COE 212 – Engineering Programming

Welcome to Exam II
Friday November 27, 2015

Instructors: Dr. Salim Haddad
Dr. Bachir Habib
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 **120 minutes** to complete the 6 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 (**15 minutes)** [14 points]

For the questions given below, consider the following statement:

```
String str = "www.someSchool.com/someDept/someFile.jpg";
```

1) Which of the following statements creates a `Scanner` object called `scan` that can be used to parse the tokens separated by forward slash (`/`) characters in `str`?
   a. `Scanner scan = new Scanner(str, "/");`
   b. `Scanner scan = new Scanner(str); scan.usesDelimiters("/");`
   c. `Scanner scan = new Scanner(str); scan.usesDelimiter("/");`
   d. **None of the above**
2) Given the `scan` object created earlier, what output does the following statement produce?
   `System.out.print(scan.next() + " " + scan.hasNext());`
   a. `www.someSchool.com/ true`
   b. `www.someSchool.com false`
   c. `www.someSchool.com/someDept true`
   d. **None of the above**
3) Which of the following statements produces the same output as the following code fragment?

```
while(scan.hasNext())
      System.out.println(scan.next());
```

   Assume that these statements are executed immediately after the creation of the `scan` object given earlier.

   a.
```
while(str.indexOf('/') != -1) {
      System.out.println(str.substring(0, str.indexOf('/')));
      str = str.substring(str.indexOf('/')+1);}
```
   b.
```
while(str.indexOf('/') >= 0) {
      System.out.println(str.substring(0, str.indexOf('/')));
      str = str.substring(str.indexOf('/')+1, str.length()-1);}
```
   c. Both of the above
   d. **None of the above**
4) Which of the following **correctly counts** the number of **dots** (`.`) characters present in `str`?
   a.
```
int counter = 0;
for(int i=0; i<=str.length(); i++)
if(str.charAt(i) == '.') counter++;
```
   b.
```
int counter = 0;
for(int i=0; i<str.length(); i++)
if(str.charAt(i).equals(".")) counter++;
```
   c.
```
int counter = 0;
for(int i=0; i<str.length(); i++){
if(str.charAt(i)== '.') counter++;}
```
   d. None of the above
5) Which of the following statements prints on the same line the capital letters present in `str` in reverse order (backward)?
   a.
```
for(int i=str.length()-1;i>=0;i++)
if(str.charAt(i) >= 'A' && str.charAt(i) <= 'Z')
System.out.print(str.charAt(i));
```
   b.
```
for(int i=str.length()-1;i>=0;i--)
if(str.charAt(i) >= 'A' || str.charAt(i) <= 'Z')
System.out.print(str.charAt(i));
```
   c.
```
for(int i=str.length()-1;i>=0;i--)
if(str.charAt(i) >= 'A' && str.charAt(i) <= 'Z')
System.out.print(str.charAt(i));
```
   d. None of the above
6) What output does the following code fragment produce?

```
int counter=0;
for(int i=0; i<str.length();i++) {
switch(str.charAt(i)) {
case 'a': counter++; break;
case 'e': counter++; break;
```

```
case 'i': counter++; break;
case 'o': counter++; break;
case 'u': counter++;}}
System.out.print(counter);
```

a. `8`
b. `9`
c. `10`
d. **None of the above**

7) Which of the following correctly prints the first character of `str` followed by its last character?
   a. `System.out.print(str.charAt(0)+str.charAt(str.length()-1));`
   b. **`System.out.print(""+str.charAt(0)+str.charAt(str.length()-1));`**
   c. Both of the above
   d. None of the above
8) Which of the following correctly creates an object called `outToFile` that can be used to write `str` to an output file called `out.txt`?
   a. `Scanner outToFile = new Scanner(new PrintWriter("out.txt"));`
   b. `PrintWriter outToFile =`
      `new PrintWriter(new BufferedWriter(new FileWriter("out.txt")));`
   c. `BufferedWriter outToFile =`
      `new BufferedWriter(new FileWriter(new PrintWriter("out.txt")));`
   d. **None of the above**
9) Assuming that the `outToFile` object from the previous question is created properly, which of the following correctly writes `str` to `out.txt`?
   a. `outToFile.write(str); outToFile.close();`
   b. `System.out.print(str);`
   c. `outToFile.println(str);`
   d. **None of the above**
10) Not using `outToFile.close();` when writing data to the `out.txt` file leads to a
   a. Logical error
   b. An empty file
   c. **Both of the above**
   d. None of the above
11) Which of the following is added to the end of the header of the `main` method when reading data from or writing data to a file?
   a. **`throws IOException`**
   b. `throw IOException`
   c. Either of the above
   d. None of the above
12) Which of the following is implemented by the `String` class?
   a. `java.util.Comparable`
   b. `java.util.Iterator`
   c. `java.lang.Iterator`
   d. **None of the above**
13) Which of the following sets of methods are part of `Comparable`?
   a. `equals, equalsIgnoreCase`
   b. `equals, compareTo`
   c. **`compareTo`**
   d. None of the above
14) Which of the following can be used to determine whether or not the content of two `String` variables `str1` and `str2` are an exact match?
   a. `If(str1.equals(str2))`
   b. `if(str1.compareTo(str2)==true)`
   c. Both of the above
   d. **None of the above**

