**COE 211/COE 212 – Computer Programming/Engineering Programming**

Welcome to

The Comprehensive Final Exam

Monday January 21 2012

Instructor: Dr. Maurice Khabbaz

 Dr. Georges Sakr

 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 **130** **minutes** to complete the 7 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:** Inheritance (**10 minutes)** [10 points]

For each question choose the **single** right answer.

1. Inheritance is also known as the
	1. “has-a” relationship
	2. “uses-a” relationship
	3. “knows-a” relationship
	4. **None of the above**
2. Which of the following is true about inheritance
	1. All the methods of the parent class are inherited by the child class
	2. All the variables of the superclass can be accessed by the subclass
	3. All of the above
	4. **None of the above**
3. Which of the following keywords allows a subclass to access a parent class method even when the sub class has overridden the base class method :
	1. base
	2. **super**
	3. this
	4. extends
4. Every class in Java extends an existing class. Which one below doesn’t extend any class
	1. Integer
	2. String
	3. **Object**
	4. None of the above
5. Overriding a method differs from overloading a method in that
6. Overloaded versions and original version of a method have the same signature
7. **Original version and overridden versions of a method have the same signature**
8. Both of the above
9. None of the above
10. private members of a superclass can be accessed in a subclass
	1. by calling private methods declared in the superclass
	2. **by calling public or protected methods declared in the superclass**
	3. directly
	4. none of the above
11. Which of the following methods is/are inherited by every class created in Java?
12. toString
13. compareTo
14. equals
15. **both (a) and (c)**
16. both (a) and (b)
17. Abstract methods are used when defining
18. interfaces
19. abstract classes
20. **All of the above**
21. None of the above
22. If a subclass does not make an explicit call to a superclass’s default constructor, then
	1. A compile time error will result
	2. **The parent’s default constructor will be called anyway**
	3. The class will be implicitly declared a abstract
	4. A run time error will result
23. All Java classes are subclasses of
	1. Class
	2. object
	3. String
	4. **None of the above**

**Problem 2:** Code analysis (**20** **minutes)** [16 points]

1. Fill in the values for the array called world after the corresponding Java code has run. You must enter your final answers into the boxes provided.

 char[] world=new char[5];

for (int i=0; i < world.length; i++) {

 world[i] = (char) (i + ‘C’);

 }

**Answer:**

*C D E F G*

 **world**

 int[] world = {3, 6, 4, 7, 5, 8};

 int c, x = world.length;

 for(int i=0; i<x/2; i++) {

 c = world[i];

 world[i] = world[x-i-1];

 world[x-i-1] = c;}

**Answer:**

 *8 5 7 4 6 3*

 **world**

 int[] B = {10, 12, 14, 16};

 int[][] world = new int[B.length/2][B.length-1];

 for(int i=0; i<B.length/2; i++) {

 for(int j=0; j<B.length/2; j++)

world[i][j]=B[2\*i+j];}

 **world**

14 16 0

10 12 0

**Answer:**

1. What is the output of the code given in the two columns below when an object of type Confuse is created and used to call the method called startUp()?

public class Confuse{

 private int b;

 private int a;

 public Confuse(){

 b=3; a=2;

 }

 private void first(int x){

 a+=x;

 }

private void second(int y,int z){

 setAB((a\*y), (b/z));

}

private void setAB(int b, int a){

 this.b = b;

 this.a = a;

}

private void display(){

 System.out.println(“”+b+a);

}

public void startUp() {

 first(b);

 second(b, a);

 display();

}

1. 125
2. 16
3. 251
4. **150**
5. None of the above

**Problem 3:** True or false (**10** **minutes)** [10 points]

1. The length() method is used to determine the lengths of both strings and arrays.

Answer: True **False**

1. A static method cannot access non-static instance variables but can access its non-static local variables.

Answer: **True** False

1. Every method must have a return type declared unless the method doesn’t return any value, in which case the return type can be left off.

Answer: True **False**

1. To swap the 1st and the last elements in an array of int values called list, you would do:

int temp = list[0];

list[0] = list[length-1];

list[length-1] = temp;

Answer: True **False**

1. A class can only have a single constructor.

Answer: True **False**

1. Given the following method definition:

 int foo(int... list) {

 int prod = 1, n = list.length;

 for(int i=0; i < n; i++)

 prod\*=list[i];

 return (n>=0) ? prod : 0;}

After the following statement executes: double x = foo(); x will hold a value of 1.0.

Answer: **True**  False

1. The following code in Java stores in the variable called sum the sum of positive even integers that are less than or equal to 10:

int sum=0, x;

 for(x = -2; x <= 10; x+=2)

if(x > 0) sum += x; else break;

Answer: True **False**

1. If a class called Transactioncontains a non-static variable called howMany, then each instance of the class will get its own copy of the variable howMany.

