**LAB Grade:**

|  |  |
| --- | --- |
| http://www.aus.edu/common/images/logo_small.gif | **American University of Sharjah**  **College of Engineering**  **Department of Computer Science and Engineering** |

**Faculty Details**

|  |  |
| --- | --- |
| Instructor: Lab Instructor :  Office:  Phone:  e-mail:  Semester: | Dr. Michel Pasquier Mr. Suresh Radder  EB2-126A  971-6-515-2924 [sradder@aus.edu](mailto:sradder@aus.edu) Spring 2017 |

**Course Details**

|  |  |
| --- | --- |
| Course:  Semester: | CMP 256 L – GUI Design and Programming  Spring 2017 |

**Lab and Assignment Details**

|  |  |
| --- | --- |
| Assignment No:  Assignment Topic:  Date:  Lab Location: | **3**  Classes, Composition and Aggregation  14th Feb 2017  EB2 -125 |

**Academic Integrity Pledge**

|  |
| --- |
| As a student of American University of Sharjah, I here by state that I will abide by the AUS Integrity Pledge that:   * I will hold myself accountable for all that I say and write. * I will hold myself responsible for the academic integrity of my work * I will not carry out unauthorized copying or printing of the work of others * I will not misrepresent my work nor give or receive unauthorized aid * I will behave in a manner that demonstrates concern for the personal dignity, rights and freedoms of all members of the community * I will respect university property and the property of others; and * I will not tolerate a lack of respect for these values.   **Student Name:**  **Student ID:** |

**CMP256 – GUI Design and Programming**

**Lab 3 - Classes**

***Objectives***

* Defining classes
* Creating *objects (instances)* of classes
* Understanding *member variables* and *member methods*
* Understanding *class variables* and *class methods*
* Understanding *“has-a”(*composition or aggregation) and *“uses-a”* (dependency) relationships between classes

***Exercise 1***

A *complex* number is a number that can be expressed in the form *a* + *bi*, where *a* and *b* are real numbers and *i* is the imaginary unit, which satisfies the equation *i*2 = −1.

*Equality of Complex Numbers:*

*a + bi == c + di* if and only if *a = c* AND *b* = *d*.

*Addition and Subtraction:*

*(a + bi) + (c + di) = (a + c) + (b + d) i*

*(a + bi) - (c + di) = (a - c) + (b - d) i*

Create a class*Complex*that represents a complex number and provide required *constructors* and *accessor and mutator*  methods. Provide the following additional methods also.

* *public void add( Complex cn)*, which adds the parameter to the instance.
* *public void subtract (Complex cn),* which subtracts the parameter from the instance.
* Override *toString()* so that it returns the string in the form *realPart + imaginaryPart i*
* Override *clone()*

Develop a main program to perform the following tasks.

* Creates an array of *complex* numbers. The *size* of the array and, *real* and *imaginary* values for each complex number must be entered by the user.
* Create a new array such that a *complex* number entered by the user is added to each *complex* number of the *original* array and result is stored in the *new* array in the respective location.
* Create a new array such that a complex number entered by the user is subtracted from each complex number of the original array and the result is stored in the *new* array in the respective location.
* Print all *complex* numbers of the *original* array and the *newly* created arrays.(After *addition* and *subtraction*)
  + Print the total number of instances of the *Complex* class created.

