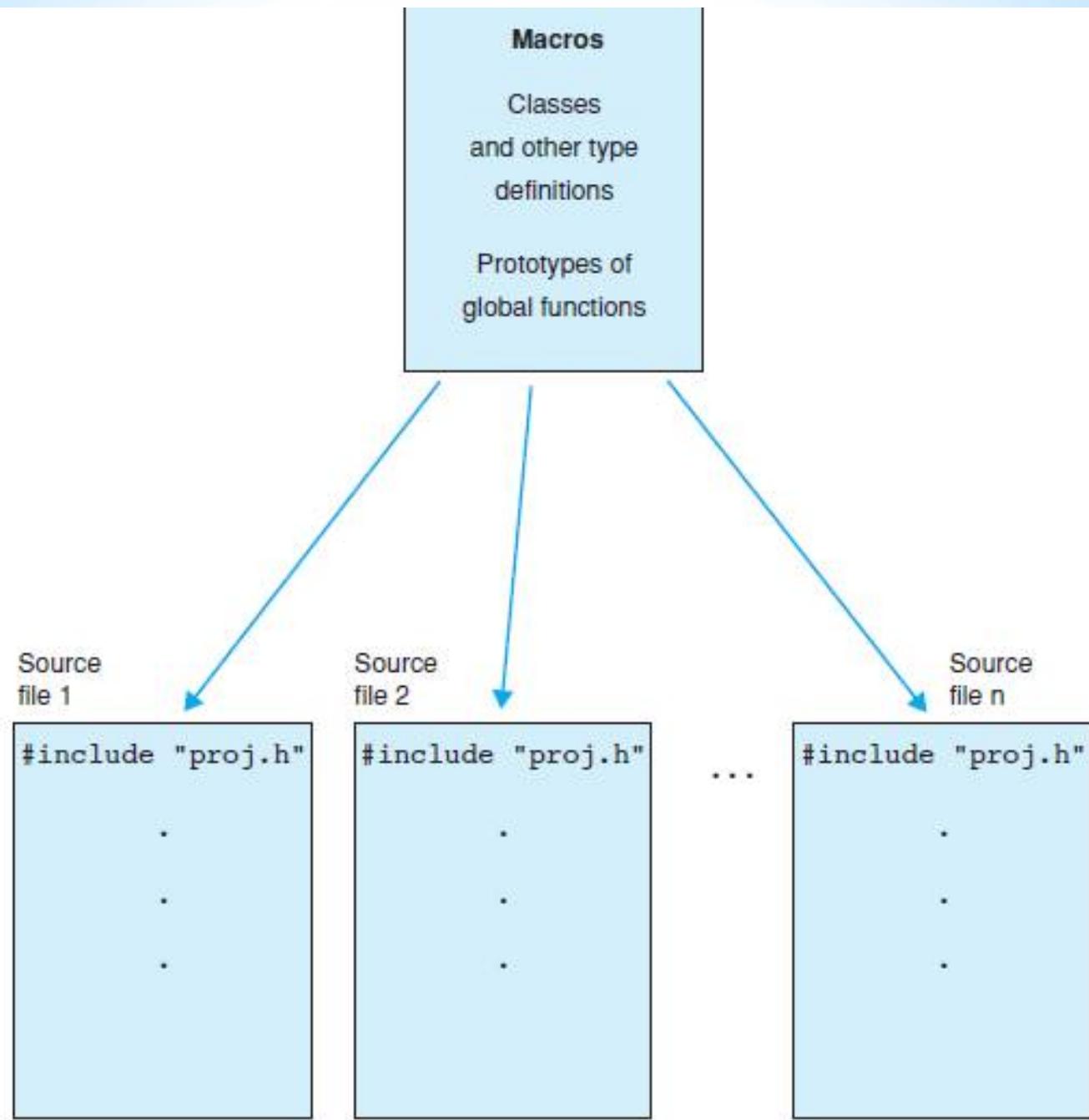


PART 3. LẬP TRÌNH NÂNG CAO VỚI C++

Nội dung.