Answer: **True** False

1. The following is a valid declaration in Java: int Class = 3;

Answer: **True** False

1. The index of an array reference can itself be an array element, such as in: theArray[theArray[3]].

Answer: **True**  False

**Problem 4:** Multiple choice questions (**30** **minutes)** [18 points]

1. Consider the method given below. What output does it produce?

void method1(int[] theNumbers){

double x = 0;

 for(int i=0 ; i<theNumbers.length ;i++)

 x = x + theNumbers[i];

 System.out.println(x / theNumbers.length) ;

}

1. The minimum value in the array theNumbers
2. The maximum value in the array theNumbers
3. The sum of the values in the array theNumbers
4. **None of the above**
5. Consider the method given below. Its output is:

void method2(int[] theNumbers) {

 int x = theNumbers[0];

 for(int i=0; i<theNumbers.length; i++) {

 if(theNumbers[i] > x)

 x = theNumbers[i];

 }

 System.out.println(x);}

1. The minimum value in the array theNumbers
2. **The maximum value in the array theNumbers**
3. The sum of the values in the array theNumbers
4. None of the above
5. What is the return value of the method given below?

int method4(){

 int i, value = 0;

 int[] theNumbers = new int[7];

 for(i=0; i<theNumbers.length-1; i++) {

 theNumbers[i] = 2\*i+3;

 }

 value = theNumbers[theNumbers[theNumbers[i]]];

 return value;

}

1. 3
2. **9**
3. 11
4. 7
5. None of the above
6. Consider the method given below. It can be described as a method that:

boolean method5(int number){

 if(number == 1 || number == 2) return true;

 for(int i=2; i < number/2; i++) {

 if(number%i == 0)

 return false;

 }

 return true;

}

1. checks to see if a number is even
2. checks to see if a number is odd
3. **checks to see if a number is prime**
4. does not compile
5. none of the above
6. Consider the code given below. If its output is:

12 16 20

15 20 25

18 24 30

21 28 35

What are the values of variables start, end, first, and last?

for(int i=start; i<=end; i++) {

 for(int j=first; j<last; j++){

 System.out.print(i\*j + “ ”);

 }

 System.out.println();

}

1. **start=4, end=7, first=3, last=6**
2. start=4, end=3, first=6, last=4
3. start=7, end=4, first=6, last=3
4. start=3, end=7, first=4, last=6
5. None of the above
6. Consider the method given below. Its output is:

void method3(){

 String s = “12358”;

 int value = s.length();

 for(int i=0; i < value/2; i++) {

 s = s.substring(1, s.length()-1);

 }

 System.out.println(s);

}

1. 358
2. 235
3. 123
4. **3**
5. None of the above

**Problem 5:** Method definition (**15 minutes)** [9 points]

You are given below the headers of 3 methods called reverseArray, incrementByMin, and pickRandomValue. Your job is to complete the definition of each one of these methods as per the provided guidelines.

1. **reverseArray** is a method that accepts an array of int values called arr and prints its elements backwards.

public void reverseArray(int[] arr) {

 for (int i=arr.length-1;i>=0;i++)

 System.out.print(arr[i]);

 }

1. **incrementByMin** is a method that increments each of the array elements found in the input array of int values called arr by the value of the minimum element in arr.

public void incrementByMin(int[] arr) {

 int min = Integer.MAX\_VALUE;

 for (int i=0;i<arr.length;i++)

 if (arr[i]<min)

 min = arr[i];

 for (int i=0;i<arr.length;i++)

 arr[i] += min;

}

1. **pickRandomValue** is amethodthat accepts as a parameter a 2-dimensional array of int values called arr and returns an int value that is randomly selected from arr.

public int pickRandomValue(int[][] arr) {

 int i = (int)(Math.random()\*arr.length);

 int j = (int)(Math.random()\*arr[0].length);

 return arr[i][j];

 }

**Problem 6:** Completing code fragments (**15** **minutes)** [12 points]

Complete the definition of the class Polynomials given below. This class represents a second degree polynomial of the form: $f(x) = ax^{2}+bx+c$. The three coefficients a, b, and c are stored in 1-D array called coef that serves as an instance variable for Polynomials. This class has two additional String instance variables x1 and x2 representing the roots of the polynomial $f(x).$

 public class Polynomials {

 // **Put the instance variables here**

double[] coef;

String x1, x2;

// **Constructor: instantiate** coef**, then**

//**insert** a**,** b **and** c **as its 1st, 2nd and 3rd elements**

// **and initialize the two roots to** “no solution”**.**

 public Polynomials(double a, double b, double c) {

 coef = {a, b, c};

 x1 = “no solution”;

 x2 = “no solution”;

 }

// **the method** eval **evaluates and returns the value**

// **of** $f(x)$ **at a certain** x **received as a parameter.**

 public double eval(double x) {

 return (coef[0]\*Math.pow(x,2)+coef[1]\*x+coef[2]);