***/\****

***\* To change this license header, choose License Headers in Project Properties.***

***\* To change this template file, choose Tools | Templates***

***\* and open the template in the editor.***

***\*/***

***package q1;***

***import java.util.Scanner;***

***/\*\****

***\****

***\* @author g00061542***

***\*/***

***public class Complex {***

***/\*\****

***\* @param cn***

***\*/***

***private int a=1,b=1;***

***public void setA(int A){A=a;}***

***public void setB(int B){B=b;}***

***public int getA(){return a;}***

***public int getB(){return b;}***

***public void add(Complex cn)***

***{***

***this.a=cn.a+this.a;***

***this.b=cn.b+this.b;***

***}***

***public void subtract(Complex cn)***

***{***

***this.a=cn.a-this.a;***

***this.b=cn.b-this.b;***

***}***

***@Override***

***public String toString()***

***{***

***return (Integer.toString(this.a)+"+"+Integer.toString(this.b)+"i");***

***}***

***/\*\****

***\****

***\* @return***

***\* @throws CloneNotSupportedException***

***\*/***

***@Override***

***public Complex clone() throws CloneNotSupportedException***

***{***

***Complex clones=new Complex();***

***clones.a=this.a;***

***clones.b=this.b;***

***return (clones);***

***}***

***public static void main(String[] args) {***

***// TODO code application logic here***

***Scanner in=new Scanner(System.in);***

***System.out.println("Please enter size of the array:");***

***int size=in.nextInt();***

***Complex arr[]=new Complex[size];***

***for(int i=0;i<size;i++)***

***{***

***arr[i]=new Complex();***

***System.out.println("a = ");***

***arr[i].setA(in.nextInt());***

***System.out.println("b = ");***

***arr[i].setB(in.nextInt());***

***}***

***Complex arrAdd[]=new Complex[size];***

***System.out.println("Please enter the values you want to add : ");***

***for(int i=0;i<size;i++)***

***{***

***arrAdd[i]=new Complex();***

***System.out.println("a = ");***

***arrAdd[i].setA(in.nextInt());***

***System.out.println("b = ");***

***arrAdd[i].setB(in.nextInt());***

***arrAdd[i].add(arr[i]);***

***}***

***Complex arrSub[]=new Complex[size];***

***System.out.println("Please enter the values you want to Subtract : ");***

***for(int i=0;i<size;i++)***

***{***

***arrSub[i]=new Complex();***

***System.out.println("a = ");***

***arrSub[i].setA(in.nextInt());***

***System.out.println("b = ");***

***arrSub[i].setB(in.nextInt());***

***arrSub[i].subtract(arr[i]);***

***}***

***System.out.println("Complex numbers : ");***

***for(int i=0;i<size;i++)***

***{***

***System.out.println(arr[i]);***

***}***

***System.out.println("Addition :");***

***for(int i=0;i<size;i++)***

***{***

***System.out.println(arrAdd[i]);***

***}***

***System.out.println("Subtraction : ");***

***for(int i=0;i<size;i++)***

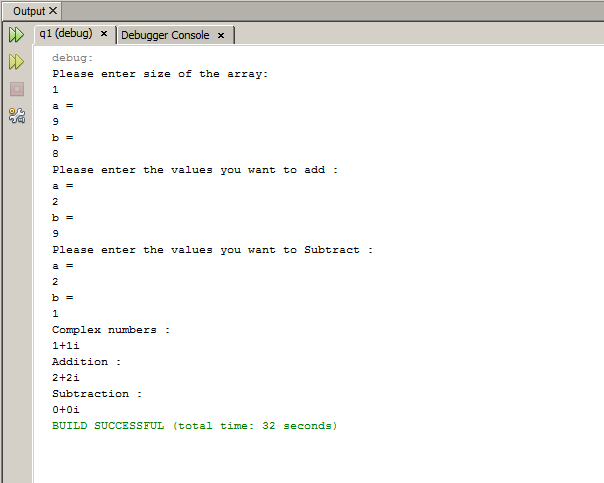
***{***

***System.out.println(arrSub[i]);***

***}***

***}***

***}***



***Exercise 2***

Create a class by name *ComplexArray* that contains an array of instances (objects) of the class *Complex.* The *ComplexArray* class must be provided with suitable constructors to set the size of the array and initialize each array element with a *complex* number that has real and imaginary value generated randomly between any values from 0 – 9 (both inclusive).

Support the *ComplexArray* class with the various methods listed below.

* + *public Complex(int array\_size)//* Constructor that creates an array of Complex numbers of size *array\_size.* Each complex number in the array has *real* and *imaginary* value that is randomly generated.
  + *public* *Complex* *getComplex(int loc) //*returns a complex number at location *loc.*
  + *public* *void setComplex(int loc, Complex cn ) //*sets the complex number to *cn* at location *loc.*
  + Override *toString()* to return the string that has all the complex numbers present in the array in the form *[ a + bi, c + di, e + fi,   . . .   , y + zi ]*
  + *publ*ic *void add (ComplexArray ca),* which adds each complex number present in the array contained in argument (ca) to each complex number of the array contained in the instance that invokes this method. If the size of the array contained in the argument is not equal to the size of the array contained in the instance that invokes this method then an exception must be thrown without performing addition.
  + *public* *void subtract (ComplexArray ca),* which subtracts each complex number present in the array contained in the argument *(ca)* from each complex number of the array contained in the instance that invokes this method. If the size of the array contained in the argument is not equal to the size of the array contained in the instance that invokes this method then an exception must be thrown without performing subtraction.
  + Override *clone()*
  + Develop a *Test* class that has the *main* program to test various methods mentioned above.
  + State the relationship involved between the classes  *ComplexArray* and *Complex.*

***Exercise 3***

A teacher creates a *quiz* which consists of a number of questions for performing *addition, subtraction and multiplication* on *complex* numbers. Every quiz question must be one of the following questions, where *a* and *b* are *real* and *imaginary* part of the complex number generated randomly.

1. *Add* the complex number *a + bi* to each element of the *complex* array
2. *Subtract* the complex number *a + bi from* each element of the *complex* array
3. *Multiply* each element of the complex array by the *complex* number *a+ bi*

Use the *Complex* class developed in *Exercise 1* and the *ComplexArray* class developed in *Exercise 2* for creating the quiz. Develop additional classes such as *QuizQuestion* ( it must be possible to *add* question or *remove* a question), and *Quiz* for the test itself, that uses both *ComplexArray* and *QuizQuestion* classes, and has required methods to set the quiz and print quiz date. Test the classes developed by creating the *quiz* in the following format:

***Note:*** The Quiz class must use (*has-a relationship*) *LocalDate* and *DateTimeFormatter* Classes. The quiz date must be the current date on which the quiz is generated.

Quiz date: 14.03.2017

Consider the following complex array:

5+9i

3+7i

2+1i

4+4i

1+3i

Question:

Add the complex number 5 + 6i to all the elements of the complex array

Answer:

Make sure to draw a UML diagram that describes the relationships between *Quiz, QuizQuestion, ComplexArray and Complex* classes. You can use *Rational Rose Enterprise Edition* (@Computer Science and Engineering) or any open source application or online tool, such as <https://drive.draw.io/>.

*Hand in:* Solution code and sample input/output

*Due Date:* At the beginning of the next lab.

*Grading policy*: Mentioned in the course outline posted on ilearn.