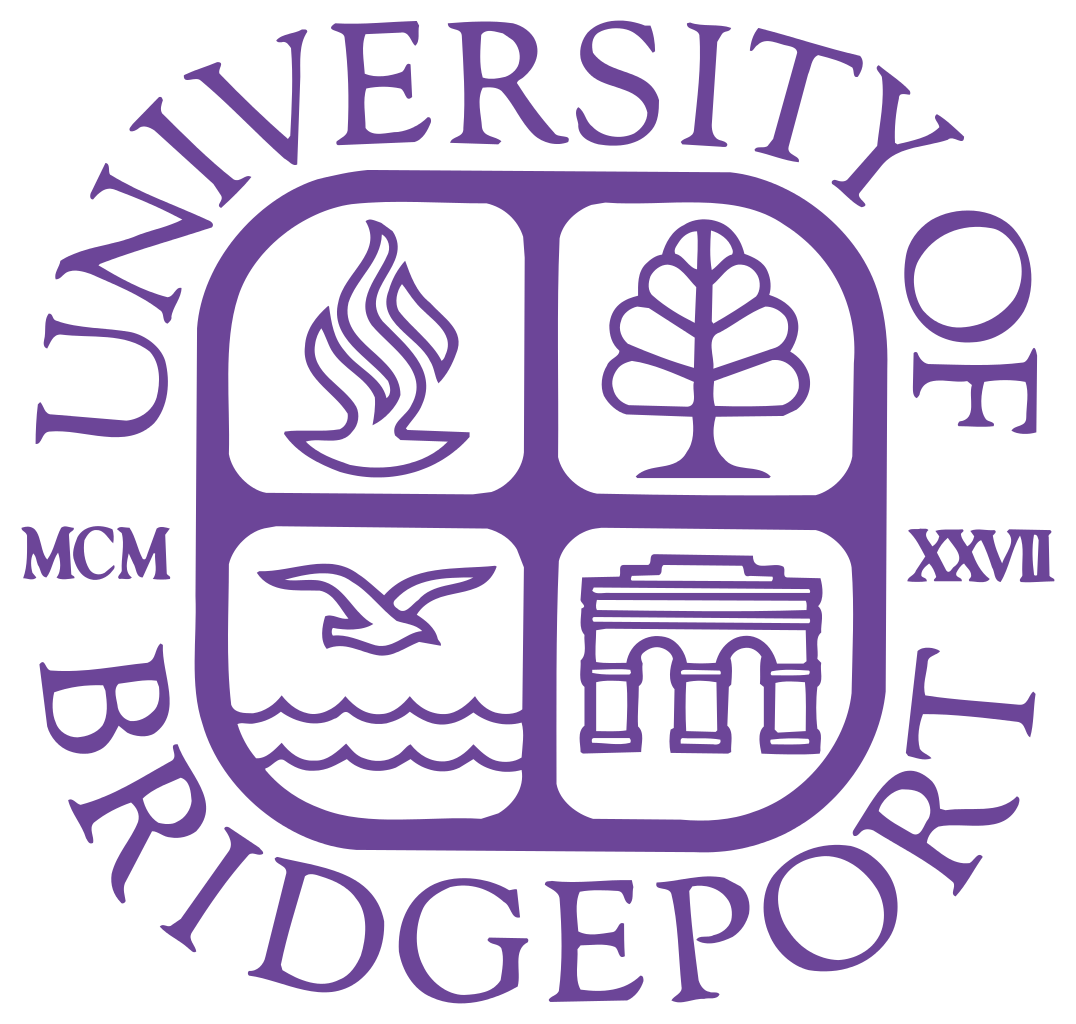
******

**Bashar Alhafni**

**Assignment #3**

**Objective:**

Creating Inheritance for a University Application. Suppose you are asked to create a university application where the following entities will be represented.

1. PartTimeStudent FirstName, LastName, SSNum, Test1, Test2, Street Address, City, Telephone ComputeGrade method with 0.4\*Test1 + 0.6 \* Test2

2. UnderGradStudent FirstName, LastName, ID, Test1, Test2, Street Address, City, Telephone ComputeGrade method with 0.45\*Test1 + 0.55 \* Test2

3. GradStudent FirstName, LastName, ID, Test1, Test2, Thesis, Street Address, City, Telephone ComputeGrade method with 0.45\*Test1 + 0.55 \* Test2

4. PhDStudent FirstName, LastName, ID, Test1, Test2, PhdAdvisor,Street Address, City, Telephone ComputeGrade method with 0.3\*Test1 + 0.7 \* Test2

5. LabManager FirstName, LastName, EmpID, Department, Street Address, City, Telephone

6. FullTimeFaculty FirstName, LastName, EmpID, Department, Rank, Street Address, City, Telephone (Rank could be Assistant, Associate or Full Professor)

