# Programming II – Grading Application.

# Grading App

A high-level view of a Grading application is shown below. You will build a system by precisely following the specifications below. A sample output is provided at the end of this document. You must match the format of the output as close as possible.

Do not use the class diagram below or even the UML object diagram to develop your code. Please read the specification/description of each type including all the members before starting to implement the system.

It is recommended that you implement each type in the order that they are specified.



Warning:

You may not use ANY techniques that was not covered in COMP123 so far.

## The Person class 6 marks

The **Person** class is described in the diagram below.

|  |
| --- |
| **Person**  **class** |
| **Fields** |
|  |
| **Properties** |
| **+ «property» Name : string**  **+ «property» Cell : string** |
| **Methods** |
| **+ «constructor» Person()**  **+ «constructor» Person(  name : string,   cell : string)** |

### Description of class members

#### Fields:

There are no fields.

#### Properties:

There are two properties. Both have public getters and setters.

**1 Mark**

**Name** – this **string** property represents the name of this person.

**1 Mark**

**Cell** – this **string** property represents the cell number of this person.

#### Constructors:

There are two constructors.

**1 Mark**

Person() {} // constructor with no arguments

**2 Mark**

Person( **string name, string cell** )// to initialize the properties

#### Methods:

**1 Mark**

**public override string ToString()** – This method returns a string representing the object

## The Student class 14 marks

The **Student** class inherits from the **Person** class. It is described in the diagram below.

|  |
| --- |
| **Student**  **Class**  **Person** |
| **Fields** |
|  |
| **Properties** |
| **+ «property» CourseName : string**  **+ «property» Quizzes : double**  **+ «property» Assignments : double**  **+ «property» Exams : double**  **+ «property setters absent» Total : double** |
| **Methods** |
| **+ «constructor» Student()**  **+ «constructor» Student(  name : string,   cell : string,**  **courseName : string**  **quizzes : double,**  **Assignments: double,**  **exams : double)**  **+ ToString() : string** |

### Description of class members

#### Fields:

There are no fields.

#### Properties:

There are three properties with both public getter and setters.

**1 Mark**

**Course Name** – this **string** property represents the course name

**1 Mark**

**Quizzes** – this **double** property represents the score of the student in all quizzes.

**Assignments** – this **double** property represents the score of the student in assignments.

**1 Mark**

**1 Mark**

**Exams**– this **double** property represents score in the exams

**Total** – this **double** computed property represents the total marks or the score of the student in the course. (Total = Quizzes + Assignments + Exams)

**2 Mark**

#### Constructors:

There are two constructors.

**Student()** – This constructor in necessary for deserialization.

**1 Mark**

**Student (string name, string cell, String courseName, double quizzes, double assignments, double exams)**

**4 Marks**

– This constructor takes six arguments and does the following:

1. Call parent constructor to initialize Name and Cell
2. Use the remaining arguments to initialize the appropriate properties

#### Methods:

There one instance methods.

**3 Marks**

**public override string ToString()** – This method returns a string representing the object.(it should also invoke the inherited to string method )

## The GradeManager class 16 marks

The **Grade manager** class is described in the diagram below.

|  |
| --- |
| **GradeManager**  **static class** |
| **Fields** |
| **-$ students : List<Students>** |
| **Properties** |
|  |
| **Methods** |
| **+$ LoadStudents(string file) : void**  **-$ CreateStudents() : List<Student>**  **+$ Show() : void**  **+$ Above(double selectedMark)**  **: double**  **+$ Average() : void**  **+$ Top() : void** |

### Description of class members

All the members of this class are static

#### Fields:

There is a single private field.

**1 Mark**

**students** – this **List<Student>** field represents a collection of students objects

#### Properties:

There are no properties.

#### Constructor:

There are a no constructors.

#### Methods:

There are six public and one private static methods.

**5 Marks**

**public static void LoadStudents(string file)** –This method initializes the field **students** to a collection of student objects by reading the data from the file **students.txt**. each line int file represent student info (name, cell, courseName, quizzes, assignments exams ) If you are not able to do this, then you may call the **CreateStudents()** method to obtain a collection of students.

**private static List<Students> CreateStudents ()** –This private will not be used unless you are not able to get the above method to work. Copy the statements below to realize this method.

List<Student> result = new List<Student>(){

new Student("Sahasan","123-4567","COMP123", 8, 17, 60),

new Student ("Kassie", "234-5678", "COMP123", 10, 20, 65),

new Student ("Maiara", "345-6789","COMP123", 7, 18, 58),

new Student ("Laura", "456-7890", "COMP123", 9, 19, 40),

new Student ("Aaron", "678-9012", "COMP123", 7, 17, 50),

new Student ("Diego", "789-0123", "COMP123", 9, 19, 60),

new Student ("Jancyben", "890-1234", "COMP123", 9, 19, 45),

new Student ("Laila", "901-2345","COMP123", 9, 15, 55),

new Student ("Enas", "123-9012", "COMP123", 7, 17, 57),

new Student ("Mel", "234-0123","COMP123", 9, 16, 47),

new Student ("Deep", "345-1234","COMP123", 9, 19, 63),

new Student ("Palk", "456-8901", "COMP123", 8, 8, 25)

};

return result;

**2 Marks**

**public static void Show()** – This method will display all the objects in the students collection. See the output to get ideas on the implementation.