- 3.1. Thiết kế chương trình
- 3.2. Xử lý ngoại lệ (Exception)
- 3.3. Bộ nhớ động (Dynamic Programming)
- 3.4. Không gian tên (Name Space)
- 3.5. Lập trình mẫu (Template)
- 3.6. Xử lý song song (Multi-Thread)
- 3.7. Lập trình với STL
- 3.8. CASE STUDY

3.1. Tổ chức chương trình



3.1. 1. Làm việc với Macro

```
#include <iostream>
#include <iomanip>
#include <cmath>
using namespace std;
#define PI 3.1415926536          // định nghĩa số PI
#define START 0.0                 // định nghĩa cận dưới START
#define END (2.0 * PI)            // định nghĩa cận trên
#define STEP (PI / 8.0)           // định nghĩa bước tính
#define HEADER (cout << \
" ***** Nguyễn Thanh Tùng D13CN1 *****\n\n")
int main(){
    HEADER; // hiển thị Title
    cout << setw(16) << "x" << setw(20) << "sin(x)\n"
        << " -----"
        << fixed << endl;
    for( double x = START; x < END + STEP/2; x += STEP)
        cout << setw(20)<<x<<setw(16)<<sin(x)<< endl;
    cout << endl << endl;
    system("pause");
    return 0;
}
```

Kết quả thực hiện:

***** Nguyen Thanh Tung D13CN *****

x	sin(x)
0.000000	0.000000
0.392699	0.382683
0.785398	0.707107
1.178097	0.923880
1.570796	1.000000
1.963495	0.923880
2.356194	0.707107
2.748894	0.382683
3.141593	-0.000000
3.534292	-0.382683
3.926991	-0.707107
4.319690	-0.923880
4.712389	-1.000000
5.105088	-0.923880
5.497787	-0.707107
5.890486	-0.382683
6.283185	0.000000

Press any key to continue . . .

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS1.h:

```
#ifndef _CLASS1_      //Khai báo tên file thư viện là CLASS1.h
#define _CLASS1_       //Định nghĩa tên file là CLASS1.h
#include   <iostream>
using namespace std;
class CLASS1 { // Khai báo các lớp một cách bình thường
public :
    void Function1(void){
        cout<<"Tap thao tac so:CLASS1.H"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS2.h:

```
#ifndef _CLASS2_      //Khai báo tên file thư viện là CLASS2.h
#define _CLASS2_       //Định nghĩa tên file là CLASS2.h
#include   <iostream>
using namespace std;
class CLASS2 { // Khai báo các lớp một cách bình thường
public :
    void Function2(void){
        cout<<"Tap thao tac so:CLASS2.H"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS3.h:

```
#ifndef _CLASS3_      //Khai báo tên file thư viện là CLASS3.h
#define _CLASS3_       //Định nghĩa tên file là CLASS3.h
#include   <iostream>
using namespace std;
class CLASS3 { // Khai báo các lớp một cách bình thường
public :
    void Function3(void){
        cout<<"Tap thao tac so:CLASS3.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS4.h:

```
#ifndef _CLASS4_      //Khai báo tên file thư viện là CLASS4.h
#define _CLASS4_       //Định nghĩa tên file là CLASS4.h
#include   <iostream>
using namespace std;
class CLASS3 { // Khai báo các lớp một cách bình thường
public :
    void Function4(void){
        cout<<"Tap thao tac so:CLASS4.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS5.h:

```
#ifndef _CLASS5_      //Khai báo tên file thư viện là CLASS5.h
#define _CLASS5_       //Định nghĩa tên file là CLASS5.h
#include   <iostream>
using namespace std;
class CLASS5 { // Khai báo các lớp một cách bình thường
public :
    void Function5(void){
        cout<<"Tap thao tac so:CLASS5.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS6.h:

```
#ifndef _CLASS6_      //Khai báo tên file thư viện là CLASS6.h
#define _CLASS6_       //Định nghĩa tên file là CLASS6.h
#include   <iostream>
using namespace std;
class CLASS6 { // Khai báo các lớp một cách bình thường
public :
    void Function6(void){
        cout<<"Tap thao tac so:CLASS6.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS7.h:

```
#ifndef _CLASS7_      //Khai báo tên file thư viện là CLASS7.h
#define _CLASS7_       //Định nghĩa tên file là CLASS7.h
#include   <iostream>
using namespace std;
class CLASS7 { // Khai báo các lớp một cách bình thường
public :
    void Function7(void){
        cout<<"Tap thao tac so:CLASS7.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS8.h:

