KỸ THUẬT LẬP TRÌNH C/C++
Lập trình khái quát và STL

Thi-Lan Le
Thi-Lan.Le@mica.edu.vn; lan.lethi1@hust.edu.vn
Webpage: http://www.mica.edu.vn/perso/Le-Thi-Lan

Khái niệm

- Đôi khi ta muốn viết một lần nhưng có thể tạo ra các hàm với tham số thuộc nhiều kiểu khác nhau, thay vì phải viết chồng nhiều hàm tương tự nhau
  - ```
    int max(int a, int b)          { return a>b ? a:b; }
    double max(double a, double b)       { return a>b ? a:b; }
    float max(float a, float b)   { return a>b ? a:b; }
    ```

→ lập trình ở mức độ khái quát cao hơn: coi kiểu của biến cũng là tham số (type parameterization)

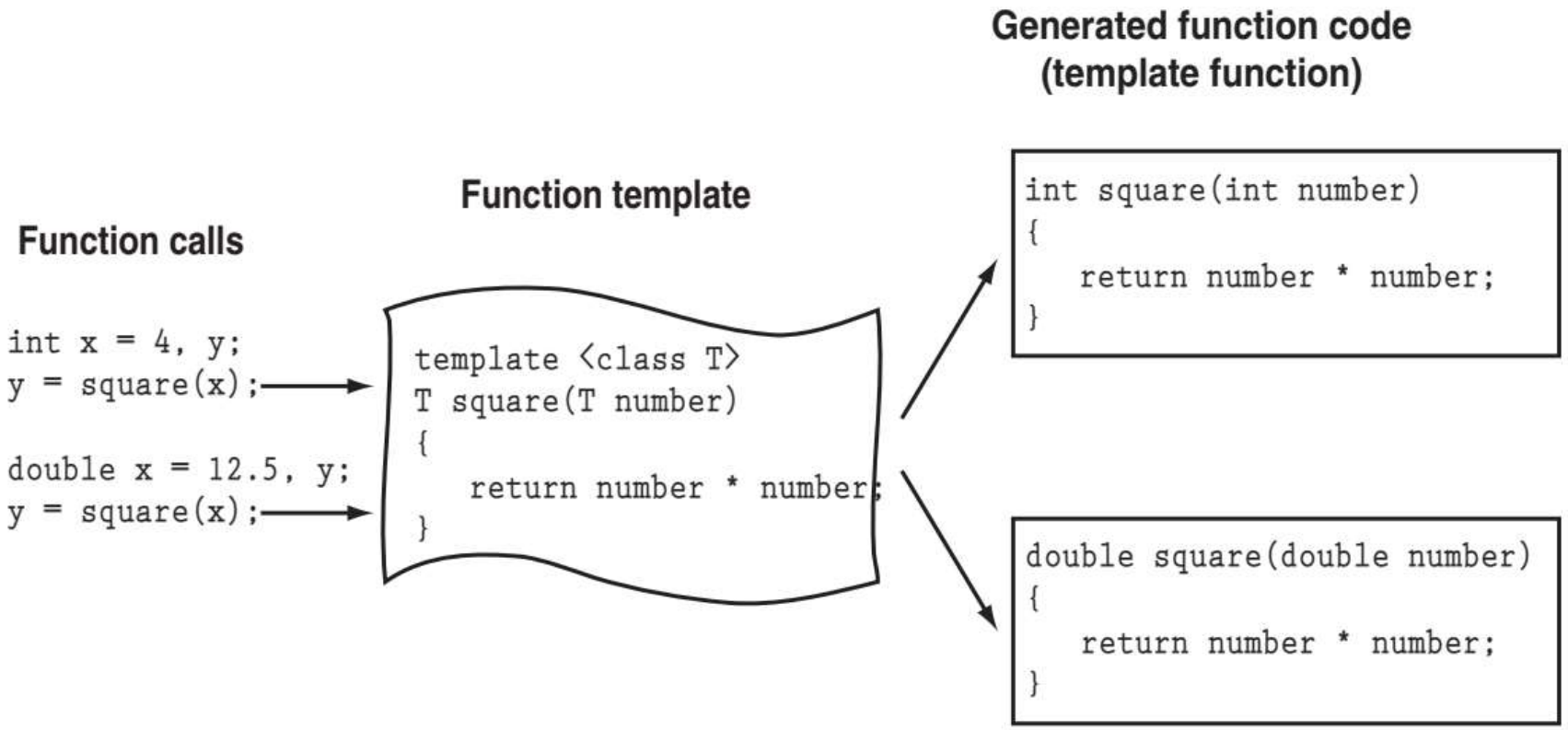
```
template <class T>
 T mymax (T a, T b)
 {
  return a>b ? a:b;
 }
```
```

