**OOP: Lab-12  
Aggregation and Composition**

**Question 1:**

A class named **Processor** has

* Two attributes i.e. processName and price
* A parameterized constructor to initialize attributes with user-defined values

Class **MainMemory** consists of

* Two attributes i.e. size and price
* A parameterized constructor to initialize attributes with user-defined values

Class **MotherBoard** has

* a data member named compName of type string
* a no-argument constructor to initialize with default name intel

Design a class named **Computer** that includes

* A data member named **proc** of type Processor
* A data member named **ram** of type MainMemory
* A data member named **mboard** of type MotherBoard
* A parameterized constructor that accept two arguments of type Processor and MainMemory to initialize members of these types. Moreover, within this constructor, instantiate object of MotherBoard to initialize mboard data field.

Write main() in a way that it clearly describes aggregation and composition

relationships between objects of implemented classes.

**Question 2:**

A university owns various departments (e.g., CS, Electrical Engineering), and each department has number of professors. If the university closes, the departments will no longer exist, but the professors in those departments will continue to exist. Therefore, a University can be seen as a composition of departments, whereas departments have an aggregation of professors. In addition, a Professor could work in more than one department, but a department could not be part of more than one university.

Write a Class named **Professor** having following attributes:

• **name** of type string

• **employeeID** of type int

• **designation** of type string

Write a Class named **Department** having following attributes:

• **name** of type string

• **deptID** of type int

• Array **profList** of type Professor

• **noOfProfessors** of type int

Write a Class named **University** having following attributes:

• **name** of type string

• Array **dept** of type Department

• **numberOfDepartments** of type int

Write following functions.

1. Write appropriate getter setter of each data member for each Class.

2. Add/delete/update Department in University class **bool addDepartment(Department D)** **bool** **deleteDepartment(string name)** **bool updateDepartment(int id, string name)** //Update name of department given department id. **void Display()** function to display university information. Also display department information in this function.

3. Add/delete/update Professor in Department class **bool addProfessor(Professor p) bool deleteProfessor (int id) bool updateProfessor (int id, string newDesignation )** //Update designation of the professor given employee id. **void Display()** function to display department information. Also display professors information in this function.