```
#ifndef _CLASS8_      //Khai báo tên file thư viện là CLASS7.h
#define _CLASS8_       //Định nghĩa tên file là CLASS7.h
#include   <iostream>
using namespace std;
class CLASS8 { // Khai báo các lớp một cách bình thường
public :
    void Function8(void){
        cout<<"Tap thao tac so:CLASS8.h"<<endl;
    }
};

#endif
```

3.1.2. Xây dựng các Header File

Ví dụ Header File : CLASS9.h:

```
#ifndef _CLASS9_      //Khai báo tên file thư viện là CLASS9.h
#define _CLASS9_        //Định nghĩa tên file là CLASS9.h
#include   <iostream>
using namespace std;
class CLASS9 { // Khai báo các lớp một cách bình thường
public :
    void Function9(void){
        cout<<"Tap thao tac so:CLASS9.h"<<endl;
    }
};

#endif
```

Chương trình chính: Các em hoàn chỉnh chương trình như thế này.

```
#include <iostream>
#include "CLASS1.h" // Khai báo sử dụng thư viện trong file class1.h
.....
#include "CLASS9.h" // Khai báo sử dụng thư viện trong file class1.h
using namespace std;
class MAIN: public CLASS1,CLASS2,CLASS3,CLASS4,CLASS5 { //Lớp kế thừa tất cả các lớp khác
public:
    void Function(void){ char phim;
        do { system("cls");
            cout<<"1. Tap thao tac so" << endl;
            ....;
            cout<<"9. Tap thao tac da thuc" << endl;
            cout<<"0. Tro ve" << endl;
            cout<<"Lua chon:"; phim = cin.get();
            switch(phim){
                case '1': Function1(); break;
                .....
                case '9': Function5(); break;
            }
        } while(phim != '0');
    };
    int main(void) {
        MAIN X; X.Function();
        system("PAUSE"); return(0);
    }
}
```

3.2. Kiểm soát ngoại lệ (Exception)

Exception là một vấn đề xảy ra trong khi chương trình đang thực hiện. Một ngoại lệ là phản hồi từ một tình huống không mong muốn có thể ảnh hưởng đến động thái thực hiện của chương trình. Chính vì vậy, OPP cung cấp một phương pháp dịch chuyển động thái thực hiện của chương trình từ trạng thái này sang một trạng thái khác. Để kiểm soát ngoại lệ, các ngôn ngữ OPP cung cấp một cấu trúc lệnh dựa trên các từ khóa : *try, catch, throw*.

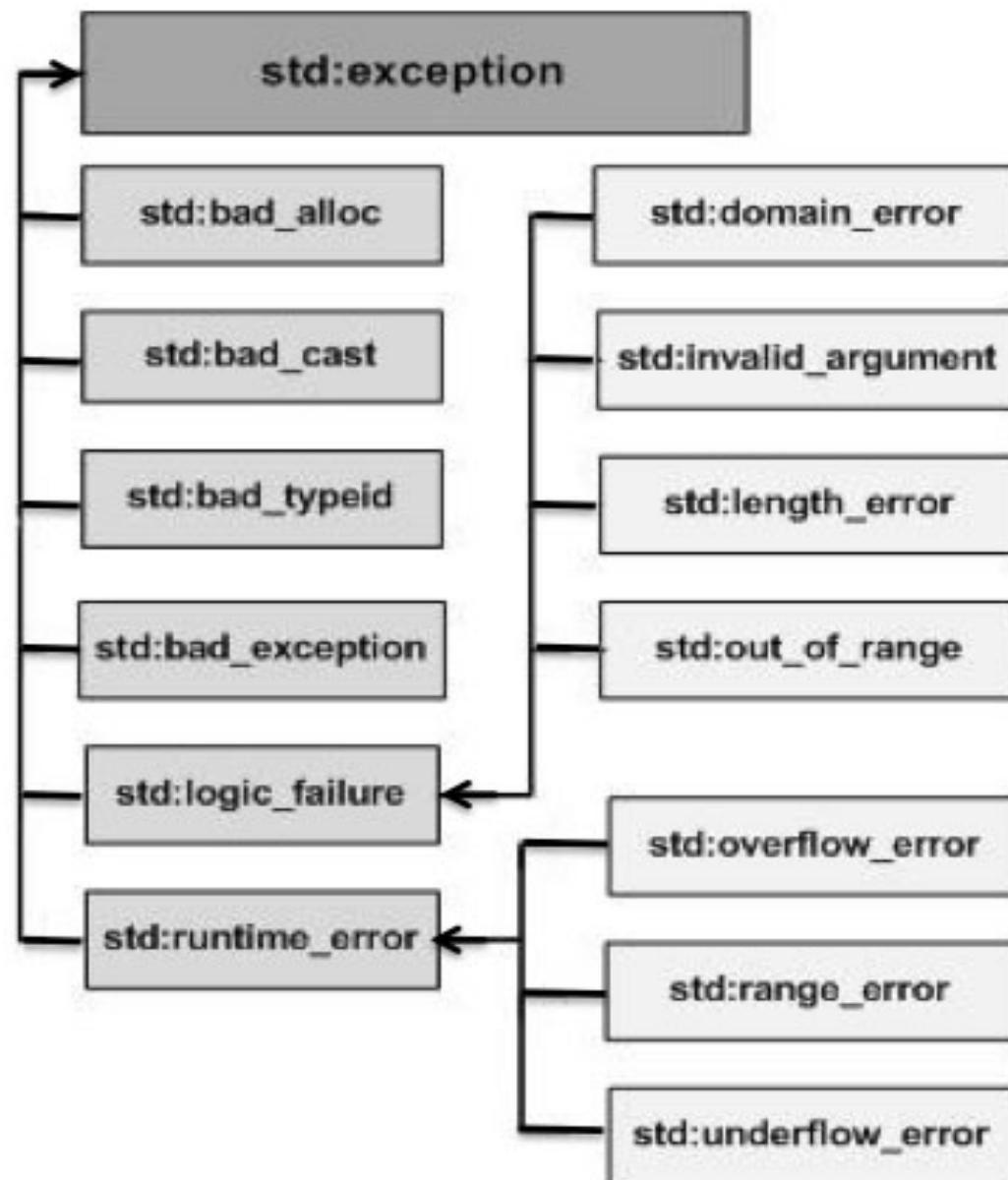
Ví dụ:

