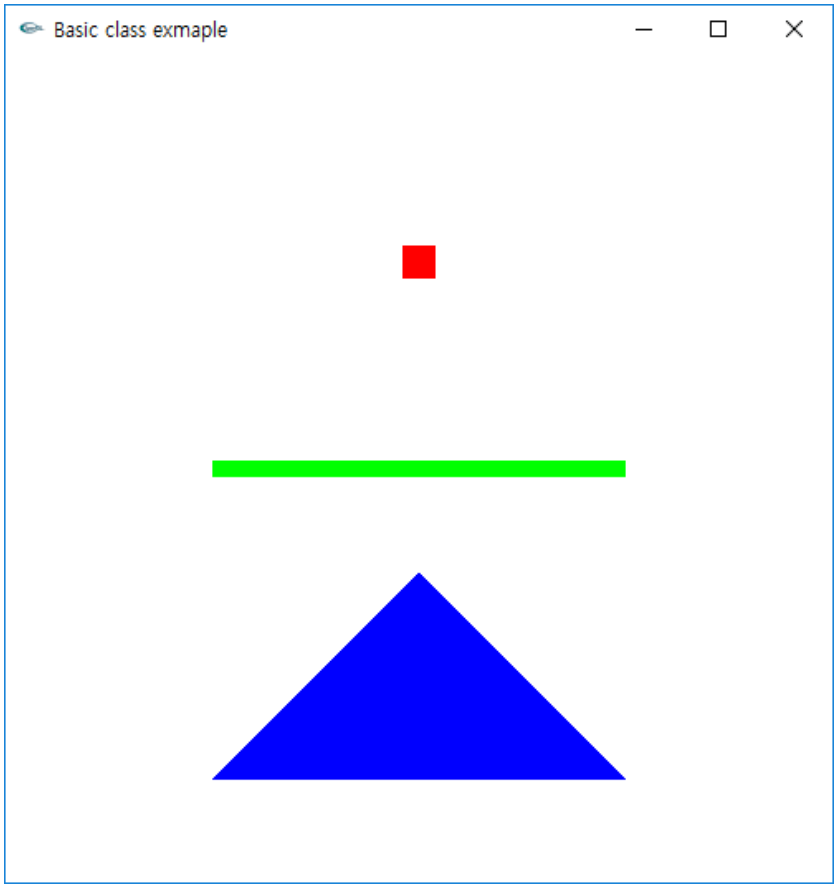
LAB I
Week 05

Seoul National University
Graphics & Media Lab
HyeonSeung Shin

Today's Mission

- Define class to draw primitives



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

```
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
}
```

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);
}
```

Member Functions

- Components of a member function
  - Return type
  - Function name
  - Parameters
  - Function body

```
#include <iostream>

class Box {
public:
   Box() {}
   void set(double h, double w, double l) {
      height = h; width = w; length = l;
   }
   double volume() const { return height*width*length; }

   double height, width, length;
};

void main() {
   Box box;
   box.set(1,1,1);
   std::cout << box.volume() << std::endl;
}
```

member functions

Member Functions

- `const` member functions
  - A `const` member function cannot change the content of the object for which the member function is called.

```
class Box {
public:
   Box() {}
   void set(double h, double w, double l) const {
      height = h; width = w; length = l;
   }
   double volume() const { return height*width*length; }

   // volume() function may read but not write to the data members of the objects

   double height, width, length;
};

 void main() {
    Box box;
    box.set(1,1,1);
    std::cout << box.volume() << std::endl;
 }
```

Error !

Inline Member Functions

- A member function whose definition lies completely within the class definition. (e.g., `void set(…)`)
- The function definition can come elsewhere if an explicit inline declaration is made.

```
class Box {
public:
   Box() {}
   void set(double h, double w, double l) {
      height = h; width = w; length = l;
   }
   double area() const;
   inline double volume() const;

   double height, width, length;
};

double Box::area() const {
   return 2*(height*width + width*length + length*height);
}

inline double Box::volume() const {
   return height*width*length;
}
```

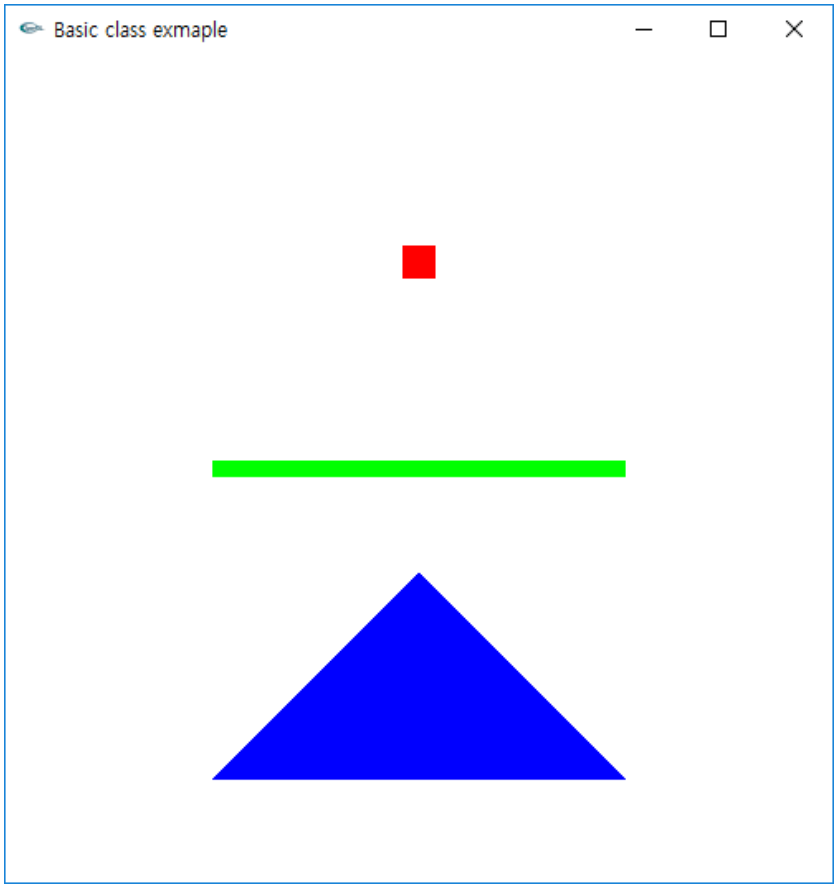
Defined within the class, thus inline

Declaration of an inline function

Definition of the inline function

Today's Mission

- Define class to draw primitives



Code example

