Lecture 6

Object-Oriented Programming II

Defining Classes

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

Contents

- Class definition (12.1.1, 12.1.3)
- Class object (12.1.5, 5.6)
- `this` pointer (12.2)

Introduction to Class Definition

- Member variables
- Member functions

```
class Box {
public:
   Box(double h, double w, double l) : height(h), width(w), length(l) {}

   double volume() { return height*width*length; }
   void print() {
      cout << height << " " << width << " " << length << endl;
   }

private:
   double height, width, length;
};
```

member functions

member variables

Introduction to Class Definition

- Member variables
- Member functions

```
class Box {
public:
   Box(double h, double w, double l) : height(h), width(w), length(l) {}

   double volume() { return height*width*length; }
   void print() {
      cout << height << " " << width << " " << length << endl;
   }

private:
   double height, width, length;
};
```

Creating a Class Object

- Member variables
- Member functions

```
class Box {
public:
   Box(double h, double w, double l) : height(h), width(w), length(l) {}

   double volume() { return height*width*length; }
   void print() {
      cout << height << " " << width << " " << length << endl;
   }

   double height, width, length;
};

void main() {
   Box box;                                    // create a class object
   box.height = box.width = box.length = 10; // access member variables
   box.print();                                // call member function
}
```

Member Functions Can be Defined Outside

- Member variables
- Member functions

```
class Box {
public:
   Box(double h, double w, double l) : height(h), width(w), length(l) {}

   double volume() { return height*width*length; }
   void print() {
      cout << height << " " << width << " " << length << endl;
   }

   double height, width, length;
};
```

Member Functions Can be Defined Outside

- Member variables
- Member functions

```cpp
class Box {
public:
   Box(double h, double w, double l) : height(h), width(w), length(l) {}

   double volume() { return height*width*length; }
   void print();

   double height, width, length;
};

void Box::print() {
   cout << height << " " << width << " " << length << endl;
}
```

Creating an Object without a Pointer

- When an object is created in the stack memory, dot is used to access the class members.

```
#include <iostream>

class Box {
public:
   Box() {}

   double volume() { return height*width*length; }
   void print() {
      std::cout << height << " " << width << " " << length << std::endl;
   }

   double height, width, length;
};

void main() {
   Box box;                                    // define a class object
   box.height = box.width = box.length = 10; // access member variable in Box class
   box.print();                                // call member function in Box class
}
```

Creating an Object with a Pointer

- When an object is created in the heap memory (i.e., using **new**), the arrow operators are used for accessing class members.

```
#include <iostream>

class Box {
public:
   Box() {}

   double volume() { return height*width*length; }
   void print() {
      std::cout << height << " " << width << " " << length << std::endl;
   }

   double height, width, length;
};

void main() {
   Box * box = new Box();                          // define a class object using new
   box->height = box->width = box->length = 10;  // use arrow operator
   box->print();                                   // call member function
}
```

How to Initialize the Object Created with ‘new’?

```
#include <iostream>

class Box {
public:
   Box(double h, double w, double l)
   {
         height = h;
         width = w;
         length = l;
   }

   double height, width, length;
};

void main() {
   Box * box = new Box(1,2,3);
}
```

this pointer

- The **`this`** pointer points to the object for which the member functi on is called.

```
#include <iostream>

class Box {
public:
   Box(double height, double width, double length)
   {
     this->height = height; // height = height; will not work
     this->width = width;   // because local var will shadow member var
     this->length = length;
   }

   double height, width, length;
};

void main() {
   Box * box = new Box(1,2,3);
}
```