```
#include <iostream>
using namespace std;
double division(int a, int b) {
    if( b == 0 ) {
        throw "Division by zero condition!";
    }
    return (a/b);
}
int main () {
    int x = 50; int y = 0; double z = 0;
    try {
        z = division(x, y);
        cout << z << endl;
    } catch (const char* msg) {
        cerr << msg << endl;
    }
    return 0;
}
```

Ví dụ về ngoại lệ (Exception)

```
#include <iostream>
#include <cmath>
using namespace std;
int Test_Ngto(int k) { int p = (int) sqrt(k);
    for(int i=2; i<=p; i++) if (k%i==0) return(0);
    return(1);
}
int Test_Dx(int k) { int p = 0, n=k;
    while(k!=0) {int du = k%10; k=k/10; p = p*10 + du;}
    if (p==n) return 1;
    return(0);
}
void Test(int k) {
    if (Test_Ngto(k)&& !Test_Dx(k)) throw 1;
    else if (!Test_Ngto(k)&& Test_Dx(k))throw 2;
    else if (Test_Ngto(k)&& Test_Dx(k)) throw 3;
    else throw 4;
}
int main (){ int k = 131;
    try { Test(k); cout<<"k="<<k<<endl; }
    catch (const int msg){cout<<"k="<<k<<endl; cerr<<msg<<endl; }
    catch( const int msg) { cout<<"k="<<k<<endl;cerr<<msg; }
    catch( const int msg) { cout<<"k="<<k<<endl;cerr<<msg; }
        system("PAUSE");return 0;
}
```

Một số ngoại lệ cơ bản:



3.3. Bộ nhớ động

Một điều quan trọng đối với người lập trình là phải hiểu được bộ nhớ động làm việc thế nào. Đối với lập trình tiên tiến, bộ nhớ của một chương trình được chia thành hai phần :

- **Stack:** Mọi biến được khai báo trong chương trình, hàm đều được quản lý bằng stack.
- **Heap:** Không gian bộ nhớ chưa dùng đến có thể được phân bổ động mỗi khi có yêu cầu trong khi thực hiện chương trình.

Ví dụ:

