1. Write a java program which initialization earning of an employee. The program should calculate the income tax to be paid by the employee as per the criteria given below:Slab rate IT rate

Upto Rs. 50,000 Nil

Upto Rs. 60,000 10% on additional amount

Upto Rs. 1,50,000 20% on additional amount

Above Rs. 1,50,000 30% on additional amount

Hint: - Run: - java calculates 1,25,000

Result: - income tax is …………………………….

public class IncomeTaxCalculator {

public static void main(String[] args) {

double salary = 125000;

System.out.println("Income Tax for a salary of Rs. " + salary + " is: Rs. " + calculateIncomeTax(salary));

}

private static double calculateIncomeTax(double salary) {

double tax = 0;

if (salary > 150000) {

tax += 0.3 \* (salary - 150000);

}

if (salary > 60000) {

tax += 0.2 \* (salary - 60000);

}

if (salary > 50000) {

tax += 0.1 \* (salary - 50000);

}

return tax;

}

}

Output:

Income Tax for a salary of Rs. 125000.0 is: Rs. 20500.0

2. Design a class for a bank database the database should support the following operations.

1. Deposit a certain amount into an account,

2. Withdrawing a certain amount from an account,

3. Return a value specifying the amount (i.e. balance) in an amount.

public class BankAccount {

private double balance;

// Constructor for BankAccount with initial balance

public BankAccount(double initialBalance) {

if (initialBalance > 0.0)

balance = initialBalance;

}

// Deposit method that takes an amount and adds it to the balance

public void deposit(double amount) {

if (amount > 0.0)

balance += amount;

else

System.out.println("Invalid deposit amount");

}

// Withdraw method that takes an amount and subtracts it from the balance

// if there's enough balance, otherwise it prints an error message

public void withdraw(double amount) {

if (amount > 0.0 && amount <= balance)

balance -= amount;

else

System.out.println("Insufficient balance or invalid withdrawal amount");

}

// Method to get the balance

public double getBalance() {

return balance;

}

// Main method to demonstrate the BankAccount class functionality

public static void main(String[] args) {

BankAccount account = new BankAccount(100.0);

account.deposit(50.0);

System.out.println("Balance after deposit: " + account.getBalance());

account.withdraw(25.0);

System.out.println("Balance after withdrawal: " + account.getBalance());

}

}

Output:

Balance after deposit: 150.0

Balance after withdrawal: 125.0

3. Define a “Clock” class that does the following: -

a. Accept hours, minutes and seconds.

b. Check the validity numbers.

c. Set the time to AM/PM mode.

Use necessary constructors and methods to do the above task.

public class Clock {

private int hours;

private int minutes;

private int seconds;

private boolean isAM;

// Constructor with hours, minutes, and seconds

public Clock(int hours, int minutes, int seconds) {

setTime(hours, minutes, seconds);

}

// Constructor with hours and minutes

public Clock(int hours, int minutes) {

this(hours, minutes, 0);

}

// Constructor with hours only

public Clock(int hours) {

this(hours, 0, 0);

}

// Constructor with no arguments, initializes to 00:00:00 AM

public Clock() {

this(0, 0, 0);

}

// Method to set the time

public void setTime(int hours, int minutes, int seconds) {

if (hours < 0 || hours > 12) {

throw new IllegalArgumentException("Invalid hours");

}

if (minutes < 0 || minutes > 59) {

throw new IllegalArgumentException("Invalid minutes");

}

if (seconds < 0 || seconds > 59) {

throw new IllegalArgumentException("Invalid seconds");

}

this.hours = hours;

this.minutes = minutes;

this.seconds = seconds;

this.isAM = hours < 12;

}

// Method to get the time

public String getTime() {

return String.format("%02d:%02d:%02d %s", hours, minutes, seconds, isAM ? "AM" : "PM");

}

// Method to set the AM/PM mode

public void setAM(boolean isAM) {

this.isAM = isAM;

if (hours == 12) {

hours = 0;

}

}

// Method to get the AM/PM mode

public boolean isAM()

Output: