CSE1007- Java Programming

Inheritance

1. A training centre conducts a total of 7 tests for its students. Students are allowed to skip few tests. Let there be 25 students in the batch. So in the main class for every student, read the number of tests taken and the marks scored in each test. A class 'TestDetails' should be defined with a 2D array of float type to store the marks of all the students. Define a method 'storeMarks()' that will receive the following details for every student from the main class and create in the 2D array, those many columns equal to the number of tests, so as to store the marks. There is no need to store the number of tests. Define another method 'displayMarks()' to print the details.

   Also the training centre wishes to keep those students in notice period who have taken < 3 tests and those who have not scored ≥ 50 in at least 3 tests. Derive another class 'NoticePeriod' from 'TestDetails' that includes a method to count and print the number of students in bench. Also it should print the ID of those students assuming the row index of the array to be their ID. While checking do not proceed to check the marks in all tests, if the student has already scored more than 50 in 3 tests. Instantiate this class from the main class and do the required processing.

2. Create an inheritance hierarchy in java using following information given below that a bank might use to represent customers' bank accounts.

Base class **Account** should include one data member of type double to represent account balance. The class should provide constructor that receives an initial balance and uses it to initialize the data member.The constructor should validate the initial balance to ensure that it is greater than or equal to 0.If not the balance is set to 0.0 and the constructor should display an error message,indicating that the initial balance was invalid. The class also provides three member functions **credit, debit**(debit amount should not exceed the account balance) and **enquiry**.Derived class **SavingsAccount** should inherit the functionality of an Account, but also include data member of type double indicating the interest rate assigned to the Account. SavingsAccount constructor should receive the initial balance, as well as an initial value for SavingsAccount's interest rate. SavingsAccount should provide public member function **calculateInterest** that returns double indicating the amount of interest earned by an account.The method **calculateInterest** should determine this amount by multiplying the interest rate by the account balance.SavingsAccount function should inherit member functions credit,debit and enquiry without redefining them. Derived class **CheckingAccount** should inherit the functionality of an Account, but also include data member of type double that represents the fee charged per transaction. CheckingAccount constructor should receive the initial balance, as well as parameter indicating fee amount. class CheckingAccount should redefine credit and debit function so that they subtract the fee from account balance whenever either transaction is performed. CheckingAccount's debit function should charge a fee only if the money is actually withdrawn (debit amount should not exceed the account balance).After defining the class hierarchy, write program that creates object of each class and tests their member functions. Add interest to SavingAccount object by first invoking its calculateInterest function, then passing the returned interest amount to object's credit function.

3. The interface GCD contains an abstract method *computeGCD(int num1, int num2)*. Class APPROACH1 implements the interface by following Euclid's algorithm and class APPROACH2 implements the interface by listing all the factors (need not be prime factors) of the two numbers and choosing the highest common factor. Write a Java program to do the above said operations.