```
#include <iostream>
using namespace std;
class Box {
public:
    Box() { cout << "Thực hiện Constructor!" << endl; }
    ~Box() { cout << "Thực hiện Destructor!" << endl; }
};
int main( ) {
    Box* myBoxArray = new Box[4];
    delete [] myBoxArray; // Delete array
    return 0;
}
```

Kết quả : - Constructor called : được triệu gọi 4 lần
- Destructor called : được triệu gọi 4 lần

3.5. Không gian tên (Namespace)

Namespace được thiết kế để giải quyết khó khăn trong việc sử dụng tên (tên biến, tên hàm, tên đối tượng) trong cùng một miền. Cùng một miền, ta có thể sử dụng cùng một tên biến, hàm, hoặc đối tượng mà vẫn điều hành được ta sử dụng tên nào trong số các tên giống nhau.

Định nghĩa namespace:

```
namespace <namespace-name> {  
    code declaration;  
}
```

Lời gọi đến namespace:

name:: code; //biến, hàm hoặc đối tượng

Ví dụ: Không sử dụng khai báo *using namespace std.*

```
#include <iostream>  
int main(void){  
    int a, b;  
    std::cin>>a; std::cout<<"\n a ="<<a; //sử dụng cin, cout trong std  
    std::cin>>b; std::cout<<"\n b ="<<b; //sử dụng cin, cout trong std  
    std::system("PAUSE"); return 0;  
}
```

Ví dụ. Tạo lập không gian tên riêng.

```
#include <iostream>
using namespace std;
// Định nghĩa không gian tên thứ nhất
namespace first_space {
    void func(){
        cout << "Inside first_space" << endl;
    }
}
// Định nghĩa không gian tên thứ hai
namespace second_space{
    void func() {
        cout << "Inside second_space" << endl;
    }
}
int main () {
    // Lời gọi hàm thuộc không gian tên thứ nhất.
    first_space::func();
    // Lời gọi hàm thuộc không gian tên thứ hai.
    second_space::func();
    return 0;
}
```

Ví dụ. Sử dụng không gian tên bên trong lớp.

```
#include <iostream>
using namespace std;
class One {
public:
    void Function(char ch){cout<<"Ky tu:"<<ch<<endl;}
};

class Two {
public:
    void Function(char *str){cout<<"String:"<<str<<endl;}
};

class Three:public One, Two {
public:
    void Function(long k){cout<<"So long:"<<k<<endl;}
    using One::Function; //Qua tai lop One::Function()
    using Two::Function; //Qua tai lop Two::Function()
};

int main(void){
    Three Sample;
    Sample.Function('P'); //Thực hiện Function trong lớp One
    Sample.Function("Hello World"); //Thực hiện Function trong lớp Two
    Sample.Function("123456789"); //Thực hiện Function trong lớp Three
    system("PAUSE");
}
```

3.6. Lập trình mẫu (Template)

Mẫu (template) là nền tảng của lập trình tiền hóa liên quan đến việc viết code theo phương pháp độc lập với các kiểu dữ liệu mô tả khác nhau. Một lớp mẫu như một bản thiết kế cho các lớp hoặc hàm tái sinh. Ví dụ, ta có thể định nghĩa một vector `<int>` khi đó vector sinh ra sẽ được qui định mỗi thành phần của nó là một số, vector `<string>` thì vector sinh ra sẽ được qui định mỗi thành phần của nó là một xâu.

Hàm mẫu được định nghĩa theo cú pháp sau:

```
template <typename T>
T      Tên-Hàm-Mẫu( T &biến1, T &biến2, ... ) {
    <Thân hàm mẫu>
    return( giá trị );
}
```

Lớp mẫu được định nghĩa theo cú pháp sau:

```
template      <class-type>      class   class-name {
    ..... .
}
```

Ví dụ về hàm mẫu.

