**Take classes from Lab 10 task 1.**

Consider UML diagram to create classes. Write given DM's & MF's in Employee class. Keep DM's protected in Employee class. Inside body of ***calcSal*** write return salary. Create Parameterized constructor of Employee class to set all DM's without any check. Also overload stream insertion operator for Employee class. Implement Derived classes details are:

* **RegEmp** (Regular Employee) has a fixed salary**,** however if employee works for additional hours' employee will be paid at the rate of 100 per hour. Implement ***calcSal*** function of **RegEmp** accordingly.
* **ConEmp** (Contractual Employee) has a fixed salary and per hour rate. Salary is calculated by multiplying hours with hour rate and fixed salary is added.
* **DWageEmp** (Daily Wage Employee) has zero fixed salary, and each employee has hour rate accordingly to skills. Implement ***calcSal*** function of **DWageEmp** simply multiply hRate by hours.

Implement parameterized constructor in all child classes.

**Task 1:**

Create class Department, where department has title (string), no of possible employees (int), current no of employees (int) and double pointer of type Employee. Write following member functions in class Department:

* **Constructor** with two parameters (title, no of possible employees): Initialize data member title and no of possible employees. Set current number of employees to 0. Declare a pointer array of type Employee according to no of possible employees. Initialize pointer array with NULL value.
* **addEmployee** with appropriate parameters: Send parameter of employee + type of employee (1. for regular employee, 2. for contractual employee & 3. for daily wage employee. Add new employee according to the current no of employees.
* **modifyEmployee** with two parameters employee no (index here) and modified salary. Modify salary of employee according to employee no.
* **calcSalary** with one parameter employee no. Calculate and print salary of employee according to the type of employee
* **calcAllSalary** with no parameter. Calculate and print salary of all employees.
* **deleteEmployee** with single parameter employee no. Delete employee according to the employee no. Shift all employees ahead (index no greater than this employee) to fill the place and create an empty place at the higher index.

Write a driver (main function) program. Declare department object with ten employees. Later add 6 to 10 different type of employees and call functions of department class.

**Task 2:**

Create class Company with two departments. Company has data members for two departments and total no of employees plus company name. Create two departments production and finance. Production has maximum 20 employees and finance has maximum 10 employees. Write following member functions:

* **Constructor** with single parameter: Assign company name and create two departments and initialize total no of employees to zero.
* **addEmployee** with appropriate parameters. Send parameters required for employee, type of employee and department no (1 for production department and 2 for finance department). For each addition of employee update total number of employees
* Overload stream extraction operator to print all data about company and its departments.

Write a driver program to demonstrate class Compnay.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* HMMMMM! O O O o o o o o p p p o o o o o O O O \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*