Chapter 14
More About Classes
Reading pp. 811-818

Spring 2017

Instance Variables

- Each class object is an instance of a class
- Each class object has its own class member variables

- What does Rectangle cR1, cR2; look like in memory?

Static Members

- **static data members** and **static member functions** do not belong to ***any*** instance of a class
- An instance of a class does not have to exist to use a static member
- static members belong to the class not an instance of a class

Static Member Example
Tree Interface

```
class GameObject
{
  public:
    GameObject ();


  private:
    static int numberOfGameObjects;
};
```

Static Member Example
Tree Implementation

```
#include "GameObject.h"

int GameObject::numberOfGameObjects = 0;

GameObject::GameObject ()
{
  ++numberOfGameObjects;
}
```

Static Member Variable Specifics

- The static variable assignment must happen outside of the class declaration
- Typically, the initialization happens in the class implementation
- A static integer will be zero unless it is defined otherwise

Static Member Variable Specifics

- The lifetime of a class's static member variables is the lifetime of the program
- Static variables come into existence BEFORE any instances of the class are created

Static Member Functions

- A static member function is of the form:

static returntype functionName (Params);

Static Member Functions

- A static member function CANNOT access any nonstatic member data
- A static member function CAN access static member variables before any class instances are defined in memory
- Modifiers such as const are not allowed on static member functions

Static Member Functions

- A static member function is accessed by using:
  - ClassName::

Static Member Example
Tree Interface

```
class GameObject
{
  public:
    GameObject ();
    static int getNumObjects ();

  private:
    static int numberOfGameObjects;
};
```

Static Member Example
Tree Implementation

```
#include "GameObject.h"

int GameObject::numberOfGameObjects = 0;

GameObject::GameObject ()
{
  ++numberOfGameObjects;
}
int GameObject::getNumObjects ()
{
  return numberOfGameObjects;
}
```

What's the output?

```
#include "GameObject.h"
#include <iostream>

int main ()
{
  GameObject cGO1, cGO2;
  GameObject acGO[5];
  std::cout << "Number Of Objects = "
            <<  GameObject::getNumObjects () << std::endl;

  return EXIT_SUCCESS;
}
```

Problem

- Consider MyMath.h as follows:

```
#ifndef MYMATH_H
#define MYMATH_H


class MyMath
{
  public:
    static const double PI;
    static int gcd (int, int);
};


#endif
```

Problem

- Create MyMath.cpp as follows:

```
#include "MyMath.h"


const double MyMath::PI = 3.14159;


int MyMath::gcd (int num1, int num2)
{
   // write greatest common divisor code
}
```

Problem

- Create MyMathDriver.cpp as follows:

```
#include <iostream>
#include "MyMath.h"

using namespace std;

int main ()
{
  int int1, int2;
  std::cout << "PI = " << MyMath::PI << std::endl;
  std::cout << "Enter Integer #1: ";
  std::cin >> int1;
  std::cout << "Enter Integer #2: ";
  std::cin >> int2;
  // Write the statement to output the gcd of
  // int1 and int2
  return EXIT_SUCCESS;
}
```