 }

// **the method** delta **returns the value of the**

// **discriminant** $Δ = b^{2} – 4ac$**.**

 public double delta() {

 return (Math.pow(coef[1],2)-4\*coef[0]\*coef[2]);

 }

// **Compute the roots of** $f(x)$ **and store them in**

// **their corresponding instance variables.**

// **Roots can be computed using the following**

// **formula:** $x\_{1}=\frac{-b+\sqrt{Δ}}{2a} $ **and** $x\_{2}=\frac{-b-\sqrt{Δ}}{2a}$

 public void solve() {

 double delta = delta();

 if (delta >= 0){

x1 = “” + (coef[1]+

Math.sqrt(dela))/((double)(2\*coef[0]));

x2 = “” + (-coef[1]-Math.sqrt(dela))/((double)(2\*coef[0]));

 }

 }

}// **end Polynomials**

**Problem 7:** Coding (**30** **minutes**) [25 points]

1. Write a JAVA program that reads an integer input S from the user. Obviously, S is a sequence of numerical digits with no spaces. The program will convert S into an array of integers digitsArr where each of that array’s elements represents one of the digits in S. Then, the program should find and display the sum of the elements in digitsArr. Finally, the program will extract the value lastElmnt of the last element in digitsArr, print it out and display a count of the number of digits within digitsArr that are less than lastElmnt.

**Sample input:** 146515
**Sample output:**

Sum of digits is: 22
Last digit is: 5
Number of digits less than 5 is: 3

import java.util.Scanner;

public class code1{

 public static void main(String[] args){

 Scanner scan = new Scanner(System.in);

 System.out.print(“Enter a sequence of ints:”);

 S = scan.nextLine();

 int[] digitsArr = new int[S.length()];

 int sum = 0, lastElmnt, count = 0;

 for (int i=0; i<S.length(); i++){

digitsArr[i] = Integer.parseInt(“”+s.charAt(i));

 sum += digitsArr[i];

 }

 lastElmnt = digitsArr[digitsArr.length-1];

 for (int i=0; i<digitsArr.length; i++){

 if(digitsArr[i] < lastElmnt)

 count++;

 System.out.println(“Sum of digits is: “ + sum);

 System.out.println(“Last digit is: “+ lastElmnt);

 System.out.println(“Number of digits less than ”+

 lastElmnt + “ is:” + count);

 }

}

1. Write a JAVA program that reads in 10 numbers into an array numArr. Then, the program should compute the product of all the elements in numArr. In addition, the program must increment each of the elements in numArr by the value of the 10th element in this array.

**An example of running this program is:**

Please enter 10 numbers: 5 3 2 8 14 22 5 3 20 3

Product of numbers is: 66528000

The 10th element is: 3

New array content is: 8 6 5 11 17 25 8 6 23 6

import java.util.Scanner;

public class code2{

 public static void main (String[] args){

 Scanner scan = new Scanner(System.in);

 System.out.println(“Enter 10 numbers:”);

 int[] numArr = new int[10];

 int product = 1;

 for (int i=0; i<numArr.length;i++){

 numArr[i] = scan.nextInt();

 product \*= numArr[i];

 }

 int tenthElmnt = numArr[9];

 for (int i=0; i<numArr.length; i++)

 numArr[i] += tenthElmnt;

 }

}

1. The coordinate values of a vector of length n are grouped into a 1-D array of n elements. Write a JAVA program that reads in a value for n followed by the n coordinate values of two vectors X and Y. The program, then, computes the *Euclidean distance* between these two vectors. The *Euclidian* *Distance* between two vectors $X\left(x\_{1},x\_{2},…,x\_{n}\right)$ and $Y\left(y\_{1},y\_{2},…,y\_{n}\right)$ is given by: $d=\sqrt{\left(x\_{1}-y\_{1}\right)^{2}+\left(x\_{2}-y\_{2}\right)^{2}+…+\left(x\_{n}-y\_{n}\right)^{2}}$

**Sample run:**

Enter n: 2

Enter x1: 1

Enter x2: 2

Enter y1: 0

Enter y2: 1

Eucledian Distance: 1.4142…

import java.util.Scanner;

public class code3 {

 public static void main(String[] args){

 Scanner scan = new Scanner(System.in);

 System.out.print (“Enter n: “);

 int n = scan.nextInt();

 double[] X = new int[n];

 double[] Y = new int[n];

 for (int i=0; i<n; i++){

 System.out.print(“Enter x”+(i+1)+

 ”: ”);

 X[i]=scan.nextDouble();

 }

 for (int i=0; i<n; i++){

 System.out.print(“Enter y”+(i+1)+

 ”: ”);

 Y[i]=scan.nextDouble();

 }

 double sum = 0, d;

 for (int i=0;i<n;i++)

 sum+=Math.pow(x[i]-y[i], 2);

 d = Math.sqrt(sum);

 System.out.println(“Eucledian Distance:“

 + d);

 }

}