Name: Somnath R. Shintre Roll No:

Class: TE CSE Batch:

**Title: -** Develop a BankAccount class which should contain all methods of Bank i.e. balanceEnquery(), withdraw(), transfer() and deposit(). You should create at least two objects of BankAccount using array and do all operations mentioned above. Also generate user defined exception LowBalanceException, NegetiveNumberException and PasswordMismatchException whenever required. To transfer amount from one account to another use two BankAccount objects.

**Program: -**

public interface Bank {

    public static final int permanentpassword = 9359;

    public abstract double balanceEnquery();

    public abstract double withdraw(double withdrawAmount) throws LowBalanceException, NegetiveNumberException;

    public abstract String transfer(int password, double transferAmount, BankAccount to)

            throws LowBalanceException, NegetiveNumberException, PasswordMismatchException;

    public abstract void deposit(double depositAmount) throws NegetiveNumberException;

}

public class BankAccount implements Bank {

    private double mainBalance;

    @Override

    public double balanceEnquery() {

        return mainBalance;

    }

    @Override

    public double withdraw(double withdrawAmount) throws LowBalanceException, NegetiveNumberException {

        if (withdrawAmount <= 0) {

            throw new NegetiveNumberException("\tInvalid amount = " + withdrawAmount);

        } else if (mainBalance < withdrawAmount) {

            throw new LowBalanceException("\tYou have insufficent balance! \n\tYour current balance is " + mainBalance);

        }

        mainBalance = mainBalance - withdrawAmount;

        return mainBalance;

    }

    @Override

    public String transfer(int password, double transferAmount, BankAccount to)

            throws LowBalanceException, NegetiveNumberException, PasswordMismatchException {

        if (password != Bank.permanentpassword) {

            throw new PasswordMismatchException("\tIncorrect Password! Try Again.");

        }

        if (transferAmount <= 0) {

            throw new NegetiveNumberException("\tInvalid amount = " + transferAmount);

        } else if (this.mainBalance < transferAmount) {

            throw new LowBalanceException("\tYou have insufficent balance! \n\tYour current balance is " + mainBalance);

        }

        this.mainBalance = this.mainBalance - transferAmount;

        to.mainBalance = to.mainBalance + transferAmount;

        return "\tAmmount Transfered";

    }

    @Override

    public void deposit(double depositAmount) throws NegetiveNumberException {

        if (depositAmount <= 0) {

            throw new NegetiveNumberException("\tInvalid amount = " + depositAmount);

        }

        mainBalance = mainBalance + depositAmount;

    }

}

public class LowBalanceException extends Exception {

    private static final long serialVersionUID = 1L;

    public LowBalanceException() {

        super();

    }

    public LowBalanceException(String msg) {

        super(msg);

    }

}

public class NegetiveNumberException extends Exception {

    private static final long serialVersionUID = 1L;

    public NegetiveNumberException() {

        super();

    }

    public NegetiveNumberException(String msg) {

        super(msg);

    }

}

public class PasswordMismatchException extends Exception {

    private static final long serialVersionUID = 1L;

    public PasswordMismatchException() {

        super();

    }

    public PasswordMismatchException(String msg) {

        super(msg);

    }

}

import java.util.Scanner;

public class Federal\_Bank {

    public static void main(String[] args) {

        System.out.println("\n\t--------------------------------------------------");

        System.out.println("\t\t \*\*\*\*\*\*\*\* FEDERAL BANK \*\*\*\*\*\*\*\*\*");

        System.out.println("\t--------------------------------------------------");

        try {

            System.out.print("\tEnter Total Number Of Accounts To be Created: ");

            Scanner sc = new Scanner(System.in);

            int numOfAcc = sc.nextInt();

            BankAccount[] acc = new BankAccount[numOfAcc];

            int chAcc = 1, accNum, oCh, oE = 1, transferto, password;

            double withdrawAmount, transferAmount, depositAmount;

            String msg;

            for (int i = 0; i < numOfAcc; i++) {

                acc[i] = new BankAccount();

            }

            do {

                try {

                    System.out.print("\n\tEnter Your Account Number[1 to " + numOfAcc + "]: ");

                    accNum = sc.nextInt();

                    if (accNum < 0 && accNum > numOfAcc) {

                        throw new ArrayIndexOutOfBoundsException();

                    }

                    do {

                        try {

                            System.out.println("\n\tOptions: ");

                            System.out.println("\t1.Balance Enquery\t2.Withdraw");

                            System.out.println("\t3.Transfer       \t4.Deposite");

                            System.out.print("\n\tEnter choise: ");

                            oCh = sc.nextInt();

                            switch (oCh) {

                                case 1:

                                    System.out.println("\tCurrent Balance: " + acc[accNum - 1].balanceEnquery());

                                    break;

                                case 2:

                                    System.out.print("\n\tEnter Withdraw Amount = ");

                                    withdrawAmount = sc.nextDouble();

                                    withdrawAmount = acc[accNum - 1].withdraw(withdrawAmount);

                                    System.out.println("\tWithdraw Amount = " + withdrawAmount);

                                    System.out.println("\tCurrent Balance: " + acc[accNum - 1].balanceEnquery());

                                    break;

                                case 3:

                                    System.out.print("\n\tAccounts: ");

