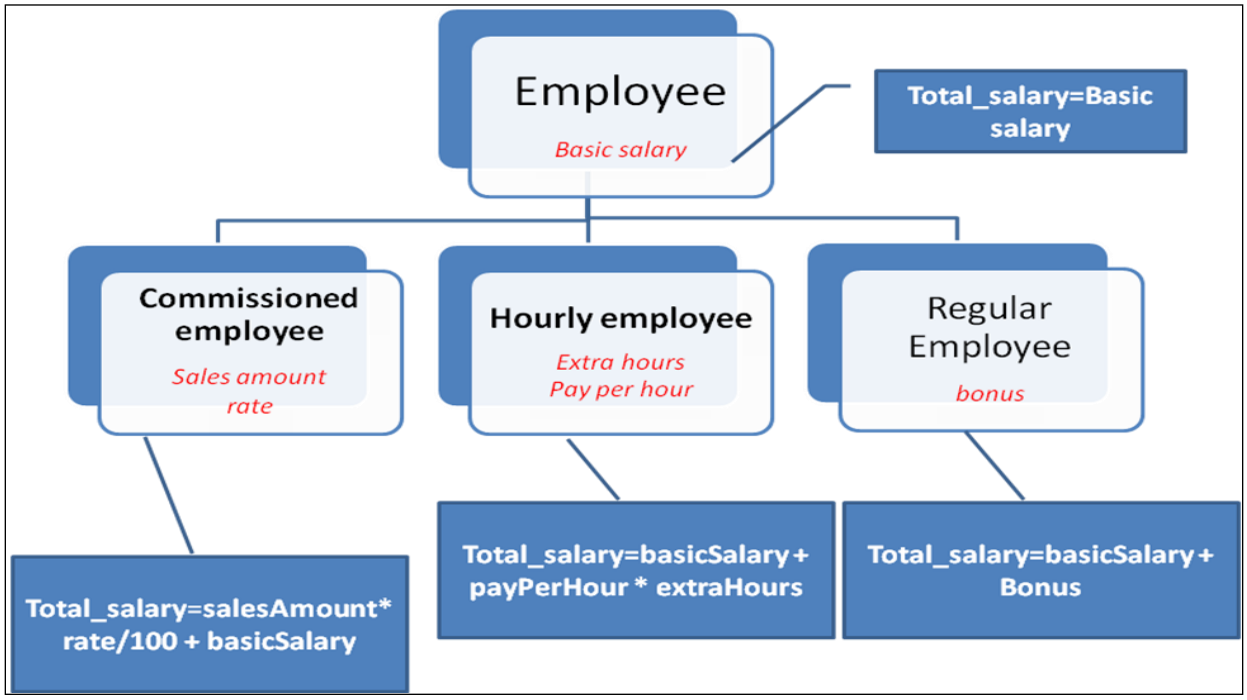
Exercise 1: (10 points)

We want to design a system for a company to calculate the salaries of its different types of employees.

Consider the following diagram:



Every employee has an employee ID and a basic salary. The Commissioned employee has a sales amount and rate. Hourly employee is paid on the basis of number of working hours. A regular employee may have a bonus.

You have to implement all the above classes. Write constructor for all classes. The main functionality is to calculate salary for each employee which is calculated as follows:

Commissioned Employee: Total Salary=sales amount*rate/100+basic salary
Hourly Employee: Total salary=basic salary + pay per hour*extra hours
Regular Employee: Total salary= basic salary + bonus

You have to define the following function in all classes:
float calculateSalary() and run the given **main()** for the following two cases:

1. when the *calculateSalary()* in base class is not **virtual**
2. when the *calculateSalary()* in base class is made **virtual**

Use the following main().

```
int  main()
{
    CommissionedEmployee  E1(25, 5000, 1000, 10);

    // CASE 1 - derived Class Pointer pointing to Derived class object

    CommissionedEmployee * ptr;
    ptr = &E1;
    cout<<" Commissioned Employee salary:"<<ptr->calculateSalary();
    cout<<endl;

    // CASE 2 - Base Class Pointer pointing to Derived class object

    Employee * eptr;
    eptr = &E1;
    cout<<" Commissioned Employee salary:"<<eptr->calculateSalary();
    cout<<endl;

    CommissionedEmployee E2 (25, 5000, 1000, 10);
    CommissionedEmployee E3 (26, 5000, 2000, 10);

    HourlyEmployee H1(27, 5000, 10, 100 );
    HourlyEmployee H2(28, 5000, 5, 100 );

    RegularEmployee R1(29, 5000, 1000 );
    RegularEmployee R2(29, 5000, 2000 );

    Employee * list [6];
    list[0] = & E2;
    list[1] = & E3;
    list[2] = & H1;
    list[3] = & H2;
    list[4] = & R1;
    list[5] = & R2;

    for(int i = 0 ; i < 6; i++)
    {
     cout<<"Employee "<<i<<" salary is : "<<list[i]>calculateSalary();
     cout<<endl;
    }

    return 0;
}
```