Queens College, CUNY
Department of Computer Science

CS 212 – Object-Oriented Programming in Java -- Exam 1 – Practice B

Last Name_____ First Name _____ Seat_____

Directions:

Read the whole question before answering. Proper use of Java concepts is expected; minor syntax errors will be overlooked.

1. (25 points) Write a Java application that accepts one String parameter from a *JOptionPane inputDialog* and prints to a *JOptionPane messageDialog* the number times *two* digits in a row occur. For example:

Input string:	messageDialog says
`Testing 12`	`Two digits happen 1 time.`
`Testing 123`	`Two digits happen 2 times.`
`123 Testing 12 23 234`	`Two digits happen 6 times.`

Hint: Recall that there is a method in the Character wrapper class:

```
boolean Character.isDigit (char c)
```

```
public static void main(String[] args) {
    String input =  JOptionPane.showInputDialog(null, "Enter strir
    int count = 0;

    for (int i = 0; i < input.length() - 1; i++) {
        char c1 = input.charAt(i);
        char c2 = input.charAt(i + 1);

        if (Character.isDigit(c1) && Character.isDigit(c2)) {
            count++;
        }
    }

    System.out.print("The two digits happen " + count + " times."
}
```

Continue the back of the page as needed

2. (25 points) Write a boolean method that will return true if the sum of the diagonal (upper-left to lower-right) of a two-dimension array is equal to the sum of a given row:

```
diagonalEqualsRow ( int [][] myArray, int row)
```

If the array is not a square matrix of the row number is out of bounds, throw an IllegalArgumentException with an appropriate message.

For example:

myArray1

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

myArray2

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

myArray3

1	3	5	6	7
0	8	7	4	3

myArray4

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

```
diagonalEqualsRow (myArray1, 0) returns true
diagonalEqualsRow (myArray2, 2) returns true
diagonalEqualsRow (myArray3, 0) throw an exception – not a square matrix
diagonalEqualsRow (myArray1, 6) throw an exception – illegal row
```

```
public boolean diagonalEqualsRow(int [][] myArray, int row) {
    int n = myArray.length;

    if (n != myArray[0].length) {
        throw new IllegalArgumentException("Not a square matrix!
    }

    if (row >= n) {
        throw new IllegalArgumentException("Illegal row! >:(");
    }

    int sum1 = 0;
    for (int i = 0; i < n; i++) { // Sum for the row
        sum1 += myArray[row][i];
    }

    int sum2 = 0;
    for (int i = 0; i < n; i++) { // Sum for the diagonal
        sum2 += myArray[i][i];
    }
}
```

```
// How to do it in one loop
int sum1 = 0, sum2 = 0;
for (int i = 0; i < n; i++) {
    sum1 += myArray[row][i];
    sum2 += myArray[i][i];
}
```

Continue the back of the page as needed

3. (25 points) Create the following class. Read the entire question first!

Write a class `Distance` with the following specifications:

Class variables:
- a private static integer called INCHES_IN_A_FOOT with a value 12;
- a private static float called FEET_IN_A_METER with a value of 0.3048f;

Instance variables:
- Two integer instance variables: feet and inches;

Constructors:
- a no argument constructor to initialize a Distance with zero feet, zero inches.
- a two argument constructor that will accept two positive integers and for feet and inches. An "IllegalArgumentException" is thrown if the number of inches is more than 11.

Get/Set methods:
- get methods for the instance variables;
- set methods for instance variables.

Public methods:

1. a method add() that adds another Distance object to itself. That is,

```
w1 = new Distance(4,9);
w2 = new Distance (3,6);
w1.add(w2);                 // w1 becomes 8 feet, 3 inches
```

2. a method metricDistance() that will return (as a float) the number of meters in the distance represented by the object.

That is, for w1 above, w1.metricDistance() returns 1.4478f

```
public class Distance {
    // class variables
    private static int INCHES_IN_A_FOOT = 12;
    private static float FEET_IN_A_MEETER = 0.3048f;

    // instance variables
    private int feet, inches;

    // constructors
    public Distance() {
        feet = 0;
        inches = 0;
    }

    public Distance(int f, int i) {
        if (f < 0 || i < 0 || i > 11) {
            throw new IllegalArguementException("Invalid inputs!
        }
        feet = f;
        inches = i;
    }

    public int getInches() {
        return inches;
    }

    public float getFeet() {
        return feet;
    }

    public void setInches(int i) {
        if (i < 0 || i > 11) {
            throw new IllegalArgumentException("Invalid input! >
        }
        inches = i;
    }

    public void setFeet(int f) {
        if (f < 0) {
            throw new IllegalArgumentException("Invalid input! >
        }
        feet = f;
    }
}
```

```
public void add(Distance d) {
    f = w1.getFeet() + w2.getFeet();
    i = w1.getInches() + w2.getInches

    if (i > 11) {
        f++;
        i = i % 12; // i %= 12
    }

    w1.setFeet(f);
    w2.setInches(i);
}

public float metricDistance() {
    float f = (float) feet;
    f += (float) inches / 12;

    return f *FEET_IN_A_METER;
}

}
```

Continue the back of the page as needed

4. (25 points) What will be printed by this program?

```
public class Question4 {
   static String s1,s2,s3,s4;
   public static void main (String[] args){

      s1 = new String("CAT");
      s2 = new String("DOG");
      s3 = new String("RAT");
      animal(s1,s2);
      System.out.println(mixThem(s2,s3));

   } //main
   private static void animal (String a1, String a2) {
      if(a1.equals(a2))
         System.out.println(a1);
      else
         System.out.println(a2);
   }
   private static String mixThem (String b1, String b2){
      String c1 = b1;
      b1 = b2;
      b2 = new String ("HAT");
      System.out.println(b1);
      System.out.println(b2);
      System.out.println(c1);
      System.out.println(s2);
      return b2;
   }

} //Question4
```

Answer here:

DOG

RAT

HAT

DOG

DOG

HAT