```
#include <iostream>
#include <string>
using namespace std;
template <typename T>
T Max (T &a, T &b) {
    if (a>b ) return a;
    return b;
}
template <typename T>
void Swap( T &a, T &b) {  T temp;  temp = a; a = b; b = temp; }
int main (){
    int i = 39, j = 20; cout << "Max(i, j): " << Max(i, j) << endl; //max = 39
    double f1 = 13.5, f2 = 20.7;
    cout << "Max(f1, f2): " << Max(f1, f2) << endl; //max = 20.7
    string s1 = "Hello", s2 = "World";
    cout << "Max(s1, s2): " << Max(s1, s2) << endl; //max =“World”
    Swap( i, j);cout<<"i="<<i<<" j="<<j<<endl; //i=20; j =39
    Swap(s1,s2);cout<<"s1="<<s1<<" s2="<<s2<<endl;
    system("PAUSE");return 0;
}
```

Ví dụ về hàm mẫu.

```
#include <iostream>
#include <string>
#include <iomanip>
using namespace std;
template <typename T>
T Display(T a[], int n) { cout<<"Kết quả:";
    for(int i=0; i<n; i++){ cout<<a[i]<<setw(5);}
}
template <typename T>
T Sort(T a[], int n) {
    for(int i=0; i<n-1; i++){
        for (int j=i+1; j<n; j++){
            if (a[i]>a[j]) {
                T temp = a[i]; a[i]=a[j]; a[j]=temp;
            }
        }
    }
}
int main(void){
    int A[] = {9, 7, 12, 8, 6, 5}, n=6; Sort(A,n); Display(A,n);cout<<endl;
    float B[] = {9.5, 7.2, 12.8, 8.92, 6.18, 5.34}; Sort(B,n); Display(B,n);cout<<endl;
    char str[] ="BCDAGHKIBC"; n=strlen(str); Sort(str,n); Display(str,n);cout<<endl;
    system("PAUSE");
}
```

Ví dụ về lớp mẫu: Tham biến của hàm tương thích với mọi kiểu dữ liệu.

```
#include <iostream>
using namespace std;
template <class Bien>
class Test {
    public:
        Test();
        ~Test();
        Bien Data(Bien);
};
template <class Bien>
    Bien Test<Bien>::Data(Bien Var0){ return Var0; }
template <class Bien>
    Test<Bien>::Test(){cout<<"Khởi tạo constructor"<<endl;}
template <class Bien>
    Test<Bien>::~Test(){cout<<"Thực hiện Destructor"<<endl;}
int main(void){
    Test<int> Var1; Test<double> Var2;Test<char> Var3;Test<char*> Var4;
    cout<<"\n Lớp template thích hợp với kiểu bất kỳ"<<endl;
    cout<<"Var1, int = "<<Var1.Data(100)<<endl; //Số nguyên được chấp thuận
    cout<<"Var2, double = "<<Var2.Data(1.234)<<endl; //Số thực được chấp thuận
    cout<<"Var3, char = "<<Var3.Data('K')<<endl; //Ký tự được chấp thuận
    cout<<"Var4, char* = "<<Var4.Data("Day la String")<<endl<<endl; //String OK
    system("PAUSE");
}
```

Ví dụ về lớp mẫu: Tham biến của hàm tương thích với mọi kiểu dữ liệu.

```
#include <iostream>
using namespace std;
template <class Bien>
class Test {
public:
    Bien Swap(Bien &, Bien &); //đổi nội dung hai biến bất kỳ
};
template <class Bien>
Bien Test<Bien>::Swap(Bien &x, Bien &y){
    Bien Temp = x; x = y; y=Temp;
}
int main(void){
    Test<int> X1; Test<char> X2;Test<string> X3;
    int a=10,b=20;X1.Swap(a,b);
    cout<<"Kieu int OK: a = "<<a<<" b="<<b<<endl;
    char x='C',y ='A';X2.Swap(x,y);
    cout<<"Kieu char OK: x = "<<x<<" y="<<y<<endl;
    string s1="Hello", s2="world";X3.Swap(s1,s2);
    cout<<"Kieu string OK: s1 = "<<s1<<" y="<<s2<<endl;
    system("PAUSE");return 0;
}
```

Ví dụ về lớp mẫu: Tham biến của hàm tương thích với mọi kiểu dữ liệu.

