0112562 OBJECT ORIENTED PROG., Section 3, 2nd MIDTERM EXAM (11/5/2015)

Duration:	90mins.		**Score:**				**Student Nr:**	**Signature:**
Grading:	**1**	**2**	**3**	**4**	**5**	**6**	**Name, Surname:**	
	10	30	10	20	15	15		

Question 1:

```
public static void testTryCatch( int b ) {
   try {
      System.out.print("A");
      int a = 1 / b ; //generates ArithmeticException if b == 0
      System.out.print("B");
   }
   catch (ArithmeticException e) {
         System.out.print("C");
   }
   finally {
         System.out.print("D");
   }
}
```

What will be printed for the following method calls?
(1) testTryCatch ((last_digit_of_your_student_nr) % 2)
(2) testTryCatch ((last_digit_of_your_student_nr+1) % 2)

Question 2:
The **Assistants** class maintains, for a set of courses, the set of TAs for each course.

```
public class Assistants {
Map<String, Set<String>> map;
        public Assistants() {
                // YOU MUST IMPLEMENT THIS METHOD
        }
        public void addTA(String course, String taName) {
                // YOU MUST IMPLEMENT THIS METHOD
        }
        public void displayTAsPerCourse() {
                // YOU MUST IMPLEMENT THIS METHOD
        }
}
```

Anwer the following questions:
(1) Implement a constructor for Assistants that creates an empty map.
(2) Implement the **addTA** method that adds a teaching assistant to a specific course. A map entry for the course must be created if one does not exist.
(3) Implement the **displayTAsPerCourse** method that prints (using System.out.println) the name of a course followed by the TAs for the course.

Question 3: Rewrite the following code so that the messages are printed by two different threads, with one thread printing the "ABC" messages and the other thread printing the "XYZ" messages. Give two different possible outputs that could be generated by your rewritten code.

```
public class MessagePrinting {
        public static void main(String[] args) {
          for (int i=0; i< 3; i++)
                System.out.println("ABC"+i);
          for (int i=0; i<3; i++)
                System.out.println("XYZ"+i);
        }
}
```

Question 4: The Grades class keeps tracks of students that have received an "A", "B", "C", "D" or "F" in a course. The class will use a HashMap to map a letter grade with a list of students that have received such letter grade. A description of the methods you must implement is provided below.

```
import java.util.*;
public class Grades {
private double[] cutoffs;
private static final String[] letterGrades = {"A", "B", "C", "D", "F"};
        // Hashmap Definition Here
        public Grades(double[] cutoffs) {
                // You must implement this method
        }
        public String findLetterGrade(double numericGrade) {
                // You must implement this method
        }
        public void add(String studentName, double numericGrade)
        {
                // You must implement this method
        }
        public List<String> get(String letterGrade) {
                // You must implement this method
        }
}
```

(1) Provide a definition of the HashMap necessary to keep track of those students that have a particular letter grade. This definition would appear where you see the comments "// Hashmap Definition Here" above.
(2) Implement the body for the constructor specified above. The cutoffs parameter specifies the cutoffs to be used for an "A", "B", "C", and "D". Any score below the "D" cutoff will be assigned an "F" letter grade.
(3) Implement the findLetterGrade method. The method will return the letter grade that corresponds to the numeric grade provided as a parameter.
(4) Implement the add method. The method will add the student's name to the list associated with the specified numeric grade.
(5) Implement the get method. The method will return the list with the names of students that have the specified letter grade.

Question 5: Write Java code for a Card class
(1) Use an enumerated type for the suits in a card deck (Spades, Hearts, Diamonds, Clubs)
(2) Implement the comparable interface for Card objects so that the suits are ranked in the order listed (Spades > Hearts > Diamonds > Clubs) Hint: Comparable interface requires implementation of "public int compareTo(Object o) {….}" method for Card objects so that the suits are ranked in the order listed. This method returns 1 if greater, 0 if equal, -1 if less than.

Question 6: Write Java code for a Deck class
(1) Uses an ArrayList to store multiple Card objects
(2) Use an anonymous inner class to generate an Iterator over Card objects in the Deck. Hint: Anonymous inner class should have "hasNext(), next() and remove()" methods.