```
class Color {
public:
	void setColor(float r, float g, float b) {
		// Set Color (RGB)
		this->r = r;
		this->g = g;
		this->b = b;
	}
	float getRed() const {
		return r;
	}
	float getGreen() const {
		return g;
	}
	float getBlue() const {
		return b;
	}

	float r, g, b;
};
```

Code example

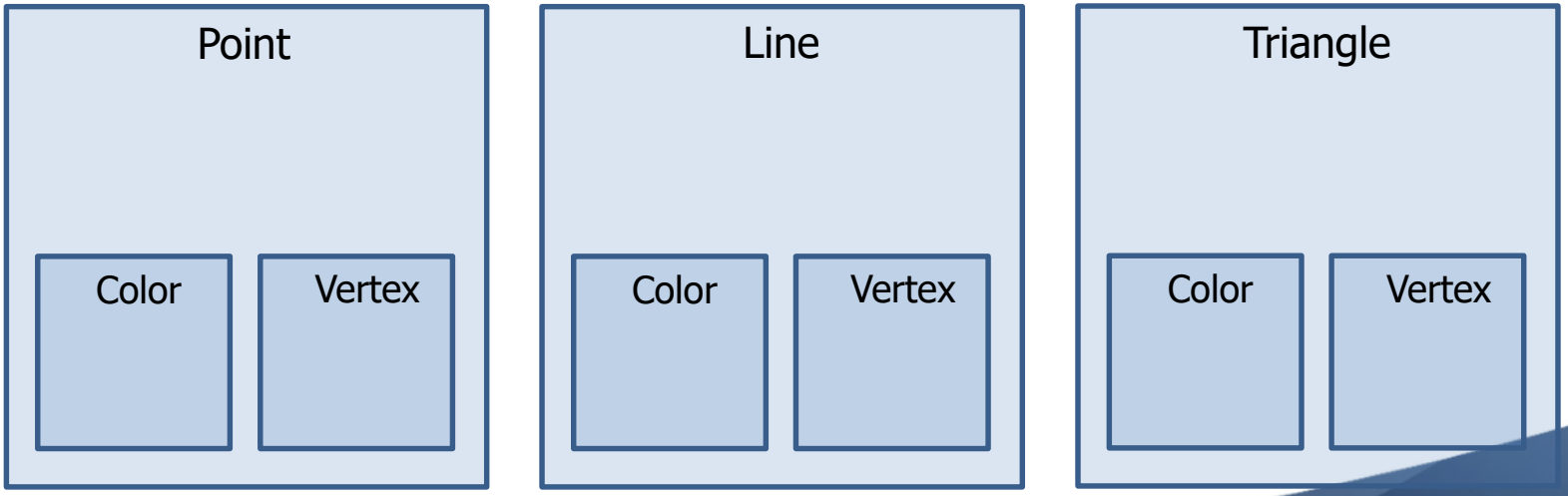
```
class Triangle {
public:
	Triangle(float x1, float y1, float z1, float x2, float y2, float z2, float x3, float y3, float z3) {
		// Initialize triangle's vertices
		vertex[0].setVertex(x1, y1, z1);
		vertex[1].setVertex(x2, y2, z2);
		vertex[2].setVertex(x3, y3, z3);
	}
	void setColor(float r, float g, float b) {
		// Set triangle's color
		color.setColor(r, g, b);
	}
	void draw() const {
		// Draw triangle
		glBegin(GL_TRIANGLES);
			glColor3f(color.getRed(), color.getGreen(), color.getBlue());
			glVertex3f(vertex[0].getX(), vertex[0].getY(), vertex[0].getZ());
			glVertex3f(vertex[1].getX(), vertex[1].getY(), vertex[1].getZ());
			glVertex3f(vertex[2].getX(), vertex[2].getY(), vertex[2].getZ());
		glEnd();
	}

	Color color;
	Vertex vertex[3];
};
```

Code example

```
void renderScene(void) {
	glClearColor(1, 1, 1, 1);

	glClear(GL_COLOR_BUFFER_BIT);

	// Declare object of triangle
	Triangle triangle(0, -0.25, 0, -0.5, -0.75, 0, 0.5, -0.75, 0);

	// Set color of triangle
	triangle.setColor(0, 0, 1);

	// Draw triangle
	triangle.draw();

	glutSwapBuffers();
}
```

Today's Mission

- Implementation (1)
  - Define class
    - Color
    - Vertex
    - Point
    - Line
    - Triangle



Today's Mission

- Implementation (2)
  - Set details of each object and draw
    - Point
      - Vertex: (0, 0.5, 0)
      - Color: red(1, 0, 0)
      - Size: 20
    - Line
      - Vertex: (-0.5, 0, 0), (0.5, 0, 0)
      - Color: green (0, 1, 0)
      - Width: 10
    - Triangle
      - Vertex: (0, -0.25, 0), (-0.5, -0.75, 0), (0.5, -0.75, 0)
      - Color: blue (0, 0, 1)