COE 212 – Engineering Programming

Welcome to Exam I
Friday March 27, 2015

Instructors: Dr. George Sakr
Dr. Joe Tekli
Dr. Wissam F. Fawaz

Name: _____

Student ID: _____

Instructions:

1. This exam is **Closed Book**. Please do not forget to write your name and ID on the first page.
2. You have exactly 110 **minutes** to complete the 5 required problems.
3. Read each problem carefully. If something appears ambiguous, please write your assumptions.
4. Do not get bogged-down on any one problem, you will have to work fast to complete this exam.
5. Put your answers in the space provided only. No other spaces will be graded or even looked at.

Good Luck!!

Problem 1: Multiple choice questions (**20 minutes)** [14 points]

For the questions given below, consider the following helper and driver classes:

```
public class Point2D{
      private int x, y;
      public Point2D(int x, int y) {this.x=x; this.y=y;}
      public void setX(int x){this.x=x;}
      public double getX(){return x;}
      public double getY(){return y;}
      public String toString() {
            String output = "["+getX()+", "+getY()+ "]";
            return output;}}
public class MultiPoints{
      public static void main(String[] args) {
            Point2D p1 = new Point2D(10, 20);
            Point2D p2 = new Point2D(30, 40);
            System.out.println(p1 + " " + p2);}}
```

1) What output does the driver class produce when executed?
   a. `p1 p2`
   b. `Point Point`
   c. `(10, 20) (30, 40)`
   d. None of the above
2) Which of the following can be used to compute the distance between p1 and p2? Note that the distance between two points p1(x1, y1) and p2(x2, y2) can be obtained using the following formula: $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
   a. `Math.sqrt((p1.getX()-p2.getX())(p1.getX()-p2.getX())+(p1.getY()-p2.getY())(p1.getY()-p2.getY()))`
   b. `Math.sqrt(Math.pow(p1.x-p2.x,2)+Math.pow(p1.y-p2.y,2))`
   c. `Math.sqrt((Math.pow(p1.getX()-p2.getX(),2)+Math.pow(p1.getY()-p2.getY(),2))`
   d. None of the above
3) Which of the following statements is true about the `Point2D` class?
   a. `Point2D` contains 2 mutator methods
   b. `Point2D` contains 2 accessor methods
   c. `Point2D` violates the encapsulation principle
   d. None of the above is true about `Point2D`
4) How many local variables are used inside the `Point2D` class?
   a. 5
   b. 6
   c. 7
   d. None of the above
5) How many formal parameters are used inside the `MultiPoints` class?
   a. 2
   b. 3
   c. 4
   d. None of the above
6) How many actual parameters are used inside the `Point2D` class?
   a. 0
   b. 1
   c. 4
   d. None of the above

7) What output does the following method call produce: `p1.toString().length()` ?
   a. 6
   b. 7
   c. 8
   d. None of the above
8) What output does the following code fragment produce?
```
String str = p2.toString();
System.out.println(str.substring(str.indexOf('0')+1));
```
   a.]
   b. , 20]
   c. The code fragment results in a syntax error
   d. None of the above
9) Which of the following statements produces a compile-time error?
   a. `int x1 = p1.getX();`
   b. `long x1 = p1.getX();`
   c. Both of the above
   d. None of the above
10) What output does the following code fragment produce? Assume that the `DecimalFormat` class was imported properly.
```
DecimalFormat fmt = new DecimalFormat("0.##");
p1.setX(10.346);
double roundedX1 = fmt.format(p1.getX());
System.out.println(roundedX1);
```
   a. 10.35
   b. 10.34
   c. The code fragment results in a run-time error
   d. None of the above
11) What output does the following code fragment produce?
```
p1 = p2;
System.out.println(p1.toString().charAt(1));
```
   a. 1
   b. 2
   c. 3
   d. None of the above
12) Consider the following statement: `String pt = p1.toString();` Which of the following returns the index of the second '0' in `pt`?
   a. `pt.substring(pt.indexOf('0')).indexOf('0');`
   b. `pt.substring(pt.indexOf('0')+1).indexOf('0');`
   c. `pt.substring(pt.indexOf('0')+1,pt.length()).indexOf('0');`
   d. Both (b) and (c)
13) The compilation of which of the below statements would result in an error?
   a. `System.out.println(p1.getX());`
   b. `System.out.println(p1.setX(4));`
   c. `System.out.println(p1.getX().toString());`
   d. Both (b) and (c)
14) Which of the following can be used to convert the `x` coordinate of `p1` into a `String`?
   a. `Integer.toString(p1.getX())`
   b. `p1.getX() + ""`
   c. All of the above
   d. None of the above

Problem 2: True or false questions (**10 minutes)** [12 points]

1. The following method call returns the cosine of a 90 degrees angle: `Math.cos(90)`

Answer: True **False**

2. `(int) Math.random()*2+1` can be used to generate a random `int` value between 1 (inclusive) and 2 (inclusive).

