Object Oriented Programming Lab

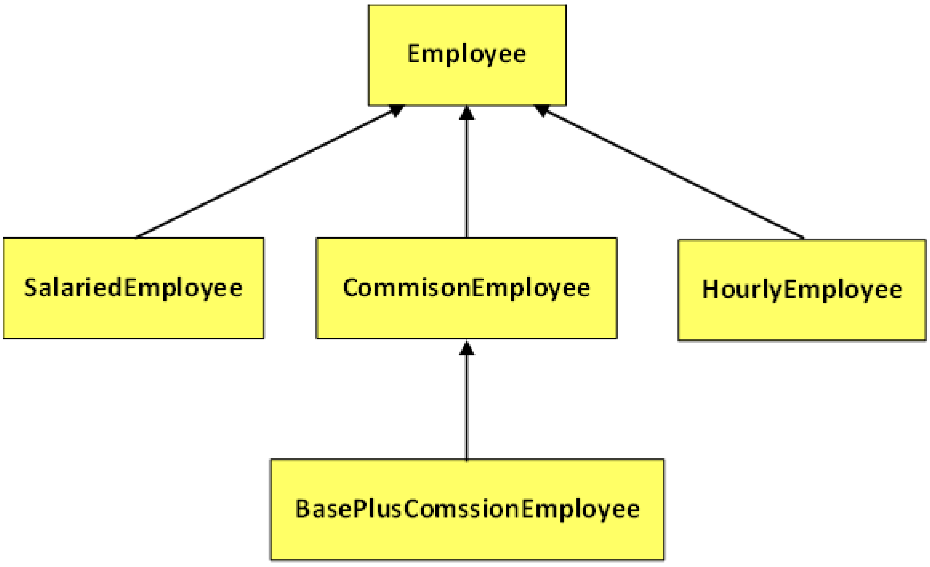
Lab 11 **Marks 10**

Instructions

- Work in this lab individually.
- You can use your books, notes, handouts etc. but you are not allowed to borrow anything from your peer student.
- Make sure to follow the best coding practices.
- Include comments to explain the logic where necessary.
- *You are strictly **NOT ALLOWED** to include any additional data-members/functions/constructors in your class.*
- Test your program thoroughly with various inputs to ensure proper functionality and error handling.
- Show your work to the instructor before leaving the lab to get some or full credit.

Employee Inheritance Hierarchy

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



Employee Class Details

- Declare three data members named **firstname**, **lastname**, and **SSN** of type string with private access.
- Implement a parameterized constructor.
- Implement get/set functions for all data members.
- Implement a virtual function named **print** that displays the name and social security number of an employee.
- Implement a pure virtual function named **earnings** that calculates and returns the earnings of an employee.

SalariedEmployee Class Details

- Declare a data member named **weeklySalary** of type double with private access.
- Implement a parameterized constructor that initializes all the data members of **SalariedEmployee**, with the default parameter set to 0 for **weeklySalary**.
- Implement get/set functions for all data members.
- Implement a virtual function named **print** that displays the name, social security number, and weekly salary of an employee.
- Implement a virtual function named **earnings** that returns the earnings of a salaried employee.

HourlyEmployee Class Details

- Declare two data members named **wage** and **hours** of type double with private access.
- Implement a parameterized constructor that initializes all the data members of **HourlyEmployee**, with the default parameters set to 0 for **wage** and **hours**.
- Implement get/set functions for all data members.
- Implement a virtual function named **print** that displays the name, social security number, wage, and hours of an employee.
- Implement a virtual function named **earnings** that calculates and returns the earnings of an hourly employee. The salary can be calculated by multiplying hours with the wage.

CommissionEmployee Class Details

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

BasePlusCommissionEmployee Class Details

- Declare a data member named **baseSalary** of type double with private access.
- Implement a parameterized constructor that initializes all the data members of **BasePlusCommissionEmployee**, with the default parameter set to 0 for **baseSalary**.
- Implement get/set functions for all data members.
- Implement a virtual function named **print** that displays the name, social security number, gross sales, commission rate, and base salary of an employee.
- Implement a virtual function named **earnings** that calculates and returns the earnings of a base-plus-commissioned employee. The salary can be calculated by adding the **CommissionEmployee::earnings** to the **baseSalary**.

Main Program Details

- Create objects of each class created above with relevant information and display the personal information of each employee with their salaries.

- Create a pointer array of type **Employee** with a size of 4. Each location of this array should point to an object of **SalariedEmployee**, **HourlyEmployee**, **CommissionEmployee**, and **BasePlusCommissionEmployee** created above.

- 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`.