Lecture 21

Template II

Class Templates

Prof. Hyeong-Seok Ko
Seoul National University
Graphics & Media Lab

Contents

- Class Template (16.1.2)
- Instantiation (16.2 until 16.2.1)
- Class Template Member Functions (16.4.1)
- Template Argument for Non-Type Parameters (16.1.5, 16.4.2)

Class Template

- **Class template** provides a means to re-use the class definition for different data types without rewriting for each data type.
  - Avoids code duplication

Example 1. Containers

- It is desirable that general containers (such as array or list) can hold objects of different types.
  - It can be achieved by using **class templates.**

```
#include <vector>
#include <list>
#include <stack>
#include <queue>

using namespace std;

void main() {
   vector<float> f_vec;
   list<float> f_list;
   stack<float> f_stack;
   queue<float> f_queue;
}
```

Example 2. 2D and 3D Vectors

- It is desirable 2D and 3D vectors can hold values of different types.
  - It can be achieved by using **class templates.**

```
void main() {
   vec2<int> v2_i;
   vec2<float> v2_f;
   vec2<double> v2_d;

   vec3<int> v3_i;
   vec3<float> v3_f;
   vec3<double> v3_d;
}
```

Vec2 Class Template

- Definition of Vec2 class template

```
template<typename T>
class Vec2 {
public :
   Vec2();

   void set(const Vec2&);
   void set(const T& a, const T& b);

   T& operator[](std::size_t i) const;

   T val[2];
};
```

Instantiation of Class Template

- When we use a class template, we must explicitly specify arguments for the template parameters.

```
void main() {
   Vec2<int> v2_i;
   Vec2<float> v2_f;
   Vec2<double> v2_d;
}
```

Putting the Definition of a Class Template Member Function Elsewhere

- A precaution is needed when putting the definition of a class template member function outside of the class definition?

```
template<typename T>
class Vec2 {
public :
   Vec2() {}
   void set(const T& a, const T& b);

   T val[2];
};

template<typename T>
void Vec2<T>::set(const T& a, const T& b) {
   val[0] = a;
   val[1] = b;
}
```

Class Template Inline Member Functions

- How to define a class template `inline` member function?
  - “`inline`” should appear in both prototype and definition.

```
template<typename T>
class Vec2 {
public :
    Vec2() {}
    inline void set(const T& a, const T& b);

    T val[2];
};

template<typename T>
inline void Vec2<T>::set(const T& a, const T& b) {
    val[0] = a;
    val[1] = b;
}
```

Template Argument for Non-type Parameters

- Definition of class Vec for arbitrary dimension

```
template<typename T, int DIM>
class Vec {
public :
   Vec() {}

   T val[DIM];
};

void main() {
   Vec<float, 2>      a;     // 2-dimensional float vector
   Vec<double, 3>     b;     // 3-dimensional double vector
}
```