```
#include <iostream>
using namespace std;
template <class Bien>
class Test {
public:
    Bien Swap(Bien &, Bien &);
    Bien Sort(Bien [], int);
    Bien Result(Bien [], int);
};
template <class Bien>
Bien Test<Bien>::Swap(Bien &x, Bien &y){  Bien Temp = x; x = y; y=Temp;  }
template <class Bien>
Bien Test<Bien>::Sort(Bien A[], int n){
    for(int i=0; i<n-1; i++)
        for (int j=i+1; j<n; j++) if (A[i]>A[j]) Swap(A[i], A[j]);
}
template <class Bien>
Bien Test<Bien>::Result(Bien A[], int n){  cout<<"Ket qua:";
    for(int i=0; i<n-1; i++) cout<<A[i]<<setw(6);
    cout<<endl;
}
int main(void){ Test<int> X1; Test<float > X2;Test<char > X3;
    int A[] = {9, 7, 12, 8, 6, 5}, n=6; X1.Sort(A,n); X1.Result(A,n);
    float B[]={9.5, 7.6, 4.2, 10.8, 7.3, 8.1 }; n =6; X2.Sort(B,n); X2.Result(B,n);
    char str[]="XYABZWCD"; n=strlen(str); X3.Sort(str,n); X3.Result(str,n);
    system("PAUSE");return 0;
}
```

3.7. Lập trình mẫu với STL

Ngôn ngữ C++ cung cấp một thư viện chuẩn dưới dạng các lớp mẫu được gọi là STL (Standard Template Library). Sử dụng STL trong lập trình sẽ giúp ta giảm được thời gian và chi phí. Tuy nhiên, sử dụng STL cùng với việc hiểu được bản chất của STL mới là chìa khóa của sự thành công trong lập trình. Ví dụ: muốn sử dụng tốt thư viện Algorithm ta cần phải hiểu một thuật toán nào đó thực hiện việc gì, nó thực hiện thế nào và kết quả thực hiện của nó ra sao. Một số lớp mẫu chuẩn dưới đây ta cần phải làm quen trong khuôn khổ của giáo trình này:

- | | |
|---|------------------------|
| 1. Lớp mẫu array được khai báo với header file | : #include <array> |
| 2. Lớp mẫu vector được khai báo với header file | : #include <vector> |
| 3. Lớp mẫu set được khai báo với header file | : #include <vector> |
| 4. Lớp mẫu iostream được khai báo với header file | : #include <iostream> |
| 5. Lớp mẫu fostream được khai báo với header file | : #include <fstream> |
| 6. Lớp mẫu ofstream được khai báo với header file | : #include <ofstream> |
| 7. Lớp mẫu iomanip được khai báo với header file | : #include <iomanip> |
| 8. Lớp mẫu string được khai báo với header file | : #include <string> |
| 9. Lớp mẫu complex được khai báo với header file | : #include <complex> |
| 10. Lớp mẫu algorithm được khai báo với header file | : #include <algorithm> |
| 11. Lớp mẫu cmath được khai báo với header file | : #include <cmath> |
| 12. Lớp mẫu cstring được khai báo với header file | : #include <cstring> |
| 13. Lớp mẫu cstdio được khai báo với header file | : #include <cstdio> |
| 14. Lớp mẫu cconio được khai báo với header file | : #include <cconio> |
| 15. Lớp mẫu cstdlib được khai báo với header file | : #include <cstdlib> |

Tài liệu tham khảo và ví dụ các em tham khảo tại:

Ví dụ. Lập trình mẫu với STL (complex):

