COE 212 – Engineering Programming

Welcome to Exam I
Thursday June 21, 2018

Instructor: 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 **minutes** to complete the 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)** [20 points]

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

```
public class ProblemI {
public static void main(String[] args) {
Team team1 = new Team("Brazil", 3, 2);
team1.play();
System.out.println(
team1.getName().substring(3));
System.out.println(team1.getWins());
Team team2 = team1;
team2.play();
System.out.println(team2.getLosses());
System.out.println(team2.getMatches());
Team team3 = new Team("Germany", 4, 1);
int index = team3.toString().indexOf(",");
System.out.println(index);
System.out.println(
team3.toString().substring(index)+1);
team3.setWins(team1.getWins());
System.out.println(team3.getWins());}
}
```

```
public class Team {
private String name;
private int wins, losses, matches;
public Team(String n, int w, int L) {
name=n; wins=w; losses=L; matches=0;}
public void play() { matches++; wins++; }
public String getName() {
      return name; }
public int getWins() {
      return wins; }
public void setWins(int w) {
       wins = w; }
public int getLosses() {
      return losses; }
public int getMatches() {
      return matches; }
public String toString() {
return name + ":" + wins + "," + losses;
}}
```

1) How many accessor methods does the helper class include?
   a. 1
   b. 3
   c. **4**
   d. None of the above
2) How many mutator methods does the helper class include?
   a. **1**
   b. 3
   c. 4
   d. None of the above
3) What output does the first `println` statement of the driver class produce?
   a. `razil`
   b. `azil`
   c. **`zil`**
   d. None of the above
4) What output does the second `println` statement of the driver class produce?
   a. 2
   b. 3
   c. **4**
   d. None of the above
5) What output does the third `println` statement of the driver class produce?
   a. **2**
   b. 3
   c. 4
   d. None of the above
6) What output does the fourth `println` statement of the driver class produce??
   a. **2**
   b. 3
   c. 4

d. None of the above

7) What output does the fifth `println` statement of the driver class produce?

a. `6`

b. `7`

c. `8`

d. **None of the above**

8) What output does the sixth `println` statement of the driver class produce?

a. `1`

b. `,1`

c. **`,11`**

d. None of the above

9) What output does the seventh `println` statement of the driver class produce?

a. `3`

b. `4`

c. **`5`**

d. None of the above

10) Assume that immediately after the creation of the `team3` object inside the driver class, an object called `team4` is created as follows:

```
Team team4 = new Team("Germany", 4, 1);
```

Which of the following pairs of variables of the driver class are aliases of each other?

a. **`team1` and `team2`**

b. `team3` and `team4`

c. Both of the above

d. None of the above

11) Which of the following statements correctly extracts the last character of the name associated with the `team3` object and stores it in the `char` variable called `last`?

a. `char last = team3.getName().charAt(team3.length());`

b. `String last = team3.getName().substring(team3.length());`

c. Both of the above

d. **None of the above**

12) Which of the following statements doubles the number of wins stored in `team2`?

a. `team2.getWins()*2;`

b. **`team2.setWins(team2.getWins()*2);`**

c. Both of the above

d. None of the above

13) Which of the following statements prints the length of the name stored in `team1`?

a. `System.out.println(team1.length());`

b. `System.out.println(team1.toString().length());`

c. Both of the above

d. **None of the above**

14) Which of the following statements computes the square root of the number of losses stored in `team3` and correctly assigns the resulting value to the variable called `output`?

a. `int output = Math.sqrt(team3.getLosses());`

b. `float output = Math.sqrt(team3.getLosses());`

c. Both of the above

d. **None of the above**

15) Which of the following correctly divides the value of π by the number of matches associated with `team3` and then stores the resulting value in a variable called `root`?

a. `Double root = Math.PI/team3.getMatches();`

b. `double root = Math.PI()team3.getMatches();`

c. `double root = Math.PI/team3.getMatches();`

d. **Both (a) and (c)**

<u>**Problem 2:**</u> True or false questions (**10 minutes)** [20 points]

1. The following code fragment prints: `111010`

```
int x=9, y, z;
y = ++x;
z = y++;
System.out.print(“”+y+x+z);
```

Answer: **True** False

2. The following code fragment prints: `8`

```
double x = Math.sqrt(Math.abs(-4));
x *= Math.pow(x, 2) ;
System.out.print(x);
```