Problem 2: True or false questions (**15 minutes)** [16 points]

1. The following method correctly returns the maximum of its two input parameters `num1` and `num2`.
```
public int method1(int num1, int num2) {
       if(num1 > num2)
              return num1;}
```
Answer: True **False**

2. The following method correctly finds and returns the minimum of its three input parameters.
```
public int method2(int num1, int num2, int num3) {
       int min;
       if(num1<num2)
       if(num1<num3)min=num1;
       else min = num3;
       else if(num2 < num3)
       min = num2;
       else min = num3;
       return min;}
```
Answer: **True** False

3. Non-static methods cannot reference static variables since non-static methods do not operate in the context of a particular object.

Answer: True **False**

4. Consider the following static method that returns the sum of its two input parameters:
```
public static int method3(int num1, int num2) { return num1 + num2;}
```
The following method correctly uses `method3` to return the sum of its three input parameters:
```
public int method4(int num1, int num2, int num3) {
       return method3(num1, method3(num2, num3));}
```
Assume that `method3` and `method4` are part of the same class.

Answer: True **False**

5. The following code fragment outputs: `11`
```
int z=1;
for(;z<=10;) z++;
System.out.print(z);
```
Answer: **True** False

6. The following code fragment correctly computes the sum of the values from `1` (inclusive) to `100` (inclusive):
```
int sum;
for(int i=1; i<=100; i++) { sum=0; sum+=i;}
```
Answer: True **False**

7. The following code fragment correctly computes the product of the square root of the values from `1` (inclusive) to `100` (inclusive):
```
int prod = 1;
for(int i=1; i<=100; i++) prod *= Math.sqrt(i);
```
Answer: True **False**

8. The following code fragment correctly computes the sum of the digits contained in the `int` variable called `val`.
```
int sum=0;
while(val != 0)
       sum = sum + val % 10;
       val = val/10;
```
Answer: True **False**

9. Consider the following nested loops:
```
for(int i=1; i<=10; i++) {
       int j=0;
       while(true){if(j==10) break; j++;}
}
```
The body of the inner `while` loop is executed 100 times.

Answer: **True** False

10. The following Java code fragment fails to swap the values of `val1` and `val2`.

```
public static void main(String[] args) {
       int val1=1, val2=2;
       swap(val1, val2);
       System.out.println("Val1:"+val1+", val2:"+val2);
}
private static void swap(int num1, int num2) {
       int temp=num1;
       num1=num2;
       num2=temp;
}
```

Answer: **True** False

11. The following code fragment generates two random numbers `num1` and `num2` between `1` (inclusive) and `10` (inclusive) that are different from each other.

```
int num1 = (int) (Math.random()*10+1);
do {
       num2 = (int) (Math.random()*10 + 1);
} while(num1 == num2);
```

Answer: **True** False

12. The following statement prints `1 star` when `rating` is equal to `1` and the value of `rating` followed by a white space character and then the word `stars` otherwise.

```
System.out.println(rating + ((rating==1)? " stars":" star"));
```

Answer: True **False**

13. The following code fragment outputs `3` rows each of which consisting of `3` back to back asterisk (*) characters.

```
for(int i=1; i<=3; i++)
       for(int j=1; j<=3; j++) System.out.print("*");
```

Answer: True **False**

14. The following code fragment uses multiple overloaded versions of the `nextInt` method of the `Random` class.

```
Random rnd = new Random();
int x = rnd.nextInt();
int y = rnd.nextInt(6) + 1;
```

Answer: **True** False

15. The following code prints the lowest common multiple for the two `int` variables `num1` and `num2`.

```
int lcm = num1;
while(true) {
       if(lcm % num2 == 0)
              break;
       lcm += num1;
}
System.out.println(lcm);
```

Answer: **True** False

16. `PI` is a static method of the `Math` class.