Khái niệm

- Khuôn mẫu hàm (function template): là khái niệm giúp định nghĩa những hàm mà chưa xác định kiểu của các tham số
  - Có thể hiểu là viết gộp chung các hàm chồng giống nhau về mặt thuật toán
  - Kiểu của các tham số là tham số của khuôn mẫu
- Ví dụ 2:

```
template <class T>
T square(T number)
{
return number * number;
}
```

Khái niệm


Generated function code
(template function)
Function template
Function calls
int x = 4, y;
y = square(x);
double x = 12.5, y;
y = square(x);
template <class T>
T square(T number)
{
return number * number;
}
int square(int number)
{
return number * number;
}
double square(double number)
{
return number * number;
}

Định nghĩa hàm khái quát

```
#include <iostream>
#include <vector>
using namespace std;
template <class T>
 void myswap(T& a, T& b)
 {
       T c = a; a = b; b = c;
 }
int main()
{
     int a=10, b=5;
     cout<<a<<"  "<<b<<endl;
     myswap(a, b);
     cout<<a<<"  "<<b<<endl;
     return 0;
}
```

Định nghĩa hàm khái quát

```
template <class T1, class T2, class T3>
 void echoAndReverse(T1 a1, T2 a2, T3 a3)
 {
  cout << "Original order is: "
  << a1 << " " << a2 << " " << a3 << endl;
  cout << "Reversed order is: "
  << a3 << " " << a2 << " " << a1 << endl;
 }

 int main()
 {
  echoAndReverse("Computer", 'A', 18);
  echoAndReverse("One", 4, "All");
 return 0;
 }
```

Bài tập

- Viết hàm tổng quát sắp xếp các giá trị trong một mảng cho trước.
  - template <class T>
  - void Sort_array(T arr[ ], int size)

Lớp khái quát

- Tương tự như hàm khái quát

```
template <class T>
class Class_Name
{
};
```

Lớp khái quát

```
class Rectangle
{
private:
        double width;
        double length;
        double area;
public:
        void setData(double w, double l)
                { width = w; length = l;}
        void calcArea()
                { area = width * length; }
        double getWidth()
                { return width; }
        double getLength()
                { return length; }
        double getArea()
                { return area; }
};
```

Lớp khái quát

- Xây dựng lớp Rectangle cho các kiểu dữ liệu khác nhau:

```
template <class T>
class Rectangle
{
    private:
            T width;
            T length;
            T area;
    public:
    void setData(T w, T l)
            { width = w; length = l;}
    void calcArea()
            { area = width * length; }
    T getWidth()
            { return width; }
    T getLength()
            { return length; }
    T getArea()
            { return area; }
};
```

Lớp khái quát

```
int main()
    {
     Rectangle<float> rec;
     rec.setData(2.5, 5.1);
     rec.calcArea();
     cout<<rec.getArea()<<endl;
     Rectangle<int> rec_int;
     rec_int.setData(2, 5);
     rec_int.calcArea();
     cout<<rec_int.getArea()<<endl;
     return 0;
    }
```

STL là gì?

- STL (viết tắt của ***Standard Template Library***), là tập thư viện chứa các lớp mẫu (template classes) cung cấp một số cấu trúc dữ liệu như: list, array, vector, set,…; cũng như các giải thuật cơ bản liên quan.

Các thành phần chính của STL

Thành phần	Ý nghĩa
Container	Chứa tập các đối tượng thuộc kiểu nào đó. Có nhiều loại container khác nhau như vector, deque, list, map, …
Algorithm	Chứa cài đặt cho các giải thuật cơ bản như sắp xếp, tìm kiếm, biến đổi (transform) cho các phần tử trong container.
Iterator	Được sử dụng để duyệt qua các phần tử trong các container.
Functor	Là đối tượng được sử dụng như hàm.

Các loại container

Tên	Ý nghĩa
vector	Cấu trúc danh sách được tổ chức bằng cấu trúc lưu trữ tuần tự.
list	Cấu trúc danh sách được tổ chức bằng cấu trúc lưu trữ móc nối kép
deque	(Double ended queue): cấu trúc hàng đợi, là cấu trúc danh sách được tổ chức đặc biệt để phù hợp với các thao tác bổ sung và loại bỏ ở hai đầu.
stack	Cấu trúc ngăn xếp
queue	Cấu trúc hàng đợi
array	Lớp mảng nhằm thay thế cho kiểu mảng thông thường trong C/C++ mà có một số hạn chế khi truyền tham số thường mất thông tin về kích thước mảng.