Answer: True **False**

3. The following code fragment outputs: `.46`

```
DecimalFormat fmt = new DecimalFormat(“0.##”);
String value = fmt.format(23.457);
System.out.print(value.substring(value.length()-3));
```

Answer: **True** False

4. The following statement prints a random `double` between 1 and 6:

```
System.out.print((int) (Math.random()*6+1));
```

Answer: True **False**

5. The `Random` class is part of the `java.util` package and offers a constructor that does not accept any parameter.

Answer: **True** False

6. The following code fragment prints: `12`

```
String str = “COE 212”;
int index = str.indexOf(‘2’);
str = str.substring(index+1);
System.out.print(str.charAt(0) + str.charAt(1));
```

Answer: True **False**

7. A getter method provides read only access to one of the instance variables of a class.

Answer: **True** False

8. The following code fragment decrements the variable called `val` by `3`.

```
val--;  --val; val -= 1;
```

Answer: **True** False

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

Answer: True **False**

10. The following statement outputs: `2.5`

```
System.out.print((double) (5/2));
```

Answer: True **False**

<u>Problem 3:</u> Evaluating Java expressions (**15 minutes**) [20 points]

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

(1)
```
int a = 0, b = 4 ;
String S = “Good Morning Byblos”;
String x = S.substring(a+1, b-1);
```
Answer: x=“oo”

(2)
```
DecimalFormat fmt = new DecimalFormat(“000.##”);
double z = 20/3;
z += z/1000;
String x = fmt.format(z);
```
Answer: x= “006.01”

(3)
```
String S = “The exam is fun”;
S = S.replace(‘e’, ‘x’);
String x = S.replace(‘X’, ‘E’);
```
Answer: x=“Thx xxam is fun”

(4)
```
double a = Math.pow(3, 2);
double b = Math.pow(3, 3);
double x = Math.pow(a + b, 1.0/2);
```
Answer: x=6.0

(5)
```
String x = “x”;
String y = “z”;
x = x.concat(x).concat(“\\”).concat(y);
```
Answer: x=“xx\z”

(6)
```
double val1 = 123.456;
double val2 = Math.ceil(val1);
double x = val2 – (int) val1 ;
```
Answer: x=1.0

(7)
```
String str = “World Cup”;
char x = str.charAt(str.length() –
str.substring(4).length());
```
Answer: x=‘d’

(8)
```
int y = 5, z = 6;
String x = z - y + “” + y + z;
```
Answer: x=“156”

(9)
```
String S1 = “Java is Fun”;
String S2 = S1.toUpperCase().toLowerCase();
boolean x = S1.equalsIgnoreCase(S2);
```
Answer: x=true

(10)
```
String nbs = “567.8”;
 double z =
 Double.parseDouble(nbs.substring(2,nbs.length()));
 int y = Integer.parseInt(nbs.charAt(0)+“”);
 double x = z – y ;
```
Answer: x=2.8

(11)
```
double angle = Math.PI/2;
double x = Math.cos(angle);
```
Answer: x=0.0

(12)
```
String str = “123456”;
 boolean x = str.equals(1+“2”+3+4+“56” );
```
Answer: x=true

(13)
```
String S = new String(“////”);
 String x = S.length() + 2 + 2 + “”;
```
Answer: x= “8”

<u>Problem 4:</u> Coding Problems (**40 minutes**) [40 points]

1. Write a Java application that generates a random integer radius (r) and height (h) for a cylinder in the range of 4 (inclusive) to 10 (inclusive). Your program should then compute and print out to the screen the volume and surface area of the cylinder rounded to 2 decimal places. Note that the volume and area can be calculated as follows:
    - $\boldsymbol{Volume = \pi r^2 h}$
    - $\boldsymbol{Area = 2\pi r h}$

```
import java.util.Random;
import java.text.DecimalFormat;
public class Sample {
     public static void main(String[] args) {
          Random rnd = new Random();
          DecimalFormat fmt = new DecimalFormat("0.##");
          int r, h;
          double volume, area;

          r = rnd.nextInt(7) + 4;
          h = rnd.nextInt(7) + 4;

          volume = Math.PI * r * r * h;
          area = 2 * Math.PI * r * h;

          System.out.println("Volume: " + fmt.format(volume));
          System.out.println("Area: " + fmt.format(area));
     }
}
```