```
#include <iostream>      // std::cout
#include <complex>      // std::complex
#include <iomanip>
using namespace std;
int main () {
    complex <double> X(2.0,3.0), Y(1.0,2.0), Z; //khai báo các số phức X, Y, Z
    cout<<"X=" << X << setw(10) << "Y=" << Y << endl;
    Z = X+Y; cout<<"Tổng Z=" << Z << endl;
    Z = X-Y; cout<<"Hiệu Z=" << Z << endl;
    Z = X*Y; cout<<"Tích Z=" << Z << endl;
    Z = X/Y; cout<<"Thươngg Z=" << Z << endl;
    X+=Y; cout<<"X= X+Y: " << X << endl;
    X-=Y; cout<<"X= X-Y: " << X << endl;
    X*=Y; cout<<"X= X*Y: " << X << endl;
    X/=Y; cout<<"X= X/Y: " << X << endl;
    Z = pow(X,2); cout<<"Lũy thừa=" << Z << endl;
    Z = sqrt(Z); cout<<"Căn bậc 2 Z=" << Z << endl;
    Z = sin(X); cout<<"Sin(X) =" << Z << endl;
    system("PAUSE");
}
```

Ví. Lập trình mẫu với STL (vector):

```
#include <vector>
#include <iostream>
using namespace std;
int main(void){ unsigned int i;
    vector <int> vec1, vec2, vec3;
    cout<<"vector thu nhat: ";
    for(i=1; i<=10; ++i) vec1.push_back(i);
    for(i=0; i<vec1.size(); ++i) cout<<vec1[i]<<' ';
    cout<<"\n vector thu hai: ";
    for(i=11; i<=20; ++i) vec2.push_back(i);
    for(i=0; i<vec2.size(); ++i) cout<<vec2[i]<<' ';
    cout<<"\n vector thu ba: ";
    for(i=1; i<=10; ++i) vec3.push_back(i);
    for(i=0; i<vec3.size(); ++i) cout<<vec3[i]<<' ';
    cout<<endl<<endl;
    if(vec1 != vec2) cout<<"vec1 khac vector vec2"<<endl;
    if(vec1 == vec3) cout<<"vec1 bang vec3"<<endl;
    if(vec2 > vec1) cout<<"vec2 lon hon vec1."<<endl;
    else cout<<"vec2 khong lon hon vec1."<<endl;
    system("PAUSE");return 0;
}
```

3.7. Lập trình mẫu với STL

Ngôn ngữ C++ cung cấp một thư viện chuẩn dưới dạng các lớp mẫu được gọi là STL (Standard Template Library). Sử dụng STL trong lập trình sẽ giúp ta giảm được thời gian và chi phí. Tuy nhiên, sử dụng STL cùng với việc hiểu được bản chất của STL mới là chìa khóa của sự thành công trong lập trình. Ví dụ: muốn sử dụng tốt thư viện Algorithm ta cần phải hiểu một thuật toán nào đó thực hiện việc gì, nó thực hiện thế nào và kết quả thực hiện của nó ra sao. Một số lớp mẫu chuẩn dưới đây ta cần phải làm quen trong khuôn khổ của giáo trình này:

- | | |
|---|------------------------|
| 1. Lớp mẫu array được khai báo với header file | : #include <array> |
| 2. Lớp mẫu vector được khai báo với header file | : #include <vector> |
| 3. Lớp mẫu set được khai báo với header file | : #include <vector> |
| 4. Lớp mẫu iostream được khai báo với header file | : #include <iostream> |
| 5. Lớp mẫu fostream được khai báo với header file | : #include <fstream> |
| 6. Lớp mẫu ofstream được khai báo với header file | : #include <ofstream> |
| 7. Lớp mẫu iomanip được khai báo với header file | : #include <iomanip> |
| 8. Lớp mẫu string được khai báo với header file | : #include <string> |
| 9. Lớp mẫu complex được khai báo với header file | : #include <complex> |
| 10. Lớp mẫu algorithm được khai báo với header file | : #include <algorithm> |
| 11. Lớp mẫu cmath được khai báo với header file | : #include <cmath> |
| 12. Lớp mẫu cstring được khai báo với header file | : #include <cstring> |
| 13. Lớp mẫu cstdio được khai báo với header file | : #include <cstdio> |
| 14. Lớp mẫu cconio được khai báo với header file | : #include <cconio> |
| 15. Lớp mẫu cstdlib được khai báo với header file | : #include <cstdlib> |

Tài liệu tham khảo và ví dụ các em tham khảo tại: <http://www.cplusplus.com/reference>