COE 212 – Engineering Programming

Welcome to the Final Exam
Tuesday December 15, 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 **135 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: Arrays (**20 minutes)** [12 points]

For the questions given below, consider the following Java program. Note that numbers are placed at the beginning of each statement to simplify the process of referencing these statements in the subsequent questions. So, the considered program compiles correctly and hence is syntactically valid.

```
1. import java.util.ArrayList;
2. public class Problem {
3.    public static void main(String[] args) {
4.          String[] arr = {“fun”, “extremely”, “is”, “Exam”};
5.          System.out.println(arr[2].charAt(arr[2].length()-1));
6.          ArrayList<String> list = new ArrayList<>();
7.          System.out.println(“Size: ” + list.size());
8.          System.out.println(“Substring: ”+list.get(1).substring(5));
9.          for(int i=0; i<arr.length; i++)
10.               list.add(arr[i]);
11.         int size = list.size();
12.         for(int j=0; j < size; j++)
13.               System.out.print(list.get(j) + “ ”);
14.         System.out.println();
15.         System.out.println(“Size: ” + list.size());
16.         int k=0;
17.         while(k < arr.length) {
18.               arr[k] = list.remove(list.size()-1);
19.               k++;}
20.         System.out.println(“K: ” + k);
21.         System.out.println(“Size: ” + list.size());
22.         for(int m=0; m<arr.length; m++)
23.               System.out.print(arr[m] + “ ”);}}
```

1) What output does the `println` statement at **line number 5** produce?
   a. y
   b. u
   c. m
   d. **None of the above**
2) What output does the `println` statement at line **number 7** produce?
   a. `Size: -1`
   b. **`Size: 0`**
   c. `Size: 4`
   d. None of the above
3) What output does the `println` statement at line **number 8** produce?
   a. `emely`
   b. `n`
   c. `am`
   d. **None of the above**
4) What output do the statements at lines **number 12 and 13** produce?
   a. **`fun extremely is Exam`**
   b. `Exam is extremely fun`
   c. `fun extremely is`
   d. None of the above
5) What output does the `println` statement at line **number 15** produce?
   a. `Size: 3`
   b. `Size: 0`
   c. **`Size: 4`**
   d. None of the above

6) What output does the `println` statement at line **number 20** produce?
   a. `K: 3`
   b. **`K: 4`**
   c. `K: 5`
   d. None of the above
7) What output does the `println` statement at line **number 21** produce?
   a. `Size: -1`
   b. **`Size: 0`**
   c. `Size: 4`
   d. None of the above
8) What output do the statements at lines **number 22 and 23** produce?
   a. `fun extremely is Exam`
   b. **`Exam is extremely fun`**
   c. `extremely is Exam`
   d. None of the above
9) Which of the following correctly creates an `ArrayList` that can be used to hold the lengths of the respective String values stored in the `arr` array?
   a. `ArrayList<int> lengthsList = new ArrayList();`
   b. `ArrayList<Int> lengthsList = new ArrayList();`
   c. `ArrayList<integer> lengthsList = new ArrayList();`
   d. **None of the above**
10) Assuming that the array list from the previous question was created properly, which of the following can be used to populate it with the appropriate values.
   a.
```
int counter; String str;
for(int i=0; i < arr.length(); i++) {
      str = arr[i];
      counter=0;
      for(int j=0; j < str.length; j++)
            counter++;
      lengthsList.add(counter);}
```
   b.
```
String str;
for(int i=0; i < arr.length; i++) {
      str = arr[i];
      lengthsList.add(str.length());}
```
   c. Both of the above
   d. None of the above
11) Consider the following statements:
```
int[][] a = {{3, 6, 7}, {8, 9, 12}};
int[][] b = a;
```
Which of the following is true?
   a. **The array called `b` consists of two rows**
   b. `b[0].length` returns a value of `2`
   c. Both of the above
   d. None of the above
12) Given the following declarations, which of the following variables are arrays?
```
int[] a, b;
int d[], c, e;
```
   a. `b, c, e`
   b. `a, b, c, e`
   c. **`a, b, d`**
   d. None of the above

Problem 2: True or false questions (**20 minutes)** [13 points]

1. Consider the following interface:

```
public interface Interface1 {
      public void method1();}
public interface Interface2{
      public void method2();}
```

A class that implements both interfaces must provide an implementation for both `method1` and `method2`; otherwise, a run-time error occurs.