**2 Marks**

**public static void Above(double selectedMark)** all the objects in the students collection who’s score is above the **selectedMark**. See the output to get ideas on the implementation.

**public static double Average()** – This method will compute and return the average mark for list of students ( the average = sum of total for all student / number of students )

**3 Marks**

**public static void Top()** – This method will display the name of the student with highest score or Total

**3 Marks**

–

.

## Test Harness

Copy the following statements into your main method.

string file = "students.txt";

//Load list of students from specified json file

Console.WriteLine($"\n\nLoading list of students from file {file}");

GradeManager.LoadStudents(file);

Console.WriteLine("\n----------------------\n");

//Showing all the students

Console.WriteLine($"\n\nAll Students:");

GradeManager.Show();

Console.WriteLine("\n----------------------\n");

//Showing the Students with total greater than selected mark

int selectedMark = 50;

Console.WriteLine($"\nStudents with total mark above { selectedMark }:");

GradeManager.Above(selectedMark);

Console.WriteLine("\n----------------------\n");

//Showing the average score

Console.WriteLine($"\n\n Class average”);

GradeManager.Average();

Console.WriteLine("\n----------------------\n");

//Showing all the Students available on Tuesday @ 11:15

Console.WriteLine($"\n\ The Highest score Obtained by:");

GradeManagerr.Top();

## Program output

Loading students list from file students.txt

All Students:

Name: Sahasan Cell: 123-4567 CourseName: COMP123 Quizzes: 8 Assignments: 17 Exams: 60 Total: 85

Name: Kassie Cell: 234-5678 CourseName: COMP123 Quizzes: 10 Assignments: 20 Exams: 65 Total: 95

Name: Maiara Cell: 345-6789 CourseName: COMP123 Quizzes: 7 Assignments: 18 Exams: 58 Total: 83

Name: Laura Cell: 456-7890 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 40 Total: 68

Name: Aaron Cell: 678-9012 CourseName: COMP123 Quizzes: 7 Assignments: 17 Exams: 50 Total: 74

Name: Diego Cell: 789-0123 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 60 Total: 88

Name: Jancyben Cell: 890-1234 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 45 Total: 73

Name: Laila Cell: 901-2345 CourseName: COMP123 Quizzes: 9 Assignments: 15 Exams: 55 Total: 79

Name: Enas Cell: 123-9012 CourseName: COMP123 Quizzes: 7 Assignments: 17 Exams: 57 Total: 81

Name: Mel Cell: 234-0123 CourseName: COMP123 Quizzes: 9 Assignments: 16 Exams: 47 Total: 72

Name: Deep Cell: 345-1234 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 63 Total: 91

Name: Palk Cell: 456-8901 CourseName: COMP123 Quizzes: 8 Assignments: 8 Exams: 25 Total: 41

----------------------

Students with total mark above 50:

Name: Sahasan Cell: 123-4567 CourseName: COMP123 Quizzes: 8 Assignments: 17 Exams: 60 Total: 85

Name: Kassie Cell: 234-5678 CourseName: COMP123 Quizzes: 10 Assignments: 20 Exams: 65 Total: 95

Name: Maiara Cell: 345-6789 CourseName: COMP123 Quizzes: 7 Assignments: 18 Exams: 58 Total: 83

Name: Laura Cell: 456-7890 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 40 Total: 68

Name: Aaron Cell: 678-9012 CourseName: COMP123 Quizzes: 7 Assignments: 17 Exams: 50 Total: 74

Name: Diego Cell: 789-0123 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 60 Total: 88

Name: Jancyben Cell: 890-1234 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 45 Total: 73

Name: Laila Cell: 901-2345 CourseName: COMP123 Quizzes: 9 Assignments: 15 Exams: 55 Total: 79

Name: Enas Cell: 123-9012 CourseName: COMP123 Quizzes: 7 Assignments: 17 Exams: 57 Total: 81

Name: Mel Cell: 234-0123 CourseName: COMP123 Quizzes: 9 Assignments: 16 Exams: 47 Total: 72

Name: Deep Cell: 345-1234 CourseName: COMP123 Quizzes: 9 Assignments: 19 Exams: 63 Total: 91

----------------------

Class average :

77.5

----------------------

The Highest score Obtained by:

Name: Kassie, Marks: 95