KING SAUD UNIVERSITY
COLLEGE OF COMPUTER AND INFORMATION SCIENCES
COMPUTER SCIENCE DEPARTMENT

CSC 111: Introduction to Programming with Java	Final Lab Make-up	1st Semester 1441/1442

Name (Arabic):……………………………..…………… ID:……………………………....
Section#.. Serial#:………………………..
PC#:…………………………………………………………… Lab#...........................

INSTRUCTIONS:

1- Create a **folder** on the **desktop** with your full name **firstname_lastname**
2- The first three lines of your code should specify your **full name, your ID and your lab section or time** as comments.
3- Use the same variable names in the UML, and meaningful names for other variables.
4- Reuse code whenever possible.
5- The duration of the exam: **2 hour 45 minutes**
6- Lab instructor email:

We want to implement a program to manage the online exams of a language center . The program should be able to create an exam for the student, read the students answers and give the student final result

1. Create a class exam according to the following UML:

exam
- studentId : String - grade: int - level: String - key:char[] - answers: char[] - answeredQuestions:int **+ nOe: int**
+ exam (String s , char[] k) + readAnswers():int + gradeExam(): int **+** studentLevel():void +display(): void +getters/setters as necessary

Where:

- **studentId:** The id of the student.
- **grade:** student grade.
- **level:** The level of the student (Advanced – Moderate – Beginner).
- **key []:** an array that holds the key answers to the exam .
- **answers []**: An array that holds the student answers .
- **answeredQuestions:** total number of questions answered by the student
- **nOe:** total number of exams , each time an exam is created nOe is incremented by 1.

- **exam (String s , char [] k):** a non-default constructor to initialize attribute (studentId , key) to given values.
    - Exams have five questions.

- **readAnswers():int:** The method read the answers of the students , each index in the array represent a question and the value is the answer of that question.
    - Answers values are (A , B , C , D).
    - The exam answers are case sensitive.
    - If the user enter an invalid option it should be handled.
    - It should keep reading students answers until exam is complete or user enter **x**
    - Method return total number of answered questions

- **gradeExam(): double :** this method grads the student exam based on the key answers and return the total points for the student

- **studentLevel():void :** this method assign student level based on his/her grade

grade	**level**
0-1	Beginner
2-4	Moderate
5	Advanced

- **display():** displays the exam information as follow:
  studentId: XXXXX
  level :"XXXX"

2. Create an application class examCenter that contains the following:

a. Create an array *exams* of type exam , maximum size 50.

b. In the main:

- Add the following exams information :

Student id	key
1111	A , B , C , C , D
2222	B , B , D , A , A

- Read the answers for both exams
- Display number of answered questions for each student
- Display the information of the students who answered all the questions in the exam
- Assuming the array is filled , Calculate grade and level for each student
- Move every advanced student to new array *Astudent*

  note: size of the array should be suitable (you can't add any more students)

Sample Run:

```
first student
Reading answers for student 1111
Enter answer of Question 1
a
Enter answer of Question 2
C
Enter answer of Question 3
F
Invalid input
Enter answer of Question 3
D
Enter answer of Question 4
C
Enter answer of Question 5
x
finish reading answers
Total answered questions 4
Second student
Reading answers for student 2222
Enter answer of Question 1
b
Enter answer of Question 2
b
Enter answer of Question 3
D
Enter answer of Question 4
A
Enter answer of Question 5
A
Total answered questions 5
student who answered all exam :
studentId: 2222
level "Advanced"
Moved 1 student
```

Class Math	Class String
int/long/double/float **abs** (int/long/double/float x) long **round**(double x) double **ceil**(double x) double **floor**(double x) double **exp**(double x) double **log**(double x) double **log10**(double x) double **sqrt**(double x) double **pow**(double x, double y) int/long/double/float **min** (int/long/double/float x, int/long/double/float y) int/long/double/float **max** (int/long/double/float x, int/long/double/float y) double **sin**(double x) double **cos**(double x) double **tan**(double x)	int **length**() String **toLowerCase**() String **toUpperCase**() String **concat**(String str) String **replace** (char CharToBeReplaced, char CharReplacedWith) char **charAt**(int index) int **indexOf** (char ch) int **indexOf** (char ch, int pos) int **indexOf** (String str) int **indexOf** (String str, int pos) String **substring** (int beginIndex) String **substring** (int beginIndex, int endIndex) boolean **equals** (String str) int **compareTo** (String str)
	Class Character
	boolean **isDigit**(char ch) boolean **isLetter**(char ch) boolean **isLowerCase**(char ch) boolean **isUpperCase**(char ch) char **toLowerCase**(char ch) char **toUpperCase**(char ch)