Answer: True **False**

2. Consider the following parent class implementation:

```
public class Parent {
      public final void method1() {
            System.out.println("Inside parent class.");}
```

A child class derived from this parent class can override the implementation of `method1` as follows:

```
public class Child extends Parent {
      public final void method1() {
            System.out.println("Inside child class.");}
```

Answer: True **False**

3. In Java, a class that does not extend any other class is automatically derived from the `object` class.

Answer: True **False**

4. The following code fragment correctly prints the smallest value in the array called `arr`.

```
int[] arr = {1, 5, 3, 7};
int temp = arr[arr.length-1];
for(int i=arr.length-2; i>=0; i++)
      if(temp<arr[i]) temp = arr[i];
System.out.print(temp);
```

Answer: True **False**

5. The following code fragment correctly prints the index of the largest value in the array called `arr`.

```
int[] arr = {1, 3, 6, 4, 5};
int temp = 0;
for(int i=1; i < list.length; i++)
      if(arr[i]>temp) temp = i;
System.out.print(temp);
```

Answer: True **False**

6. The following method can be used to determine whether or not the input parameter called `val` has a fractional part (For example, 23.5 has a fractional part in which case `foo(23.5)` returns true while `23.0` does not and hence `foo(23.0)` returns false).

```
public boolean foo(double val) {
      return ((val – (int) val) > 0.0);}
```

Answer: **True** False

7. Any code written using a `do…while` loop can be rewritten using a `while` loop instead.

Answer: **True** False

8. The following code fragment prints: `I think therefore I am`

```
boolean[][] arr={{true,false},{false, true}};
if(arr[1][1]) System.out.print("I think ");
if(arr[2][2]) System.out.print("therefore I am");
```

Answer: True **False**

9. The following method correctly performs linear search trying to find the input parameter `k` in the array called `pool` returning true if `k` is found within `pool` and false otherwise.

```
public boolean foo(int[] pool, int k) {
      int index = pool.length-1;
      boolean found=false;
      while(!found && index >= 0) {
            if(pool[index]==k)
                  found = true;
            else index--;}
      return found;}
```

Answer: **True** False

10. The following code uses the selection sort algorithm to sort the input parameter array called `arr` in a descending order.

```
public void foo(int[] arr) {
      for(int index=0;index<arr.length;index++){
            int temp1, temp2=index;
            for(int scan=index+1;scan<arr.length;scan++)
                  if(arr[scan]>arr[temp2]) temp2=scan;
            temp1 = arr[index];
            arr[index] = arr[temp2];
            arr[temp2] = temp1;}}
```

Answer: **True** False

11. Consider a class called `Foo` that does not extend any other class. Assume that `Foo` has a `0` parameter constructor. The following code fragment prints: `They are not equal`

```
Foo foo1 = new Foo();
Foo foo2 = new Foo();
if(foo1.equals(foo2)) System.out.print("They are equal");
else System.out.print("They are not equal");
```

Answer: **True** False

12. Consider the following two dimensional array called `mat`:
`int[][] mat = {{2, 3, 4}, {5, 6, 7}};`
The statement: `System.out.print(mat.length());` prints a value of `2`.

Answer: True **False**

13. Method overriding is the process of creating inside the same class multiple methods having the same name but different headers. Method overloading on the other hand is the process of creating inside the child class a method having the same signature as one defined in the parent class.

Answer: True **False**

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

1) Consider the following Java program:

```
import java.util.ArrayList;
import java.util.Scanner;
public class Problem3a {
      public static void main(String[] args) {
            String[] names={“Salim”, “Wissam”, “Joe”, “Bachir”};
            Scanner scan = new Scanner(System.in);
            int k; String output;
            System.out.println(“Enter k:”);
            k = scan.nextInt();
            output = solve(names, k);
            System.out.println(“Output: ” + output);}
      private static String solve(String[] names, int k) {
            ArrayList<String> list = new ArrayList();
            String output = “”;
            for(int i=0; i<names.length; i++)
                  list.add(names[i]);
            while(list.size() > 1) {
                  for(int j=1; j <= k; j++)
                        list.add(list.remove(0));
                  list.remove(0);
            }
            output = list.remove(0);
            return output;}}
```

