Practice Exercise on Interfaces & Abstract Classes

I. Write set of classes to represent University students for the following categories. Write abstract classes to represent abstraction wherever necessary.

1. B.Tech CSE Students
2. B.Tech ECE Students
3. MBA Students
4. MHRM Students

All the above categories of students shall have the following methods. The following methods should work and present the data of the respective Student level classes (ex:CSEStudent, ECEStudent, MBAStudent, and MHRMStudent)

A. void displayProfile()
b. double calculateSemesterFee()

All B.Tech Students (both CSE and ECE students) shall have the members (rollno, name, SemesterScores(SemNo,SubjectCode,grade)) and the following methods. These Semester Subjects will be different for CSE and ECE students.

A. byte getEngDrawingMarks()
B. byte getEngMathsMarks()

Write CSEStudent class with the member called specialization-[ex:AI,Data science,CyberSecurity] and the following method

A. byte getProgrammingScore()

Write ECEStudent class with the member called IndustrialVisitInfo and the following method

A. byte getIndustrialVisitScore()

All Management Students (both MBA and MHRM students) shall have the members(rollno, name, AnnaualMarks(year,SubjectCode,marks)) following methods.These Annual Subjects will be different for MBA and MHRM students.

A. byte getOrgBehaviourMarks()
B. byte getRiskManagementMarks()

Write MBAStudent class with the member called InternshipInfo and the following method

A. byte getInternshipScore()

Write MHRMStudent class with the member called CaseStudyTitle and the following method

A. byte getCaseStudyScore()

Write a StudentPerformance class with a method performance(..) that accepts any student class (ex:CSEStudent,ECEStudent,MBAStudent,MHRMStudent) object and display their profile, semester fee and their respective scores and marks. And then write a class StudentDemo with main() method by calling StudentPerformance class method performance(..) by passing different student objects.

II. Write a Worker class to represent a worker concept and then write two sub classes DailyWorker and SalariedWorker from it. Every worker has a name and a salary rate.

Write method ComputePay(int hours) to compute the week pay of every worker. A DailyWorker is paid on the basis of number of days she/he work. The SalariedWorker gets paid the wage for 40 hours a week no matter what actual hours is.

Write a PayRoll class with a method calculateMonthlyPay(Worker w) that accept either SalariesWorker or DailyWorker to calculate the pay of workers. Finally write a class with a main method to demonstrate the PayRoll class method calculateMonthlyPay(Worker w) usage for different type of employees.

III. Write an interface PriceCalculator with the method: double calculatePrice()

Write an abstract Product class with the members (prodId,name,price,gst) and the following methods

A. double calculateShippingCharges(int pincode) Concrete method with logic-If the pincode is from 100001 to 299999 shipping charges are 5% of the product and 7% in case if the pincode ranges from 300001 to 899999.

B. void displayProfile() -an abstract method to display product profile. Override this method in the sub classes to display the product details including all the possible details.

Write the following 3 classes with the implementation of above interface PriceCalculator

A. Write a class ElectronicProduct with a member class ProductSpec(members:specName,description) array and implement the calculatePrice() method. Electronic goods are surcharged 3% on the product price.

B. AgriculturalProduct with a members ProductDescription and maxUsagePeriod and implement the calculatePrice() method. Agricultural goods are discounted 2% on their gst price.

C. ImportedProduct with a member importDuty and implement the calculatePrice() method. Imported products are shall have extra importyDuty percentage on their original price.

Write a Shopping class with a method shop that accepts any of the above 3 product objects and display the details of the product and also the price(product price + shipping charges).

Finally write a class with a main method to demonstrate the Shopping class method shop(..) usage for different type of products.