Langara College

# Department of Computing Science & Information Systems

# CPSC1160 – Algorithms and Data Structures I

###### **Lab3: Objects and Classes**

**Problems: [40 marks]**

**Instructions:**

1. Create a folder named **Lab3** and inside the folder **Lab3** create a folder for each problem.
2. For each problem, separate definition from implementation and create a make file to do compiling and linking.
3. All of your programs must have good internal documentation. You must document every class, every constructor, and every function.

**Problem 1: [10 marks] *The* Rectangle *class*** (Filename: Rectangle.cpp)

Design a class named **Rectangle** to represent a rectangle. The class contains:

1. Two double data fields named width and height that specify the width and height of the rectangle.
2. A no-arg constructor that creates a default rectangle with width 1 and height 1.
3. A constructor that creates a rectangle with the specified width and height.
4. The accessor and mutator functions for all the data fields.
5. A function named getArea() that returns the area of this rectangle.
6. A function named getPerimeter() that returns the perimeter.

Implement the class but don’t separate definition from implementation. Write a test program that creates two Rectangle objects. Use the no-arg constructor to create first object and assign width 4 and height 40 to the second object. Display the properties of both objects and find their areas and perimeters.

**The makefile for this problem should look like the following:**

TestRectangle: TestRectangle.o Rectangle.o

g++ TestRectangle.o Rectangle.o -o TestRectangle

TestRectangle.o: TestRectangle.cpp Rectangle.h

g++ -c TestRectangle.cpp

Rectangle.o: Rectangle.cpp Rectangle.h

g++ -c Rectangle.cpp

**Problem 2: [15 marks] *The* Account *class***

(Filenames: Account.h, Account.cpp, TestAccount.cpp)

Design a class named **Account** that contains:

1. An int data field named id for the account.
2. A double data field named balance for the account.
3. A double data field named annualInterestRate that stores the current interest rate.
4. A no-arg constructor that creates a default account with id 0, balance 0, and annualInterestRate 0.
5. The accessor and mutator functions for id, balance, and annualInterestRate.
6. A function named getMonthlyInterestRate() that returns the monthly interest rate.
7. A function named withdraw(amount) that withdraws a specified amount from the account.
8. A function named deposit(amount) that deposits a specified amount to the account.

Implement the class through separating definition from implementation and write a test program that creates an Account object with an account ID of 1122, a balance of $20000, and an annual interest rate of 4.5%. Use the withdraw function to withdraw $2500, use the deposit function to deposit $3000, and print the balance, and the monthly interest.

**Problem 3: [15 marks] *The* Fraction *class***

(Filenames: Fraction.h, Fraction.cpp, TestFraction.cpp)

Design a class named **Fraction**. This class is used to represent a ratio of two integers. Include mutator functions that allow the user to set the numerator and the denominator. Also include a member function that returns the value of the numerator divided by the denominator as double. Include an additional member function that returns the value of the fraction reduced to lowest terms as string. For example, instead of returning 20/60 the function should return 1/3. This will require finding the greatest common divisor for the numerator and denominator, and then dividing both by that number. Include a private function that returns the greatest common divisor of two integers.

**Hint:** To append an integer to a string, first you need to use a stringstream object defined in <stringstream> to convert the integer to a string, or use the itoa function defined in <cstdlib> to convert the integer to an array of characters.

Implement the class through separating definition from implementation and write a test program that tests several examples of fractions and displays the results on a console.

**When to hand in**

By the end of the lab time, demo **Problem 1** to the instructor.

By 11:59pm, Thursday, September 28, 2017, zip the other problems and submit them to D2L.