What output does the program produce if the user enters a value of `2` for the variable `k`?

a. Salim
b. Joe
c. Wissam
d. Bachir
e. Empty `String`

2) What is the output of the following Java program?

```
public class Problem3b {
  public static void main(String[]args){
      String[] arr=
      {“Joe”,“Bachir”,“Salim”,“Wissam”};
      String output = solve(arr);
      System.out.print(output);
  } // end of main method

 private static String solve(String[] arr){
      sort(arr);
      String output = arr[0];
      return output;
 } // end of solve method
```

```
private static void sort(String[] arr) {
      int k; String temp;
      for(int i=0;i<arr.length-1;i++){
       k = i;
       for(int j=i+1;j<arr.length;j++)
        if(arr[j].length()
           <arr[k].length())
            k=j;
        temp = arr[i];
        arr[i] = arr[k];
        arr[k] = temp;
       }
    }// end of sort method
} // end of public class
```

a. Salim
b. Joe
c. Wissam
d. Bachir
e. Empty `String`

Problem 4: Evaluating Java Expressions (**25 minutes**) [16 points]

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

(1)
```
int x = 4;
int [] T = { 1, 2, 3, 4};
for ( int n = 0; n < T.length; n++)
x = x - T[n%T.length];
```
Answer: x= **-6**

(2)
```
String x = "The Martian";
int S = x.length();
for(int i=0; i < S%3; i++) {
      x = x.concat(x.substring(i, i+3));
}
```
Answer: x= **"THE MARTIANTHEHE"**

(3)
```
int [] A = new int[5];
for (int i = 0; i < A.length; i+=2)

A[i] = i + 2;
int x=0;
for (int i = 0; i < A.length/2; i++)
x += A[i]+A[i+1];
```
Answer: x= **6**

(4)
```
int [] T = new int[5];
int x = -1;
for (int i = T.length - 1; i >= 0; i-=1 )
{T[i] = i + 3; x++;
x = T[0]+T[2];}
```
Answer: x= **8**

(5)
```
int x = 3;
int[] A = {0, 2, 9, 10, -6};
for(int i=0; i< A.length; i++)
{

   if(i%3 == 0)  {x = A[i]; A[i] = --x;}
   else { x = A[i]+1;}
 }
```
Answer: x= **-5**

(6)
```
int x = 0;
int [] arr = new int[5];
for (int k = arr.length - 1; k >= 0; k -=2 )
{arr[k] = k + 10 + x; x = arr[k];}
```
Answer: x= **36**

(7)
```
double x = -2;
ArrayList <Double> A = new ArrayList<Double>();
A.add(Math.pow(x, 2));
A.add(x++);
A.add(x);
for (int i=0; i<A.length; i++)
x += A.get(i);
```

Answer: x= **Syntactic Error**

For each of the following code fragments, what are the values of the array elements after the statements are executed? (6 pts)

(8)
```
int[] T = {1, -1, 2, -2, 3, -3};
for (int i = 1; i < T.length-1; i++) {
    T[i] = (T[i-1] + T[i+1])+1; }
```

Indexes	0	1	2	3	4	5
Values	**1**	**4**	**3**	**7**	**5**	**-3**

(9)
```
int[][] A = {{1, 1}, {2, 2}, {3, 3}};
    for (int i = 0; i < A[i].length; i++)
        for(int j=0; j<A.length-1; j++)
                A[i][j] = A[1][j] + A[i][1];
```

Indexes	0	1
0	**3**	**3**
1	**4**	**4**
2	**3**	**3**

(10)
```
int[][] A = {{1, 3, 5, 7}, {2, 4, 6, 8}};
for (int i = 0; i < A.length; i++)
   for(int j= A[i].length-1; j>0; j--)
      A[j%(i+1)][j]++;
```

Indexes	0	1	2	3
0	**1**	**4**	**7**	**8**
1	**2**	**5**	**6**	**9**

Problem 5: Code Analysis (**15 minutes**) [9 points]

Consider the classes *Price* and *Accounting* given below:

```
class Price {
   private int p;
   public Price(int p){ this.p = p;}
   public String toString(){ return "" + p; }
   public int getPrice(){return p;}
   public static void addTax(int p, double tax) {
      double total = p + (p*tax);
      System.out.println("Price $"+p+" with tax " + tax*100 + "% = $" + total);
      p = p + 1;
      tax += 0.1;
    }
   public static void addTax(Price P, double tax){
      double total = P.p + (P.p*tax);
      System.out.println("Price $"+P.p+" with tax "+tax*100+"% = $" + total);
      P.p = P.p +1;
      tax -= 0.1;
   }
}
```