Các loại container (tiếp)

Tên	Ý nghĩa
pair	Cấu trúc gồm 2 thành phần <first, second>
map	Cấu trúc danh sách mà mỗi phần tử lại gồm một cặp <key, value>. Thực chất chính là một danh sách các phần tử kiểu *pair*, mà trong đó các phần tử được sắp xếp theo giá trị *key* (phần *first* trong một *pair*)
set	Cấu trúc tập hợp

Các thao tác cơ bản chung

Tên	Ý nghĩa
size();	trả về kích thước (số phần tử) của cấu trúc
push_back();	bổ sung một phần tử vào phía cuối
insert();	chèn một phần tử vào vị trí bất kỳ
pop_back();	loại bỏ một phần tử ở phía cuối
front();	trả về phần tử ở đầu
back();	trả về phần tử ở cuối
empty();	kiểm tra xem cấu trúc có rỗng không

Ví dụ sử dụng lớp vector

```
#include <iostream>
#include <vector>

using namespace std;

int main()
{
    cout << "Hello vector!" <<
endl;
    vector<int> v;  //Khai bao
vector
    //Bo sung vao vector
    for (int i=0;i<10;i++)
        v.push_back(i+1);

    //Truy nhap vao vector su
dung ham at()
    for (int i=0;i<v.size();i++)
        cout<<v.at(i)<<" ";
 cout<<endl;

    //Truy nhap vao vector su
dung toan tu []
    for (int i=0;i<v.size();i++)
        cout<<v[i]<<" ";

    cout<<endl;
    //Loai bo dan cac phan tu
khoi vector
    while (!v.empty()){
        cout<<v.back()<<" ";
        v.pop_back();
    }
    cout<<endl;
    return 0;
}
```

Iterator

- Là kiểu con trỏ được xây dựng trong các container để trỏ đến phần tử trong đó.
- Nó được dùng để đơn giản hóa các thao tác duyệt các phần tử trong các container.

Một số thao tác cơ bản liên quan đến iterator

Tên	Ý nghĩa
begin();	hàm thành viên của một container, nhằm trả về iterator mà trỏ đến phần tử đầu tiên.
end();	hàm thành viên của một container, nhằm trả về iterator mà trỏ đến phần tử cuối cùng.
next(it, n);	trả về iterator mà trỏ vào phần tử kế tiếp thứ n mà sau iterator it đang trỏ đến.
it++;	tương đương câu lệnh: it = next(it); với it là một iterator.
prev(it, n);	ngược lại của next(it, n).

Ví dụ sử dụng iterator

```
#include <iostream>
#include <vector>

using namespace std;

int main()
{
    cout << "Hello vector!" <<
endl;
    vector<int> v;  //Khai bao
vector
    //Bo sung vao vector
    for (int i=0;i<10;i++)
v.push_back(i+1);

    //Truy nhap vao vector su
dung ham at()
    for (int i=0;i<v.size();i++)
cout<<v.at(i)<<" ";
    cout<<endl;
```

```
 //Truy nhap vao vector su dung
iterator
    vector<int>::iterator it;
    for (it = v.begin(); it !=
v.end(); it++)
        cout<<*it<<" ";
    cout<<endl;

    return 0;
}
```

Các giải thuật cơ bản

Tên	Ý nghĩa
sort();	Sắp xếp dãy các phần tử
binary_search()	Tìm kiếm nhị phân, yêu cầu dãy phải được sắp xếp trước.
...	

Ví dụ 1 về hàm sort(): Sắp xếp tăng dần

```
#include <iostream>
#include <algorithm>

using namespace std;

void show(int a[], int n)
{
    for(int i = 0; i < n;
++i)
        cout << '\t' << a[i];
    cout<<endl;
}
```

```
int main()
{
    int n = 9;
    int a[n]= {1, 5, 8, 9, 6,
2};
    cout << "The array before
sorting is : \n";
    show(a,n);

    sort(a, a+n);

    cout << "\n The array
after sorting is : \n";
    show(a,n);

    return 0;

}
```

Ví dụ 2 về hàm sort():
Sắp xếp chẵn trước, lẻ sau

```
#include <iostream>
#include <algorithm>
using namespace std;
void show(int a[], int n)
{
   for(int i = 0; i < n; ++i)
       cout << '\t' << a[i];
   cout<<endl;
}
//So chan roi den so le
bool rule(int x, int y){
 //x, y cùng chẵn hoặc cùng lẻ
   if (x*y%2 || (x%2==0 && (y%2==0)))
      return (x<y);
   else
   //x chẵn < y lẻ
   if (!(x%2) and y%2)
       return true;
   else //x lẻ > y chẵn
       return false;
}
```

```
int main()
{
    int n = 9;
    int a[n]= {1, 5, 8, 9, 6, 7, 3, 4, 2};
    cout << "The array before sorting is :
\n";
    show(a,n);

    sort(a, a+n, rule);

    cout << "\n The array after sorting is :
\n";
    show(a,n);

    return 0;

}
```