The objective of this lab is to:

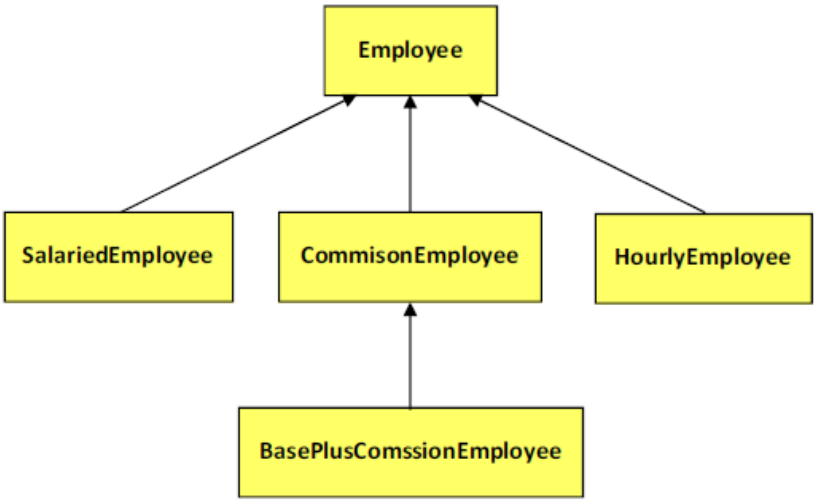
Understand and abstract classes, pure virtual functions, function templates.

Instructions!

1. Strictly follow good coding conventions (commenting, meaningful variable and functions names, properly indented and modular code.
2. Save your work frequently. Make a habit of pressing **CTRL+S** after every line of code you write.

Task 01: [20 Marks]

Implement the following class hierarchy, the inheritance access level should be public for all classes.



Employee Class:

- Declare three data members named *firstName*, *lastName* and *SSN* of type string with private access.
- Implement a parameterized constructor.
- Implement getter and setters.
- Implement a virtual function named print that displays the name and social security number of a particular employee.
- Implement a pure virtual function named earnings that calculates and return the earning of a particular employee.

SalariedEmployee Class:

- Declare a data member named weeklySalary of type double with private access.
- Implement a parameterized constructor, which initializes all the data members of SalariedEmployee with default parameter set to 0 for weeklySalary.
- Implement getter and setters.
- Implement a virtual function named print that displays the name, social security number and weekly salary of a particular employee.
- Implement a virtual function named earnings that return the earning of a particular salaried employee.

HourlyEmployee Class:

- Declare two data members named wage and hours of type double with private access.
- Implement a parameterized constructor, which initializes all the data members of HourlyEmployee with default parameter set to 0 for wage and hours.
- Implement getter and setters.

- Implement a virtual function named print that display the name, social security number, wage and hours of a particular employee.
- Implement a virtual function named earnings that calculates and return the earning of a particular hourly employee. The salary can by calculated by multiplying hours with wage

CommissionEmployee Class:

- Declare two data members named grossSales and commissionRate of type double with private access.
- Implement a parameterized constructor, which initializes all the data members of CommissionEmployee with default parameter set to 0 for grossSales and commissionRate.
- Implement get/set function for all data members.
- Implement a virtual function named print that display the name, social security number, gross sales and commission rate of a particular employee.
- Implement a virtual function named earnings that calculates and return the earning of a particular commissioned employee. The salary can by calculated by multiplying commission rate with gross sales.

BasePlusCommissionEmployee Class Details

- Declare data members named baseSalary of type double with private access.
- Implement a parameterized constructor, which initializes all the data members of BasePlusCommissionEmployee with default parameter set to 0 for baseSalary.
- Implement get/set function for all data members.
- Implement a virtual function named print that display the name, social security number, gross sales, commission rate and base salary of a particular employee.
- Implement a virtual function named earnings that calculates and return the earning of a particular base plus commissioned employee. The salary can by calculated as: CommissionEmploy::earnings+baseSalary.

In main program, create objects of each class created above with relevant information and display the personal information of each employee with their salaries. Then, create a pointer array of type Employee with size 4, each location of this array should point to object of SalariedEmployee, HourlyEmployee, CommissionEmployee and BasePlusCommsissionEmployee created above. Now loop through the entire pointer array and display the information of each employee using only two statements:

```
eptr[i] -> print();
eptr[i] -> earnings();
```

Where *eptr* is a pointer array of type Employee.

Task 02 [20 Marks]

1. Implement a function template *getMin( )* that takes two arguments of generic type and returns the minimum of them.
2. Implement a function template that takes two generic type 2D arrays and sizes of them. Return product of both the arrays. Assume that arrays square matrices.
3. Implement a function template that takes an linear array of generic type and display the content of this array on console.
4. Now, use the function templates you created in task1 and task 3 and pass them the Rectangle objects and array of Rectangles respectively and demonstrate its behaviour in *main( )* function.
   Note: For this task you may copy Rectangle class code from following path: \\printsrv\Faculty Data\Madiha\CMP-142 Object Oriented Proramming\Lecture Codes. You can make necessary changes to the Rectangle class to make it work for both the functions. i.e. you may overload required operators.