COE 212 – Engineering Programming

Welcome to Exam I
Friday March 11, 2016

Instructors: Dr. Salim Haddad
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 **115 minutes** to complete the **five** 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 (**25 minutes)** [14 points]

For the questions given below, consider the following Java statements:

```
double valD = 23.456;
String str;
Float valF;
```

1) Which of the following statements correctly prints the fractional portion of `valD` out?
   a. `System.out.println(valD – Math.ceil(valD));`
   b. `System.out.println(valD – (Int) valD);`
   c. Both of the above
   d. None of the above
2) Which of the following exactly stores the value of `valD` in `str`?
   a. `DecimalFormat fmt = new DecimalFormat("000.###");`
      `str = fmt.format(valD);`
   b. `String dummy = " ";`
      `str = dummy + valD;`
   c. `str = (String) valD;`
   d. None of the above
3) Assume that `str` correctly receives the value of `valD`. Which of the following can be used to extract the integer portion of `valD` and then multiply it by a factor of `10`?
   a. `Integer.parseInt(str.substring(0, 2) * 10)`
   b. `Integer.parseInt(str.substring(0, str.indexOf('.'))*10)`
   c. Both of the above
   d. None of the above
4) Assuming that `str` correctly receives the value of `valD`, which of the following can be used to extract the digits that make up the fractional part of `valD` out?
   a. `str.charAt(3) + str.charAt(4) + str.charAt(5) + "";`
   b. `str.substring(str.indexOf('.')+1);`
   c. Both of the above
   d. None of the above
5) Which of the following correctly uses `str` as a formal parameter when calling the method `foo` having a header of: `void foo(double val)`?
   a. `foo(str)`
   b. `foo(Double.parseDouble(str))`
   c. `foo(Double.toString(str))`
   d. None of the above
6) Which of the following correctly stores in `valF` the value of `valD`?
   a. `valF = new Float(valD);`
   b. `valF = (float) valD;`
   c. `valF = valD;`
   d. None of the above
7) Consider the method `foo` having a header of: `void foo(float val)`. Which of the following calls of `foo` results in a compile-time error?
   a. `foo(valD)`
   b. `foo(str)`
   c. Both of the above
   d. None of the above

8) Which of the following correctly stores in `str` the sum of the values of `valF` and `valD`? Assume that `valF` was initialized to `100.0`.
   a. `str = "" + valF.doubleValue() + valD;`
   b. `str = (new Double(valF.doubleValue() + valD)).toString();`
   c. Both of the above
   d. None of the above
9) Assuming that `str` correctly receives the sum of `valF` and `valD` in (8), which of the following can be used to compute the sum of the first and last digits stored in `str`?
   a. `Integer.parseInt(str.charAt(0))+`
      `Integer.parseInt(str.charAt(str.length()-1))`
   b. `Integer.parseInt(str.substring(0,1))+`
      `Integer.parseInt(str.substring(str.length()-1))`
   c. Both of the above
   d. None of the above
10) Assuming that `str` is now storing a value of `123.456`. Which of the following correctly extracts the decimal point (`.`) from `str`?
   a. `str.charAt(str.length()-3)`
   b. `str.charAt(str.length()/2)`
   c. Both of the above
   d. None of the above
11) Given the value of `str` from the previous question, which of the following can be used to format the value in `str` and then store the formatted value in a variable called `val`? Assume that `fmt` is a `DecimalFormat` object that was instantiated properly.
   a. `double val = Double.parseDouble(fmt.format(str)) ;`
   b. `double val = fmt.format(Double.parseDouble(str));`
   c. `double val =`
      `Double.parseDouble(fmt.format(Double.parseDouble(str))) ;`
   d. None of the above
12) Which of the following statements can be used to select a character at random from `str` and then store it in a variable called `letter`?
   a. `char letter = str.charAt((int) Math.random()*str.length());`
   b. `char letter = str.charAt((int) str.length()*Math.random());`
   c. Both of the above
   d. None of the above
13) Which of the following packages must be imported before the variable `valF` can be used?
   a. `java.lang`
   b. `java.util`
   c. `java.text`
   d. None of the above
14) Which of the following correctly raises `valD` to the `valF`th power?
   a. `math.pow(valD, valF)`
   b. `math.pow(valF, valD)`
   c. `math.Pow(valD, valF)`
   d. None of the above

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

1. Applying the `private` visibility modifier to a method inside a helper class leads to a violation of the encapsulation principle.