Answer: True **False**

3. Consider a `String` variable called `str`. If `str` is declared and instantiated properly, then the method call: `str.toUpperCase()` will modify the value of `str` converting all of its characters into upper-case.

Answer: True **False**

4. Given that a forward slash (/) is used to mark the beginning of an escape sequence, it is not possible to print the forward slash character (/) using a `System.out.print` statement.

Answer: True **False**

5. Consider 2 String objects called `str1` and `str2` respectively. The statements given below, and that we labeled as **Statement1** and **Statement2**, are functionally equivalent.
   Statement1: `String str = str1 + " " + str2;`
   Statement2: `String str = str1.concat(str2);`

Answer: True **False**

6. The following code fragment creates two variables that are aliases of each other.
   `int val1 = 1, val2 = 2; val1 = val2;`

Answer: True **False**

7. All the methods of the `Math` class are static and return a `double` output value.

Answer: True **False**

8. When a method is called, each formal parameter is copied and stored in the corresponding actual parameter.

Answer: True **False**

9. The header of a method defines the code that will be executed when that method is called.

Answer: True **False**

10. If the calling method and the called method belong to two different classes, then the called method should be invoked through an object created from the class that contains it.

Answer: **True** False

11. If `letter` is a `char` variable, then `letter.toUpperCase()` returns a lower case version of the character stored in `letter`.

Answer: True **False**

12. Java is a strongly typed language.

Answer: **True** False

Problem 3: Code analysis (**15 minutes**) [10 points]

1) Consider the two helper classes given below, along with a driver class.

```
public class ClassA {
      private int x;
      public ClassA() { x = 4; }
      public void scaleValue(int x){
            int temp = x;
            this.x*=temp;}
      public int getX(){return x;}
}
public class ClassBDriver {
public static void main(String[] args) {
        ClassB b=new ClassB();
        int x = 2;
        b.scaleValues(2);
        System.out.println(
        "Answer is: "+
         b.getObj1X()+b.getObj2X());}}
```

```
public class ClassB {
      private ClassA obj1, obj2 ;
      public ClassB(){
            obj1 = new ClassA() ;
            obj2 = new ClassA() ;
      }
      public void scaleValues(int x){
            obj1.scaleValue(x) ;
            obj2.scaleValue(x) ;
      }
      public int getObj1X() {
            return obj1.getX() ;
      }
      public int getObj2X() {
            return obj2.getX() ;
      }
}
```

When running the `ClassBDriver` class, what output is produced?

a. `Answer is: 4`
b. `Answer is: 8`
c. `Answer is: 22`
d. It doesn't compile correctly
e. None of the above

2) Consider the class given below, along with a driver class for it.

```
public class ClassC {
      private int x, y;
      public ClassC() {
            x = 5; y = 1;}
      public int first(int x){
            y=x+1;
            return y+1;}
      public void second(int x,int y) {
            y = first(x);
            this.y=y+1;}
      public void doIt() {
            int val = first(y) ;
            second(y, val) ;
            System.out.print(x+y);}
}
```

```
public class ClassCDriver {
public static void main(String[] args){

        ClassC c=new ClassC();
        c.doIt();
      }
}
```

When running `ClassCDriver` class, what output is produced?

a. `4`
b. `6`
c. `7`
d. It doesn't compile correctly
e. None of the above

Problem 4: Evaluating Java Expressions (**15 minutes**) [14 points]

For each of the following code fragments, what is the value of **x** after the statements are executed?

(1) `DecimalFormat fmt = new DecimalFormat("00.###");` (2 pts)

```
double a = 1.2345;
double b = 100;
b = a/b;
String x = fmt.format(b);
x += b;
```

Answer: x= "00.0120.012345"

(2)
```
int u = 10;
double v = Math.ceil(u*10.28);
int x = ((int)Math.floor(v) - u/10);
```

Answer: x= 102

(3)
```
String x = "A mind is like a parachute: it doesn't work if
it's not open";
x = x.replace('k','m').substring(0).replace(x.charAt(12),
x.charAt(2));
```

Answer: x= "A mind is lime a parachute: it doesn't worm if it's not open"

(4) `Random rnd = new Random();` (2 pts)

```
double y = rnd.nextInt(20)/3;
int x = (int)(rnd.nextFloat() / y) % 5;
```

Answer: x= 0

(5)
```
DecimalFormat fmt = new DecimalFormat("000.#");
String x = fmt.format(1.23).substring(0, 3);
x += x.charAt(x.length()-1);
```

Answer: x= "0011"

(6)
```
int a=13, b=29;
String str = "Good Morning Ladies and Gents";
String x = "\""+str.substring(a,b)+"\"";
```