Answer: True **False**

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

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

```
public class ClassA {
       private static int b=3;
       private int a;
       public ClassA() {a=2;}
       public void first(int c){
              a++; b+=c; c+=1;
       }
       public void second() {
              int b=1;
              a+=2; b+=2;
       }
       public void startIt() {
              int c=1;
              first(c);
              second();

       System.out.print(
              ""+a+b+c);}
}
```

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

      ClassA obj1 = new ClassA();
      ClassA obj2 = new ClassA();

      obj1.first(1);
      obj2.first(1);

      obj2.startIt();
}
}
```

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

a. `661`
b. `171`
c. `131`
d. `173`
e. None of the above

2) Consider the method given below. If its output is: `9876`

```
public void method2(int number){
      while(number >= 10) {
            System.out.print(number%10);
            number = number/10;
      }
}
```

3)

How was the method called?

a. `method2(16789)`
b. `method2(98761)`
c. `method2(26789)`
d. Either (a) or (c)
e. None of the above

Problem 4: Evaluating Java Expressions (**15 minutes**) [10 points]

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

(1)
```
int x=1, i=5;
do {
++i; x += i;
i*=2;
} while(i < 20);
x -= i;
```
Answer: x = -6

(2)
```
int answer = 2; int x = 10;
switch(answer) {
case 3:
     x /= 2;
case 2:
     x -= 2;
case 1:
     x *= 2;
     break;
case 0:
     x /= 2;
     break;
default:
     x =2;
}
```
Answer: x = 16

(3)
```
boolean x=false;
for(int i=-6; i<1; i++)
if(i/2 == 0)
x = (((2-i) < 0) && x)&& true;
```
Answer: x = false

(4)
```
int x=1, v;
for(v=x; v<10; v+=2) x +=v;
x--;
```
Answer: x = 25

(5)
```
int n=600; int x=n, i=1;
while(x>2) {
     x=x/2; i++;}
for(int j=1; j<i; j +=2)
     n= n/2;
x=n;
```
Answer: x = 37

(6)
```
int y = 90;
boolean x =(y%2 == 0 && y / 10 != 0) && (y / 5 == 0);
```

Answer: x = false

(7)
```
int x = 1;
for(int i=2; i <= 6; i++)
for(int j=3; j <= i; j++){
x += j;
j++;
}
```

Answer x = 23

(8)
```
String S = "Bond! James Bond!", x = "";
for(int i=1; i < S.length(); i+=3)
x += S.charAt(i);
```

Answer x = “o!aso!”

(9)
```
int z=0, w= 123, x=0;
do {
z = w%10;
if(z>x)
x=z;
w/=10;
} while(w > 3);
```

Answer: x = 3

(10)
```
int x=0, s;
for(s=++x; s<=3; ++s)
x+=s;
s++;
```

Answer: x = 7

Problem 5: Code Completion (**10 minutes**) [10 points]

You are given below an incomplete code for a class called `Voting` made up of two static methods called `inputAge` and `checkEligibility`, as well as a `main` method. Your job is to complete the definitions of the static methods `inputAge` and `checkEligibility`, and then invoke both of them inside the `main` method as per the provided guidelines.

1. `inputAge` is a method that does not accept any input parameters. It uses a `Scanner` object to read from the user her/his age as an integer value, and then verifies that the entered value is comprised between 1 (inclusive) and 130 (inclusive) (maximum age for a human being). The method then returns the age value obtained from the user.
2. `checkEligibility` is a method that accepts an integer parameter, which represents a person's age. The method then returns `true` if the person is eligible to vote, and returns `false` otherwise. Note that a person is considered to be eligible to vote if he/she is older than or equal to 18 years old.
3. `main` invokes both `inputAge` and `checkEligibility` methods, and prints on-screen `"Eligible to vote"` or `"Not eligible to vote"` based on the person's age.