2. The equation of a circle is given by: $x^2+y^2+ax+by+c=0$
   One can determine the coordinates of the center O of the circle as follows:
   $$\boldsymbol{X_O = \frac{-a}{2} \; and \; Y_O = \frac{-b}{2}}$$
   And the radius of the circle is given by:
   $$R = \sqrt{\frac{a^2+b^2}{4} - c}$$
   Write a Java program called `CircleStats` that reads from the user 3 `int` values representing the coefficients a, b, and c present in the circle's equation given above. Your program should then print out to the screen the coordinates of the center of the circle, its radius, its area given by πR^2 , and its perimeter given by $2\pi R$. Note that the area and perimeter must be displayed rounded to 3 decimal places.

```
import java.util.Scanner;
import java.text.DecimalFormat;
public class Sample {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int a,b,c;
        double x, y, radius, area, perim;
        DecimalFormat fmt = new DecimalFormat("0.###");

        System.out.println("Enter a: "); a = scan.nextInt();
        System.out.println("Enter b: "); b = scan.nextInt();
        System.out.println("Enter c: "); c = scan.nextInt();
        x = -a/2.0; y = -b/2.0;
        radius = Math.sqrt((a*a+b*b)/2.0 - c);
        area = Math.PI * radius * radius;
        perim = 2 * Math.PI * radius;

        System.out.println("X: " + x);
        System.out.println("Y: " + y);
        System.out.println("Radius: " + radius);
        System.out.println("Area: " + fmt.format(area));
        System.out.println("Perimeter: " + fmt.format(perim));


    }
}
```

3. A complete phone number is made up of a country code, an area code and the corresponding actual number separated by minus signs. For example, in the following phone number 001-617-9874325, 001 is the country code, 617 is the area code and 9874325 is the corresponding actual number. Write a program called `NumberSplitter` that reads from the user a complete phone number as a string and prints out its associated country code, area code and corresponding actual number.

Sample run 1:
Enter Phone Number: 001-619-6587649
Country code: 001
Area code: 619
Corresponding Number: 6587649

Sample run 2:
Enter Phone Number: 00961-05-123456
Country code: 00961
Area code: 05
Corresponding Number: 123456

```
import java.util.Scanner;
public class Sample {
     public static void main(String[] args) {
          Scanner scan = new Scanner(System.in);
          String phone, country, area, actual;
          int index;
          System.out.println("Enter phone number: ");
          phone = scan.nextLine();

          index = phone.indexOf("-");
          country = phone.substring(0, index);
          phone = phone.substring(index+1);
          index = phone.indexOf("-");
          area = phone.substring(0, index);
          actual = phone.substring(index+1);

          System.out.println("Country code: " + country);
          System.out.println("Area code: " + area);
          System.out.println("Actual: " + actual);
     }
}
```

4. Write a Java application called `NumberDissector` that reads from the user an integer and breaks down into a sequence of individual digits. For example, if the input value is 16384, then the output produced by your code should be: 1 6 3 8 4
   Assume that the user does not enter more than five digits and that the input value cannot be negative. That is, it is safe to assume that the maximum number of digits typed by the user is equal to 5, although the user may type fewer digits as illustrated in the example below. If the user ever decides to enter fewer digits as illustrated below, then the output must be stuffed with 0s at the beginning to make the total number of individual digits printed out equal to 5.

Sample run:

Enter an integer composed of at most 5 digits: 123
Output: 0 0 1 2 3

```
import java.util.Scanner;
import java.text.DecimalFormat;
public class Sample {
     public static void main(String[] args) {
Scanner scan = new Scanner(System.in);
DecimalFormat fmt = new DecimalFormat("00000");
int input; String inputStr;
System.out.println("Enter an int: ");
input = scan.nextInt();
inputStr = fmt.format(input);
System.out.print("Output: " + inputStr.charAt(0) + " " +
inputStr.charAt(1) + " " + inputStr.charAt(2) + " " +
inputStr.charAt(3) + " " + inputStr.charAt(4));

}
}
```