Answer: True **False**

2. Consider the following method definition:
   ```
   public String foo(String str) {return str.substring(0,2);}
   ```
   The following code fragment that uses `foo` outputs: `Joe`
   ```
   System.out.print(foo(Joe));
   ```

Answer: True **False**

3. Instance variables of a given class have a wider scope than the local variables of that class but local variables live in the memory longer than instance variables.

Answer: True **False**

4. Static methods such as `parseDouble, random,` and `PI` can be invoked through the class name without having to create an object of the class.

Answer: True **False**

5. The following statement leads to a logical error:
   ```
   int val = (int) Math.random()*6 + 1;
   ```

Answer: **True** False

6. The following statement leads to a compile-time error:
   ```
   int val = Integer.parseInt("23.5");
   ```

Answer: True **False**

7. An instance variable and a local variable can have the same name in which case the `this` reserved word can be used to reference the local variable.

Answer: True **False**

8. Variables declared with private visibility are directly accessible only to the methods of the class in which they are declared.

Answer: **True** False

9. The following formatting pattern `"00.###"` indicates that at most 2 digits must be printed to the left of the decimal point.

Answer: True **False**

10. The number of actual parameters must match that of formal parameters when invoking a given method; otherwise, a run-time error occurs.

Answer: True **False**

11. An accessor method changes the state of an object created from the helper class.

Answer: True **False**

12. The following Java statement is valid: `Int val = new Int(23);`

Answer: True **False**

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

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

```
import java.text.DecimalFormat;
public class ClassB {
      private DecimalFormat fmt;
      private double val;
      public ClassB() {
      fmt = new DecimalFormat("00.##");
      val = 3.45;}
      public String first(int a) {
            return "" + val + a;}
      public String toString() {
            val =
            Double.parseDouble(
            first(7));
            return fmt.format(val);}
}
```

```
public class ClassBDriver {
public static void main(String[] args){
        ClassB b=new ClassB();
        System.out.println(b);
      }
}
```

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

a. `10.45`
b. `3.46`
c. `73.46`
d. It doesn't compile correctly
e. None of the above

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

```
public class ClassA {
      private Random rnd;
      private int x;
      public ClassA() {
            rnd = new Random();
            x = 1;
      }
      public void first(int a) {
            x = rnd.nextInt(1)*a;
      }
      public void second(int b) {
            first(x);
            x /= 7;
      }
      public int getX(){return x;}
}
```

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

      ClassA obj = new ClassA();
      obj.first(6);
      obj.second(7);

      double a = obj.getX();

      System.out.print("Answer is: " + a);
}
}
```

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

a. `Answer is: 0`
b. `Answer is: 7`
c. It is impossible to predict the output with certainty
d. It doesn't compile correctly
e. None of the above

Problem 4: Evaluating Java Expressions (**25 minutes**) [24 points]

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

(1)
```
String str = "The Revenant";
char x = str.charAt(str.length() -
str.substring(5, 8).length());
```
Answer: x= 'a'

(2)
```
int y = (int) Math.random()*20;
int x = 10*y + (++y);
```
Answer: x= 1

(3)
```
DecimalFormat fmt = new DecimalFormat("00.00");
double a = 1.234;
double b = 10;
b = a--;
String x = fmt.format(a);
x += 10;
```
Answer: x= "00.2310"

(4)
```
String str = "We rise by lifting others";
str = str.replace('i', 'o');
String x = str.substring(str.indexOf('o'),
str.indexOf('s'));
```
Answer: x="o"

(5)
```
String S = new String("\\//");
String x = (S.length() + "" + (2 + 3));
```
Answer: x= "35"

(6)
```
String Moe = "in love"; String Eeny = "Let all ";
String Miny = "be done "; String Meeny = "that you do ";
String x = Eeny + Meeny + Miny + Moe + "!";
```
Answer: x= "Let all that you do be done in love!"

(7)
```
String x = "2";
x = Double.toString(Math.pow(3, Integer.parseInt(x)))+ "0";
```
Answer: x= "9.00"

(8)
```
double y = 111.111;
double z = Math.floor(y*8);
double x = z - (int)y*5;
```
Answer: x= 333.0

(9)
```
DecimalFormat fmt = new DecimalFormat("0.##");
double a = 0.12;
double b = a/10;
String x = fmt.format(b);
x += b;
```
Answer: x= "0.010.012"

(10)
```
double u = 5.11;
double v = Math.ceil(u*10);
int x = (int)(Math.floor(v) - u*2);
```
Answer: x= 41

(11)
```
String x = "I have a good grade";
 x.replace('a' , 'o');
 x = x.replace('o' , 'x');
```
Answer: x= "I have a gxxd grade"

(12)
```
double y1 = 12.329;
 double y2 = Math.ceil(12.3456*100);
 double x = (int) y2 - (int) (y1*100);
```
Answer: x= 3.0

Problem 5: Coding (**40 minutes**) [40 points]

1. Write a program called `Series` which computes the last term `a_n` and the sum `S` of an arithmetic sequence. Note that an arithmetic sequence is a sequence of numbers such that the difference between any two consecutive terms is fixed and is referred to as the step. The program will read from the user the initial term `a_0`, the step value `d`, as well as the number of terms `n`. The program will then compute `a_n` and `S` using the following formulas:

   (1) $a\_n = a\_0 + (n + 1) \times d$

   (2) $S = \frac{n}{2} \times (2 \times a\_n + (n-1) \times d)$.

Sample run:

Enter initial value a_0: 1
Enter difference d: 2
Enter number of terms n: 3
The last term a_n: 9
The sum of the first 3 terms: 22

