National University of Computer and Emerging Sciences



Lab Exercise 08

For

Object Oriented Programming Lab

|  |  |
| --- | --- |
| Course Instructor(s) | Dr. Danish |
| Lab Instructor(s) | Mr. Mughees Ismail |
| Semester | Spring 2020 |

**FAST School of Computing**

# Instructions:

1. Make a word document with the naming convention “SECTION\_ LAB#\_ROLLNO” and put all your source code and snapshots of its output in it. Make sure your word file is formatted properly.
2. Plagiarism is strictly prohibited.
3. Do not discuss solutions with one another.

# Useful links

|  |
| --- |
| **Question#1** |

Create a student database of Students using a class. Your program will have two classes One for sections and One for Students. Class student will have the following **private** attributes:

1. Name
2. CNIC
3. Gender
4. CGPA
5. Count (that counts the numbers of students of a section)

Having required accessor functions.

And the class section will the following **private** attributes:

1. An array of Objects of class Students less than 40.
2. Section name
3. Class teacher.
4. Static Count (that count the number of sections created)

Having functions **editSection(), addStudent(), updateStudent(), const printList(), printCount().**

In main function, you’ll have a pointer of class section. You’ll ask the user for number of sections he wants to create, and then create as many objects as user wants.

Now, provide a menu to perform these functionalities:

1. Edit Section Attributes
2. Add Student in a section
3. Update Student of a section
4. Print List of Students of a section
5. Print List of Sections (can be done in main())
6. Print Count of Sections and Students in each section

You can access functions of class Section directly from main(). Only call the function of class Section. Delete all the memory at the end.

|  |
| --- |
| **Question#2** |

Write a class **testing** having following **private** attributes:

1. int **test\_id**
2. string **question**
3. **obj** of struct **timer** having { hour, minute, seconds }
4. **static** count (holds the number of tests created)

having following functions:

void setTestId (int test\_id);

void setQuestion (string question);

void setTimer (timer obj);

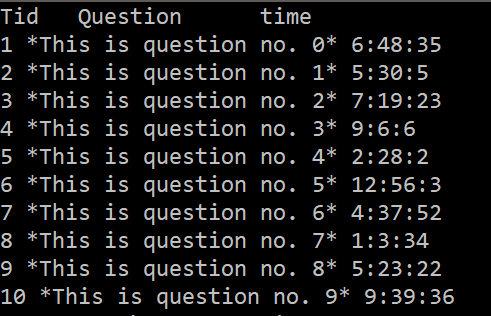
void display();

with a private constructor.

In the **main()** function, make an array of objects of user defined size by calling a static function and call the functions of class for every object. At the end, display the number of tests created.

Use the same naming convention mentioned above. Now use **this** pointer to make it easy for the compiler to understand which attribute you are referring to in the function code. Remembers test\_id is always unique.

**Sample Output:**



Call a function after this output that displays the **count** of tests.