Answer: x= "Ladies and Gents"

(7)
```
String str = "Encyclopedia Galactica"; (2 pts)
char x = str.charAt(str.length() -
str.substring(10).length());
```

Answer: x= 'i'

(8)
```
double val1 = 123.456;
double val2 = Math.floor(val1*10);
double x = val2 - (int)val1*10;
```

Answer: x= 4.0

(9)
```
int y = 10, z =10;
 String x = y + 1 + z + 1 + "" + z + y;
```

Answer: x= "221010"

(10)
```
int a = 8, b =8, c = 9; (2 pts)
a = b++;
b = ++c;
int x = a + b + c ;
```

Answer: x= 28

Problem 5: Coding (**50 minutes**) [50 points]

1. Write a Java program called `LastNameFirst` that reads from the user a `String` that contains his first name followed by his last name with a white space character in between. Your program should then create and then print out a String consisting of the last name followed by the first name separated by a white space character as illustrated in the sample output given below.

Sample run:
Enter a String: Georges Sakr
Output String: Sakr Georges

```
import java.util.Scanner;

class LastNameFirst{

public static void main (String [] args){

        Scanner scan = new Scanner(System.in);

        System.out.print("Enter a String: ");
        String S = scan.nextLine();

        String S2 = S.substring(S.indexOf(' ')+1) + " " + S.substring(0, S.indexOf(' '));

        System.out.println("Output String: " + S2);
    }
}
```

2. Write a Java program called `InsertString` that reads from the user 2 `String` values denoted by `str1` and `str2` and an integer value denoted by `k`. Your program should then create a `String` str3 that results from the insertion of `str2` at the k^{th} position in `str1`.

Sample run:
Please enter str1: is fun
Please enter str2: Exam
Please enter k: 0
Resulting String str3: Examis fun

```
import java.util.Scanner;

class InsertingString{

public static void main (String [] args){

        Scanner scan = new Scanner(System.in);

        System.out.print("Please enter str1: ");
        String str1 = scan.nextLine();

        System.out.print("Please enter str2: ");
        String str2 = scan.nextLine();

        System.out.print("Please enter k: ");
        int k = scan.nextInt();

        String str3 =  str1.substring(0, k) + str2 + str1.substring(k);

        System.out.println("Output String: " + str3);
    }
}
```

3. We know that the largest power of 2 that is less than some integer N is equal to 2^M where M is given by:

$$\boldsymbol{M = floor\left(\frac{\ln N}{\ln 2}\right)}$$

For example, the largest power of 2 that is less than 600 is 512, since

$$\boldsymbol{M = floor\left(\frac{\ln 600}{\ln 2}\right) = 9 \; and \; 2^9 = 512}$$

Write a Java program that reads from the user an integer value denoted by `N`, Your program should then do the following:

- Find and print the largest power of 2 that is less than the input value `N`
- Generate a random power of 2 that is less than `N`

Sample run:

Please enter an integer: 600
The largest power of 2 less than 600 is: 512
A random power of 2 less than 600 is: 16

```
import java.util.Scanner;
import java.util.Random;

class LargestPowerOf2{

public static void main (String [] args){

        Scanner scan = new Scanner(System.in);

        System.out.print("Please enter an integer: ");
        int N = scan.nextInt();

        double M = Math.floor(Math.log(N)/Math.log(2));
        int lp2 = (int)Math.pow(2, M);        // Largest power of 2 less than N

        System.out.println("The largest power of 2 less than " + N + " is: " + lp2);

        Random rnd = new Random();
        int m = rnd.nextInt((int)M + 1);
        int rand_p2 = (int)Math.pow(2, m);    // Random power of 2 less than N

        System.out.print("A random power of 2 less than " + N + " is: " + rand_p2);
    }
}
```

4. Write a program that simulates the job of a cashier by:
    - Reading a price *p* (in dollars, expressed as an integer value) corresponding to the purchases of a customer.
    - Reading from the costumer the amount of money *m* that he/she will pay (in Lebanese Pounds – LBP, also expressed as an integer value). We consider that the amount of money paid is always greater that the price.
    - And then computing the amount of money that the cashier program should return to the customer by displaying: the number ***Nb_100*** of bills of "100 thousand LBP, the number ***Nb_50*** of bills of 50 thousand LBP, the number ***Nb_10*** of bills of 10 thousand LBP, the number ***Nb_5*** of bills of 5 thousand LBP, the number ***Nb_1*** of bills of 1 thousand LBP, and the number ***Nb*** of bills of 500 LBP to be returned to the customer.