7. PartTimeFaculty FirstName, LastName, SSNum, Department, Street Address, City, Telephone

8. Secretary FirstName, LastName, EmpD, Department, Street Address, City, Telephone

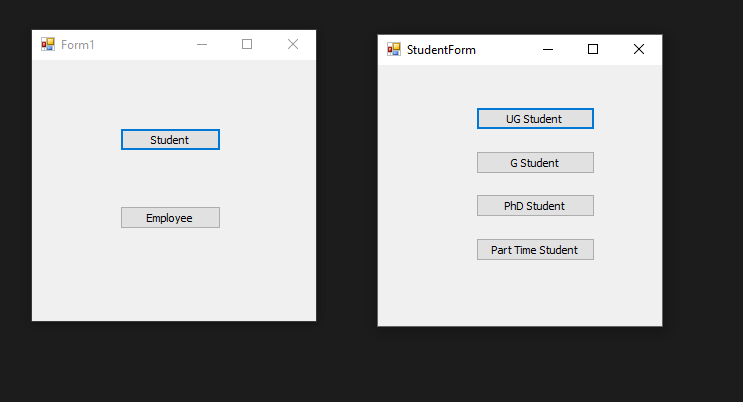
**Explanation:**

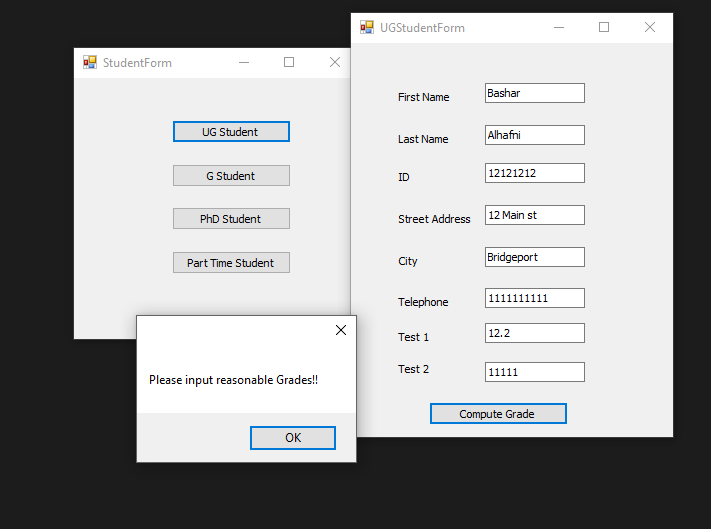
I created the abstract person that has FirstName, LastName, Street Address, City, and Telephone.

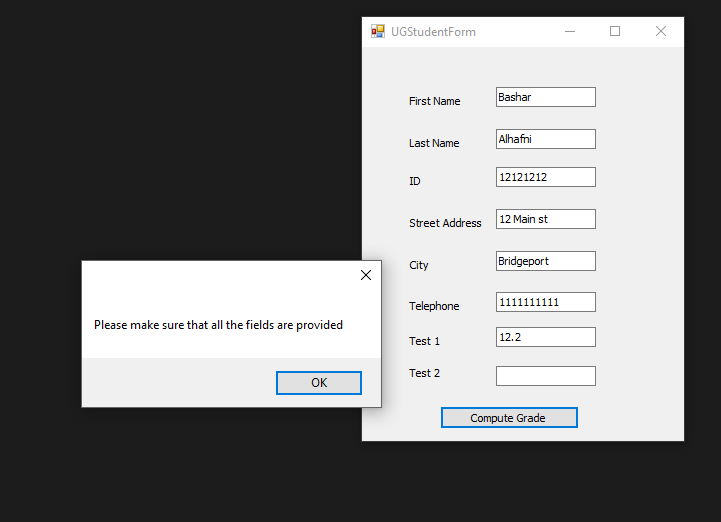
For the student part, I created a base abstract class student that is derived from the Person class, thus it has all the fields that the Person class has in addition to Test1 and Test2 as fields. It also has the abstract method ComputeGrade() to compute the grade of the student based on Test1 and Test2. There are 2 boolean methods CheckTelephone() to check if the phone number contains 10 digits or not, and CheckGrader() to check if the grade is between 0 and 100. The student class has 2 children classes, FullTimeStudent and PartTimeStudent. The FullTimeStudent is an abstract class that has the same fields as student in addition to the ID as a field. The PartTimeStudent class has the same fields as students in addition to the SSN as a field. The FullTimeStudent Class has UGStudent, GStudent and PhDStudent as derived classes. Those classes override the ComputeGrade() method differently.

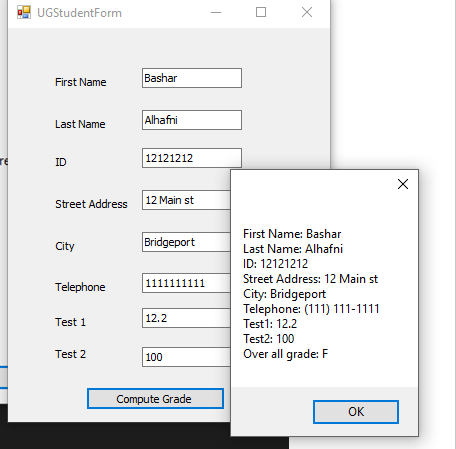
For the Employee part I also created a base abstract class called Employee which is derived from the Person class as well. It has the same fields as the Person Class in addition to the Employee Department. I also created a FullTimeEmployee and PartTimeFaculty. The FullTimeEmployee is an abstract class that has all the fields inside the Employee class in addition to the EmployeeID. For the PartTimeFaculty, it a class that has the same field as the Employee class in addition to the SSN. I also created the derived classes, FullTimeFaculty, LabManager and Secretary.

