CS250 Assignment 3
Using Static

Date assigned: Monday, February 25, 2013
Date due: Monday, March 4, 2013
Total points: 20

Problem

The purpose of this assignment is to a) implement a class using static variables and static functions, b) link object files from one project to another, and c) run a basic DarkGDK program.

Step #1:

a) Create a project called **Random**.

b) Create a class interface called **Random.h** as follows:

```
#ifndef RANDOM_H
#define RANDOM_H

class Random
{
  public:
    static void setSeed (int);
    static int getSeed ();
    static int getNumber ();
    static int getNumber (int, int);

  private:
    static int mSeed;
};

#endif
```

c) Create a class implementation called **Random.cpp** which implements each of the above static functions and assigns mSeed an initial value of 999;

```
#include "Random.h"
#include <cstdlib>

int Random::mSeed = 999;

void Random::setSeed (int seed)
{
  mSeed = seed;
  srand (mSeed);
}

int Random::getSeed ()
{
  return mSeed;
}
```

```
int Random::getNumber ()
{
  return rand ();
}

int Random::getNumber (int lower, int upper)
{
  // you write the code to return a random number between lower and upper inclusive
}
```

d) Create a driver called RandomDriver.cpp as follows:

```
#include "Random.h"
#include <iostream>
#include <ctime>

using namespace std;

int main ()
{
  cout << Random::getNumber () << endl;
  Random::setSeed (static_cast<int> (time (NULL)));
  cout << Random::getNumber () << endl;

  return EXIT_SUCCESS;
}
```

Step #2:

a) Create a project called **Rational**.

b) Create a class interface called Rational.h as follows:

```
#define RATIONAL_H

#include <iostream>

using namespace std;

class Rational
{
  public:
    Rational (int = 0, int = 1);
    void setNumerator (int);
    void setDenominator (int);
    int getNumerator () const;
    int getDenominator () const;
    void print (ostream &)  const;
    bool bIsqual (const Rational &) const;
    Rational multiply (const Rational &) const;
    void reduce ();

  private:
    int mNumerator;
    int mDenominator;
    int gcd (int, int);
};

#endif
```

c) Implement each of the methods. When you multiply two Rational objects, you are to return a reduced Rational object using the method reduce. The method reduce will need to use the private method gcd.

d) Create a driver called RationalDriver.cpp that creates an array of 10 random Rational objects where the numerator is in the range of 0 to 10 and the denominator is in the range of 1 to 10. You are to use getNumber from your Random class to generate random numbers. Your output is to look like the following:

```
********************
  Rational Objects
********************

Rational Object #1:  8/10
Rational Object #2:  5/10
Rational Object #3:  1/6
Rational Object #4:  0/8
Rational Object #5:  1/9
Rational Object #6:  0/3
Rational Object #7:  4/2
Rational Object #8:  0/6
Rational Object #9:  9/10
Rational Object #10:  2/5
Press any key to continue . . .
```

e) You will need to link the code from Random together with Rational to get this assignment to work. We will go over this in class.

Step #3

You are to grab the DarkGDK program that I will place in the CS250 Public folder by Wednesday. Run this program and then capture the window. Print out the window to a color printer and attach this printout to your program printout.

To complete this assignment you must

1. Create two new C++ projects **Random** and **Rational** in Visual Studio 2010 in your existing solution **PUNetIDAssignments.**
2. Type the solution (**fully documented/commented**) to the problem into your project.
3. Remember to enter in your name as the author of the program.
4. Make sure that your program compiles and runs correctly. If you get any errors, double check that you typed everything correctly. Be aware that C++ is case-sensitive. Also, there must not be any warnings when compiling your program or you will lose points.
5. Once you are sure that the program works correctly, it is time to submit your solution. You do this by logging on to Turing and placing your complete solution folder with four working projects (01_DNA, 02_WordSearch, Random, and Rational) in the proper **CS250 Drop** folder. Make sure that you copy your program folder and don't move the folder. If you move the folder, then you will not have your own copy!

Additional Notes

1. You must follow the coding standards found on the main CS250 Web page. Each method must be documented as shown below. The method is either a Constructor or a Method.

```
//*************************************************************************
// Constructor: Rational
//
// Description: Initializes data members to default values
//
// Parameters:  numerator   - the numerator of the rational number
//              denominator - the denominator of the rational number
//
// Returned:    None
//*************************************************************************
```

2. You must use constants when possible.

3. Your program will be graded on **efficiency**. In other words, you will be marked down for repeating code statements unnecessarily.

4. You may only use the C++ programming concepts covered thus far in class. Do not use any more advanced concepts that we have not covered or any other programming concepts that you have had experience with.

5. Your output must look **exactly** like the sample given.

6. Start early!!!!