COMSATS University Islamabad, Lahore Campus
Department of Computer Science

Assignment 3 – SPRING 2024

Course Title:	Object Oriented Programming				Course Code:	CSC241	Credit Hours: 4(3,1)
Course Instructor/s:	Mr. Imran Latif				Program Name:	BSCS	
Semester:	2nd	Section:	A&B	Batch	FA23-BSCS		
Total Marks:	**10**	Obtained Marks:			Date:	April 29, 2024	
Student's Name:					Reg. No.		

Important Instruction:

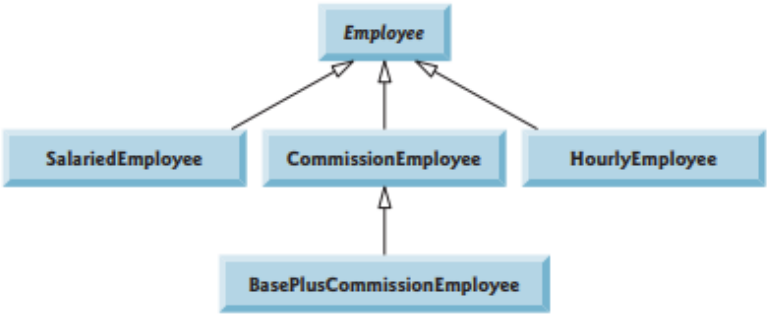
- Student is himself/herself responsible for successful submission of assignment on Microsoft teams.
- Your submission must include the following in a single pdf file(FA23-BCS-RollNo-Name).
  1. Code of all classes
  2. Snapshot of the output of submitted code.
- Copied assignment will get zero credit.
- Deadline: May 6, 2024till 11:30 PM

Bloom taxonomy level CLO3 <Applying>

Question 1:

Recall the concept of Inheritance, method overriding and polymorphism and write down the code according to requirements.
Bloom taxonomy level CLO2 <Applying>

A company pays its employees on a weekly basis. The employees are of four types: Salaried employees are paid a fixed weekly salary regardless of the number of hours worked, hourly employees are paid by the hour and receive overtime pay for all hours worked in excess of 40 hours, commission employees are paid a percentage of their sales and salaried- commission employees receive a base salary plus a percentage of their sales. For the current pay period, the company has decided to reward salaried-commission employees by adding 10% to their base salaries. The company wants to implement a Java application that performs its payroll calculations polymorphically.



	earnings	toString
Employee	abstract	*firstName lastName* social security number: *SSN*
Salaried-Employee	weeklySalary	salaried employee: *firstName lastName* social security number: *SSN* weekly salary: *weeklySalary*

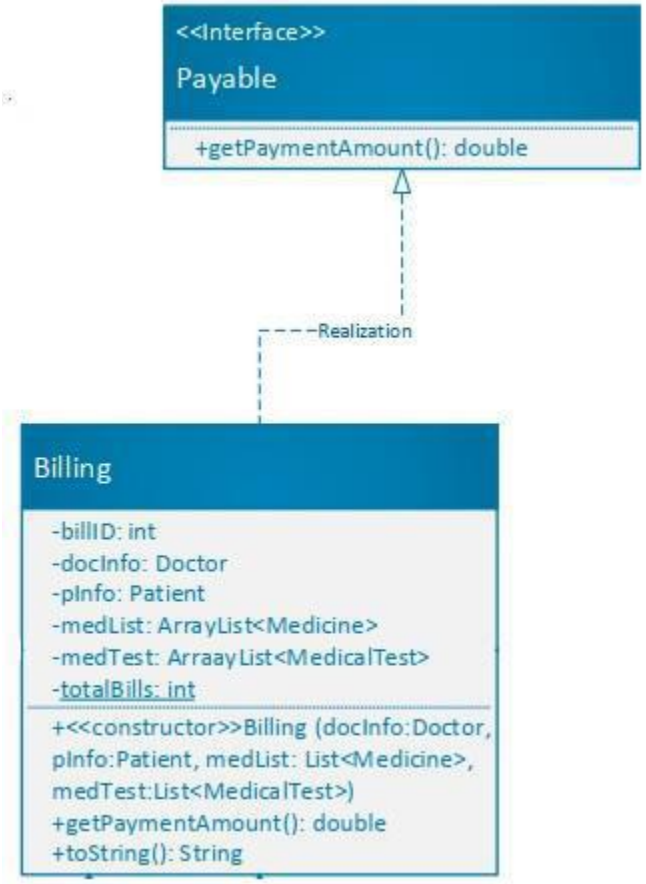
Hourly-Employee	if (hours <= 40)
	  wage * hours
	else if (hours > 40)
	{
	  40 * wage +
	  (hours - 40) *
	  wage * 1.5
	}
		hourly employee: *firstName lastName*
		social security number: *SSN*
		hourly wage: *wage*; hours worked: *hours*

Commission-Employee	commissionRate * grossSales	commission employee: *firstName lastName* social security number: *SSN* gross sales: *grossSales*; commission rate: *commissionRate*
BasePlus-Commission-Employee	(commissionRate * grossSales) + baseSalary	base salaried commission employee: *firstName lastName* social security number: *SSN* gross sales: *grossSales*; commission rate: *commissionRate*; base salary: *baseSalary*

- Modify the above payroll system to include private instance variable birthDate in class Employee. Add *get* methods to class Date. Assume that payroll is processed once per month.
- Include an additional Employee subclass PieceWorker that represents an employee whose pay is based on the number of pieces of merchandise produced. Class PieceWorker should contain private instance variables wage (to store the employee's wage per piece) and pieces (to store the number of pieces produced). Provide a concrete implementation of method earnings in class PieceWorker that calculates the employee's earnings by multiplying the number of pieces produced by the wage per piece. Create an array of Employee variables to store references to the various employee objects. In a loop, for each Employee, display its String representation and calculate the payroll for each Employee (polymorphically), and add a $100.00 bonus to the person's payroll amount if the current month is the one in which the Employee's birthday occurs or an employee object is of type BasePlusCommissionEmployee

Question 2:

Recall the concept of polymorphism and interfaces and write down the code according to requirements.
Bloom taxonomy level CLO3 <Applying>



Consider the above scenario, where Billing class has composition relationship with Doctor having private instance variables (docName, docID and docFee) and a public getDocID() method, Patient having private instance variables (pName, pID, pDisease), Medicine having private instance variables (medID, medName, medQty, medPrice), and MedicalTest having private instance variables (testID, testName, testPrice). In addition, each class has its toString method to display its object state. The getPaymentAmount() method of Billing class returns the total billing amount that includes doc fee, medicine cost and medical test fee. Suppose another method getDoc() is added to Billing class that returns Doctor.
Given the above info, you are required to do the following in the driver class:
a) The program should create an arraylist of Payable that holds three Billing objects. First Billing object comprised of two medicines and a medicalTest in addition to doctor and patient objects. The other two billing objects constitute three medicines and two medicalTest in addition to doctor and patient objects.

b) Ask the user to guess the billing amount. Traverse the arraylist using enhanced for. Print the billing details of those bills having total billing amount greater or equal to the billing amount given by the user.

c) Create an empty array **(not arrayLists)** of Doctor of size 2. Now, traverse again the Payable arraylist using enhanced for, but this time assign the doctor to the array having docID of 2.