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

   public static void main(String[] args){
   Price P1 = new Price(100);
   Price P2 = new Price(150);
   int P3 = 200; double tax = 0.3;

   Price.addTax(P1, tax);
   System.out.println("P1=$"+P1+", P2=$"+P2+", P3=$"+P3+", tax=" + tax*100 +"%");
   Price.addTax(P2.getPrice(), tax);
   System.out.println("P1=$"+P1+", P2=$"+P2+", P3=$"+P3+", tax=" + tax*100 +"%");
   Price.addTax(P3, tax);
   System.out.println("P1=$"+P1+", P2=$"+P2+", P3=$"+P3+", tax=" + tax*100 +"%");
 }
}
```

Provide in the box below the output that would be displayed on-screen when the class `ManipulatingGrades` is executed:

```
Price $100 with tax 30.0% = $130.0
P1=$101, P2=$150, P3=$200, tax=30.0%
Price $150 with tax 30.0% = $195.0
P1=$101, P2=$150, P3=$200, tax=30.0%
Price $200 with tax 30.0% = $260.0
P1=$101, P2=$150, P3=$200, tax=30.0%
```

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

1. The traditional approach toward computing the factorial of a given integer `n` consists of multiplying all the integer numbers from `1` to `n`. That is,

$$n! = 1 \times 2 \times ... \times n$$

Stirling's approximation (or Stirling's formula) is an approximation of factorials. It is a very powerful approximation leading to accurate results even for small values of `n` as follows:

$$n! = \frac{n^n \times \sqrt{2\pi n}}{e^n}$$

Write a Java program that reads a positive integer value from the user and then calculates its factorial using both of the above-presented formulas. Note that the formulas above can be used only if the value entered by the user is different from `0`; make sure to print a value of 1 out if the end user ever enters a value of `0` for `n`. Last but not least, the program is required to round Stirling's approximation of factorial to `2` decimal places.

Sample output:

Enter n: 4
Result of traditional approach: 24
Result of Stirling's approach: 23.51

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

public class Factorial
{
    public static void main(String [] args)
   {
         Scanner scan = new Scanner(System.in);
         int n, fact1 = 1;
         double fact2 = 1;

         do{
            System.out.print("Enter n:");
            n = scan.nextInt();
         } while(n < 0);

         if(n >= 1)
         {
               // Exact method
                   for(int i=2; i<= n; i++) fact1 *= i;

                // Approximate method
            fact2 = (Math.pow(n, n) * Math.sqrt(2 * Math.PI * n))/ Math.exp(n);
          }

         DecimalFormat fmt = new DecimalFormat("0.##");

         System.out.println("Result of traditional approach: " + fact1);
         System.out.println("Result of Stirling's approach: " + fmt.format(fact2));
       }
}
```

2. Write a program called `MiddleElement` which reads from the user first the sizes of two one-dimensional arrays of integers `A` and `B`. Your program has to make sure that the sizes of arrays `A` and `B` are positive and odd. Then, the program must fill up `A` and `B` with user-supplied values. Finally, the program must create a new array `C` of length 2 containing the middle elements of `A` and `B` before printing these middle elements out.

Sample output;

Enter size of array A: 3
Enter size of second array B: 6
The size of array B has to be positive and odd! Please try again: 5
Enter integer elements of array A: 12 14 90
Enter integer elements of array B: 10 3 5 12 4
Middle elements are: 14 and 5

```
import java.util.Scanner;

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

          int nA, nB;

          System.out.print("Enter size or array A: ");
          nA = scan.nextInt();

          while(nA<0 || nA%2 ==0){
             System.out.print("The size of array A has to be positive and odd!” +
                              “ Please try again: ");
             nA = scan.nextInt();
          }

          System.out.print("Enter size or array B: ");
          nB = scan.nextInt();

          while(nB<0 || nB%2 ==0){
             System.out.print("The size of array B has to be positive and odd!” +
                             “ Please try again: ");
             nB = scan.nextInt();
          }

          int [] A = new int[nA];
          int [] B = new int[nB];

          System.out.print("Enter the elements of array A: ");
              for(int i=0; i< A.length; i++)   A[i] = scan.nextInt();

          System.out.print("Enter the elements of array A: ");
              for(int i=0; i< B.length; i++)   B[i] = scan.nextInt();

          int [] C = new int[2]; C[0] = A[A.length/2]; C[1] = B[B.length/2];

          System.out.print("Middle elements are: ");
         for(int i=0; i< C.length; i++)   System.out.print(C[i] + "  ");
    }
}
```