**Screenshots:**

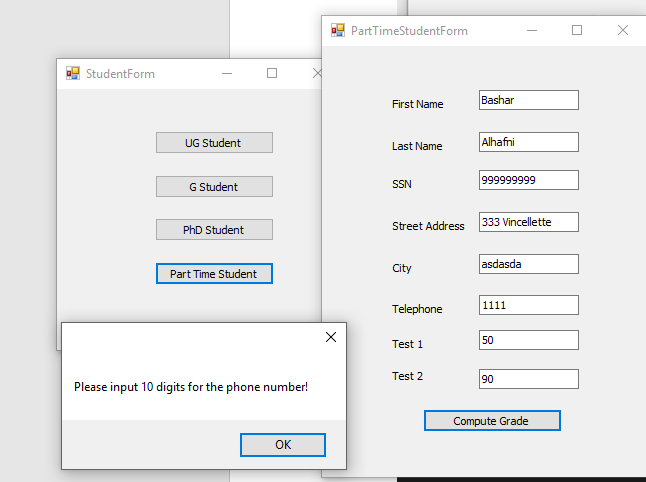


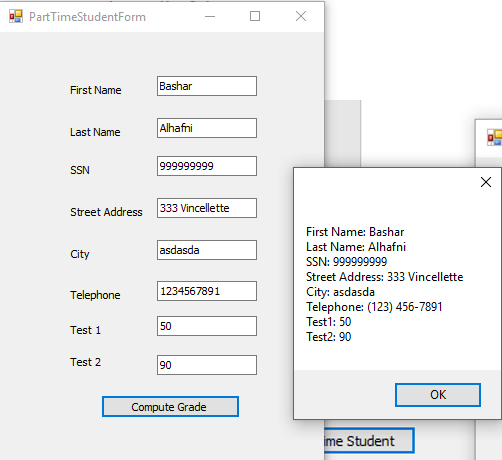


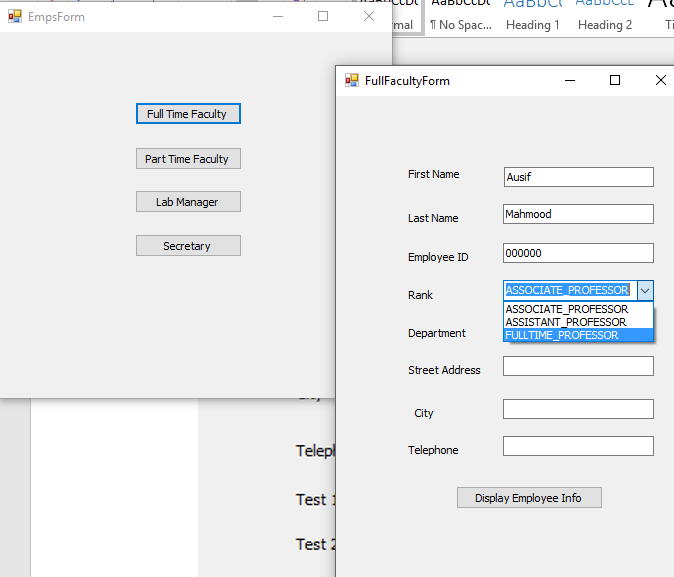


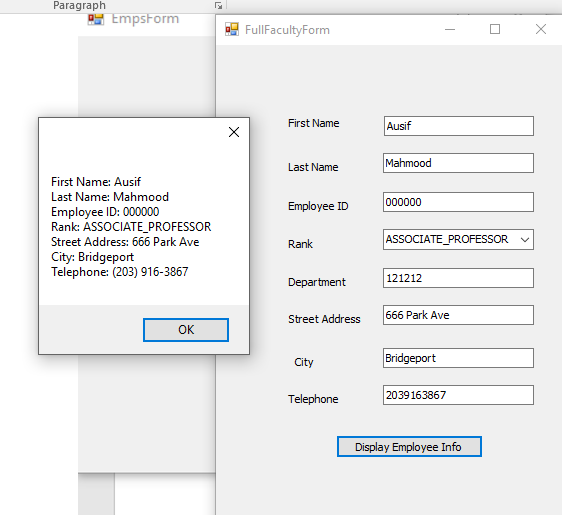


GStudent and PhDStudent forms are pretty much the same as the UGStudent form









And similarly LabManager, Secretary and PartTimeFaculty forms works the same way

**UML Diagrams:**