```
import java.util.Scanner;

public class Test
{
   public static void main(String [] args)
   {
        Scanner scan = new Scanner (System.in);

        int a_0, d, n, a_n;
        double S;

        System.out.print("Enter initial value a_0: ");
        a_0 = scan.nextInt();

        System.out.print("Enter difference d: ");
        d = scan.nextInt();

        System.out.print("Enter number of terms n: ");
        n = scan.nextInt();


        a_n = a_0 + (n + 1) * d;

        S = (n * (2 * a_n + (n-1) * d))/2.0;

        System.out.println("The last term a_n: " + a_n);
        System.out.println("The sum of the first " + n + " terms: " + S);

   }
}
```

2. Write a program called `Sec2HMS` which reads from the user a value representing a number of seconds, then prints the equivalent amount of time as a combination of hours, minutes, and seconds, as illustrated in the sample run enclosed below.

Sample run:

Enter time duration (in seconds): 9999
This time duration amounts to: 2 hours, 46 minutes, and 39 seconds

```
import java.util.Scanner;

public class Sec2HMS
{

    public static void main(String [] args)
   {

         Scanner scan = new Scanner (System.in);

               int S, H, M;

               System.out.print("Enter time duration (in seconds): ");
               S = scan.nextInt();

               H = S / 3600;
               M = (S % 3600) / 60;
               S = (S % 3600) % 60;

               System.out.println("The time duration amounts to: " + H +
                " hours, " + M + " minutes, and " + S + " seconds");

   }

}
```

3. Write a program called `Middle3` which reads from the user a `String` `S1` consisting of an odd number of characters, and then generates two new `String` objects `S2` and `S3` where `S2` consists of `S1`'s middle 3 characters, and `S3` consists of all the characters of `S1` except for its 3 middle characters.

Sample run:
Enter string S1 (having odd length): Johnathan
S2: nat
S3: Johhan

```
import java.util.Scanner;

public class Middle3
{

    public static void main(String [] args)
   {

         Scanner scan = new Scanner (System.in);

        String S1, S2, S3;

         System.out.print("Enter string S1 (having odd length): ");
         S1 = scan.nextLine();


        int x = S1.length()/2;
        S2 = S1.substring(x-1, x+2);

         S3 = S1.substring(0, x-1) + S1.substring(x+2);

        System.out.println("S2: " + S2);
        System.out.println("S3: " + S3);

   }

}
```

4. Write a Java program called `StringMutation` that reads a `String` from the user and then uses it to create a new `String` by removing two randomly selected characters from the input `String`. Your program should then print the resulting `String` out.

Sample run:
Enter a String: Salim Haddad
Randomly generated String is: Salm Hadda

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

public class StringMutation
{

    public static void main(String [] args)
   {

         Scanner scan = new Scanner (System.in);
               Random rnd = new Random();

               String S1, S2;

         System.out.print("Enter a String: ");
               S1 = scan.nextLine();

               int a = rnd.nextInt(S1.length());

               S2 = S1.substring(0, a) + S1.substring(a+1);

               System.out.print(S2);

               S1 = S2;

               a = rnd.nextInt(S1.length());

               S2 = S1.substring(0, a) + S1.substring(a+1);

               System.out.print("Randomly generated String is: " + S2);

   }

}
```