3. Write a program called `Special_3` which reads the elements of an array of integer values from the user, and then searches the elements of the array and checks whether there is an element = 2 that is immediately followed by an element = 3. Each time such a combination is located in the array, your program must change the value of the element that equals 3 to 0. Finally, your program must print out all the elements of the resulting modified array.

   Sample output;

   Enter the number of elements in the array: 4
   Enter the elements of the array: 3 2 3 13
   The updated elements of the array: 3 2 0 13

```
import java.util.Scanner;

public class Special_3
{
    public static void main(String [] args)
   {
         Scanner scan = new Scanner(System.in);

         // Reading array size
         int n;
         do{
            System.out.print("Enter the number of elements in the array: ");
            n = scan.nextInt();
         } while(n < 0);

         int [] A = new int[n];

         // Reading elements of array
         System.out.print("Enter the elements of the array: ");
       for(int i=0; i< A.length; i++)   A[i] = scan.nextInt();

         // Processing elements of array
       for(int i=0; i< A.length-1; i++)
           if(A[i] == 2 && A[i+1] == 3) A[i+1] = 0;

         // Printing elements of modified array
         System.out.print("The updated elements of the array: ");
       for(int i=0; i< A.length; i++)   System.out.print(A[i] + "  ");
   }
}
```

4. Write a program called `GradeBookStats` which reads from the user the grades of a number of students obtained on a number of courses storing them in a 2-dimensional array where each row represents the grades of a given student, and each column represents the grades on a given course. The program then computes and prints on screen the average grades per course as well as the average grades per student.

   Sample output;

   Enter the number of students: 2
   Enter the number of courses: 3
   Enter grades for student # 0 on all 3 courses: 70 80 95
   Enter grades for student # 1 on all 3 courses: 60 70 85
   The avg grade for course # 0: avg = 65.0
   The avg grade for course # 1: avg = 75.0
   The avg grade for course # 2: avg = 90.0
   The avg grade for student #0: avg = 81.67
   The avg grade for student #1: avg = 71.67

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

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

       Scanner scan = new Scanner(System.in);
       int NbStudents, NbCourses;

        do {   System.out.print("Enter the number of students: ");
               NbStudents = scan.nextInt();
        } while(NbStudents < 0);

        do{   System.out.print("Enter the number of courses: ");
              NbCourses = scan.nextInt();
        } while(NbCourses < 0);

      int [][] grades = new int [NbStudents][NbCourses];

      for(int i=0; i< NbStudents; i++) {
         System.out.print("Enter grades for student # "+ i + " on all " +
                          NbCourses + " courses: ");
           for (int j=0; j< NbCourses; j++)grades[i][j] = scan.nextInt();}

        double [] AvgPerStudent = new double[NbStudents];
        double [] AvgPerCourse = new double[NbCourses];

        for(int i=0; i< NbStudents; i++){
            int sumPerStudent = 0;
            for (int j=0; j< NbCourses; j++) sumPerStudent += grades[i][j];
            AvgPerStudent[i] = sumPerStudent / (double)NbCourses; }

       for(int j=0; j< NbCourses; j++){
          int sumPerCourse= 0;
          for (int i=0; i< NbStudents; i++) sumPerCourse += grades[i][j];
           AvgPerCourse [j] = sumPerCourse / (double)NbStudents; }

       DecimalFormat fmt = new DecimalFormat("0.##");

       for(int i=0; i< AvgPerCourse.length; i++)
           System.out.println("The avg grade for course # " + i + ": " +
                              fmt.format(AvgPerCourse[i]));

        for(int i=0; i< AvgPerStudent.length; i++)
            System.out.println("The avg grade for student # " + i + ": " +
                               fmt.format(AvgPerStudent[i]));
   }   }
```