We consider that $1 = 1500 LBP

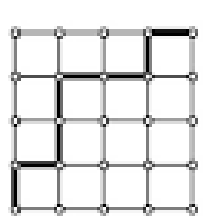
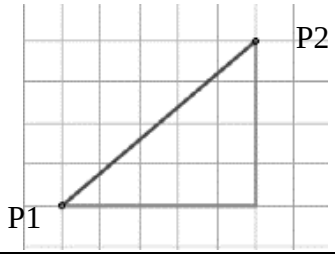
Sample run:

Enter price in Dollars: 230
Enter the amount of money that the customer pays, in Lebanese Pounds: 500000
The cashier returns:
1 bill(s) of 100,000 L.P.
1 bill(s) of 50,000 L.P.
0 bill(s) of 10,000 L.P.
1 bill(s) of 5,000 L.P.
0 bill(s) of 1,000 L.P.
0 bill(s) of 500 L.P.

```
import java.util.*;

public class Cashier
{
   public   static   void   main (String[] args)
         {

            final int dollar = 1500;
            int p;   // price (in $)
            int m;   // amount of money (in LP)
            int Nb_100;
            int Nb_50;
            int Nb_10;
            int Nb_5;
            int Nb_1;
            int Nb;

          Scanner scan = new Scanner(System.in);

          System.out.print("Enter price in Dollars: ");
```

```
        p = scan.nextInt();

        System.out.print("Enter the amount of money that the customer pays," +
                         "in Lebanese Pounds: ");
        m = scan.nextInt();

        m = (m - p*1500);          // Computing the remainder,
                                   // that the cashier should return to the customer

        Nb_100 = m/100000;         // Computing the number of 100 thousand L.P. bills
                m = m%100000;

        Nb_50 = m/50000;           // Computing the number of 50 thousand L.P. bills
                m = m%50000;

        Nb_10 = m/10000;           // Computing the number of 10 thousand L.P. bills
                m = m% 10000;

        Nb_5 = m/5000;             // Computing the number of 5 thousand L.P. bills
                m = m % 5000;

        Nb_1 = m/1000;             // Computing the number of 1 thousand L.P. bills
                m = m % 1000;

        Nb = m/500;                // The number of 500 L.P. bills

        System.out.println("The cashier returns: ");
        System.out.println(Nb_100 + " bills of 100 000 L.P.");
        System.out.println(Nb_50 + " bills of 50 000 L.P.");
        System.out.println(Nb_10 + " bills of 10 000 L.P.");
        System.out.println(Nb_5 + " bills of 5 000 L.P.");
        System.out.println(Nb_1 + " bills of 1 000 L.P.");
        System.out.println(Nb + " bills of 500 L.P.");
    }
}
```

5. **Taxicab geometry** is a form of geometry in which the distance between two points in a 2-D space is evaluated as the sum of the absolute differences of their Cartesian coordinates. It was inspired by the block layout of Manhattan in New York City and the distances taxicabs need to traverse in order to go from one point to another in the city. The distance function following taxicab geometry is called the **Manhattan distance**, in contrast to the conventional distance function based on **Euclidian geometry**. Note that both functions (namely, **Manhattan** and **Euclidean** distance functions) are given in the figure below.

$d_{Manhattan}(P_1, P_2) = \lvert x_1 - x_2 \rvert + \lvert y_1 - y_2 \rvert$	$d_{Eucledian}(P_1, P_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
	

Write a program which reads from the user the (x, y) coordinates of two points P1 and P2 in 2-D space, and then computes both Manhattan and Euclidian distance functions following the formulas above.

Sample run:

Enter x coordinate of point P1: 0
Enter y coordinate of point P1: 0
Enter x coordinate of point P2: 1
Enter y coordinate of point P2: 1
Manhattan distance (P1, P2) = 2.000
Euclidian distance (P1, P2) = 1.414

```
import java.util.Scanner;
import java.text.DecimalFormat;

public class TaxiCab
{
   public   static   void   main (String[] args)
          {

                  double x1, y1, x2, y2, Man, Euc;

                   Scanner scan = new Scanner(System.in);

                   System.out.print("Enter x coordinate of Point P1: ");
```

```
                x1 = scan.nextDouble();

                System.out.print("Enter y cooridnate of Point P1: ");
                y1 = scan.nextDouble();

                System.out.print("Enter x coordinate of Point P2: ");
                x2 = scan.nextDouble();

                System.out.print("Enter y cooridnate of Point P2: ");
                y2 = scan.nextDouble();

                Man = Math.abs(x1 - x2) + Math.abs(y1 - y2);
                Euc = Math.sqrt(Math.pow(x1 - x2, 2) + Math.pow(y1 - y2, 2));

                DecimalFormat fmt = new DecimalFormat("0.000");

                System.out.println("Manhattan distance(P1, P2) = " + fmt.format(Man));
                System.out.println("Euclidian distance(P1, P2) = " + fmt.format(Euc));

    }
}
```