```
import……;

class Voting{

public static void main (String [] args) {









}   // end of main method
```

```
private static…… inputAge(……){

} // end of inputAge method

private static…… checkEligibility(……){

} // end of checkEligibility method

}   // end of class Voting
```

```
import java.util.Scanner;

class Voting{



    public static void main (String [] args){
       int age = inputAge();
       if(checkEligibility(age) == true)
             System.out.println("Eligible to vote");
       else System.out.println("Not eligible to vote");
    }




    public static int inputAge(){
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter your age:");
        int age = scan.nextInt();

        while(age <0 || age >130)
        {
             System.out.print("Wrong input! Try again:");
             age = scan.nextInt();
        }
       return age;
    }



    public static boolean checkEligibility(int age){

        boolean eligible = false;
        if(age >= 18)
        {
           eligible = true;
        }
      return eligible;
    }
}
```

Problem 6: Coding (**45 minutes**) [40 points]

1. Write a Java program called `MultiplesOfThree` that reads a sequence of positive integer values from the user and counts the multiples of 3, until the program reads `0` (sentinel value) indicating the end of user input. The program then prints on-screen the number of multiples of 3 that were entered by the user. Note that the program has to make sure that the numbers provided by the user are positive**.**

   Sample output
   Enter a sequence of positive integers: 1 2 4 5 6 12 11 0
   Number multiples of *3*: 2

```
import java.util.Scanner;

class  MultiplesOfThree{

      public static void main (String [] args)
      {

        Scanner scan = new Scanner(System.in);


             System.out.print("Enter a sequence of positive integers: ");
             int x = scan.nextInt();
             int count = 0;

             while (x != 0){

                 if(x < 0)
                   {
                     System.out.println("Negative number! Try again!");
                 } else if(x%3 == 0) {
                     count++;
                 }

                 x = scan.nextInt();
             }

             System.out.println("\nNumber of multiples of '3':" + count);
      }
}
```

2. Write a Java program called `CountingCharacters` that reads three strings from the end user, then determines and prints how many times the letter ‘e’ appears in the first string, the letter ‘t’ appears in the second string, and the letter ‘n’ appears in the third string.

```
import java.util.Scanner;

public class CountingCharacters
{

   public static void main (String[] args) {

       Scanner scan = new Scanner(System.in);

       System.out.println("Enter three string: ");

       String S1 = scan.nextLine();
       String S2 = scan.nextLine();
       String S3 = scan.nextLine();

       int eCount = 0;
       int tCount = 0;
       int nCount = 0;

       for (int i=0; i<S1.length(); i++)
          if (S1.charAt(i) == 'e')   eCount++;

     for (int i=0; i<S2.length(); i++)
          if (S2.charAt(i) == 't')   tCount++;

       for (int i=0; i<S3.length(); i++)
          if (S3.charAt(i) == 'n')   nCount++;

        System.out.println("# of 'e' characters in 1st string: " + eCount);
        System.out.println("# of 't' characters in 2nd string: " + tCount);
        System.out.println("# of 'n' characters in 3rd string: " + nCount);

   }
}
```

3. Write a Java program called `DiagonalPrinting` which reads a line of text from the user and prints each word of the input line diagonally as illustrated in the sample output below. Assume that two consecutive words in the input line of text are separated by a single white space character. Note that to print a word consisting of 3 characters for example diagonally, precede the first character with 0 white spaces, the second character with 1 white space, and the third character with 2 white spaces.

Sample output

Enter a line of text: DEAR ALL

D
E
A
R
A
L
L

```
import java.util.Scanner;

public class DiagonalPrinting
{

   public static void main (String[] args) {

        Scanner scan = new Scanner(System.in);

        System.out.print("Provide a line of text: ");

        String S = scan.nextLine();

        Scanner scanS = new Scanner(S);

        while (scanS.hasNext())
        {
            String P = scanS.next();
              String padding = "";
              for(int i=0; i< P.length(); i++)
              {
                  System.out.println(padding + P.charAt(i));
                    padding += " ";
              }
        }
   }
}
```

4. Write a program called `IntReadWrite` which reads and processes the integers in a file. The program reads integers from a file (called `Ints1.txt`), increments them by 1, and then stores the new integers each on a separate line in a new file (called `Ints2.txt`). Note that the source file `Ints1.txt` is assumed to have two integer values per line and that the two values are separated by a white space character as illustrated below.

Sample input file `Ints1.txt` content:

```
10 40
4 6
```

Sample output file `Ints2.txt` contents:

```
11
41
5
7
```

```
import java.io.*;
import java.util.Scanner;

public class IntReadWrite
{

   public static void main (String[] args) throws IOException {

      int i;

      Scanner IntScan  = new Scanner (new File("Ints1.txt"));

      String file = "Ints2.txt";
      FileWriter fw = new FileWriter(file);
      PrintWriter outFile = new PrintWriter(fw);

      while (IntScan.hasNextInt())
      {
            i = IntScan.nextInt() + 1;
                outFile.println(i);
         }

         outFile.close();

   }
}
```