                                    for (int i = 0; i < numOfAcc; i++) {

                                        if (i == (accNum - 1))

                                            continue;

                                        System.out.print((i + 1) + " ");

                                    }

                                    System.out.print("\n\tChoose Account Number: ");

                                    transferto = sc.nextInt();

                                    System.out.print("\tEnter Amount to Transfered: ");

                                    transferAmount = sc.nextDouble();

                                    System.out.print("\n\tEnter Password: ");

                                    password = sc.nextInt();

                                    msg = acc[accNum - 1].transfer(password, transferAmount, acc[transferto - 1]);

                                    System.out.println(msg);

                                    break;

                                case 4:

                                    System.out.print("\n\tEnter Deposit Amount = ");

                                    depositAmount = sc.nextDouble();

                                    acc[accNum - 1].deposit(depositAmount);

                                    System.out.println("\tDeposited Amount = " + depositAmount);

                                    System.out.println("\tCurrent Balance: " + acc[accNum - 1].balanceEnquery());

                                    break;

                                default:

                                    System.out.println("\tPlease enter valid option...");

                            }

                            System.out.println("\n\tWant To Continue?");

                            System.out.print("\t[Press 1 to Continue or 0 to Exit] : ");

                            oE = sc.nextInt();

                        } catch (NegetiveNumberException nne) {

                            System.out.println(nne.getMessage());

                        } catch (LowBalanceException lbe) {

                            System.out.println(lbe.getMessage());

                        } catch (PasswordMismatchException pme) {

                            System.out.println(pme.getMessage());

                        }

                    } while (oE == 1);

                    System.out.println("\n\tWant To Change Account? ");

                System.out.print("\t[Press 1 to Continue or 0 to Exit] : ");

                    chAcc = sc.nextInt();

                } catch (ArrayIndexOutOfBoundsException aiobe) {

                    System.out.println("\tAccount Number Does Not Exits");

                }

            } while (chAcc == 1);

            sc.close();

        } catch (Exception e) {

            e.printStackTrace();

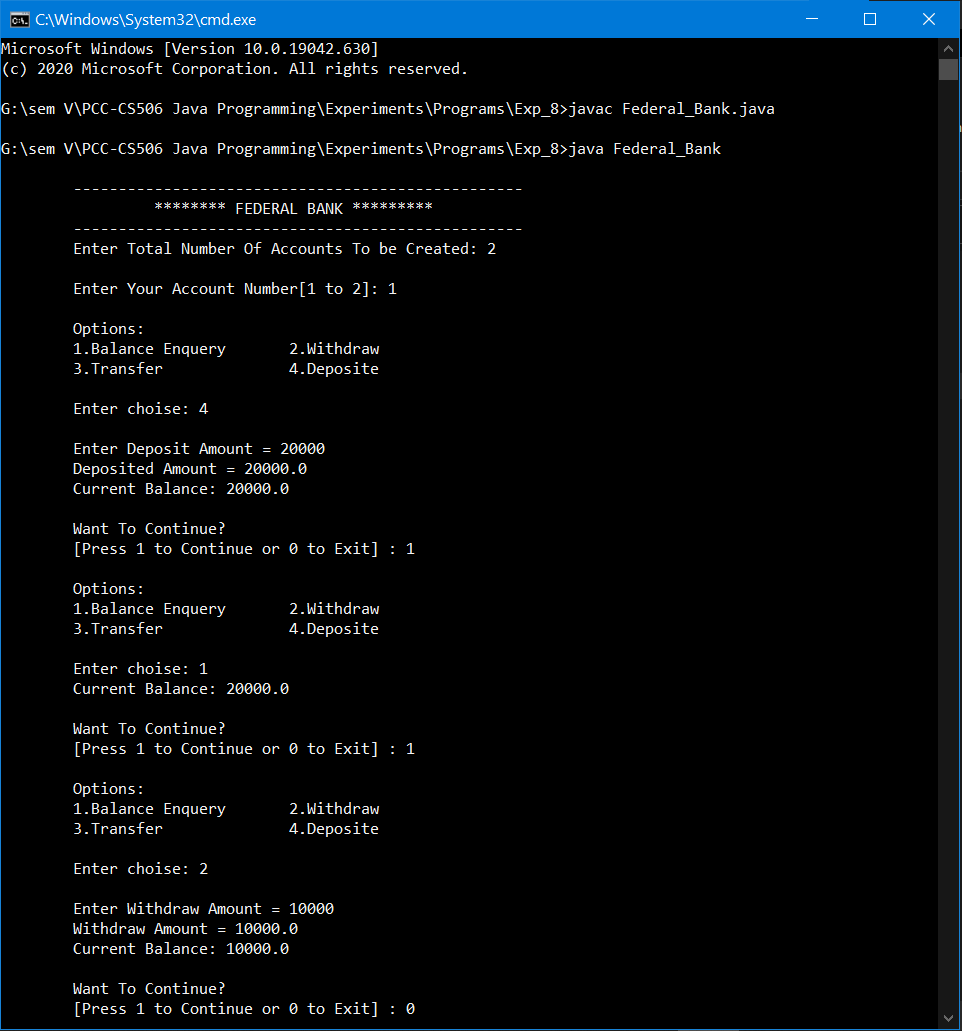
        }

    }

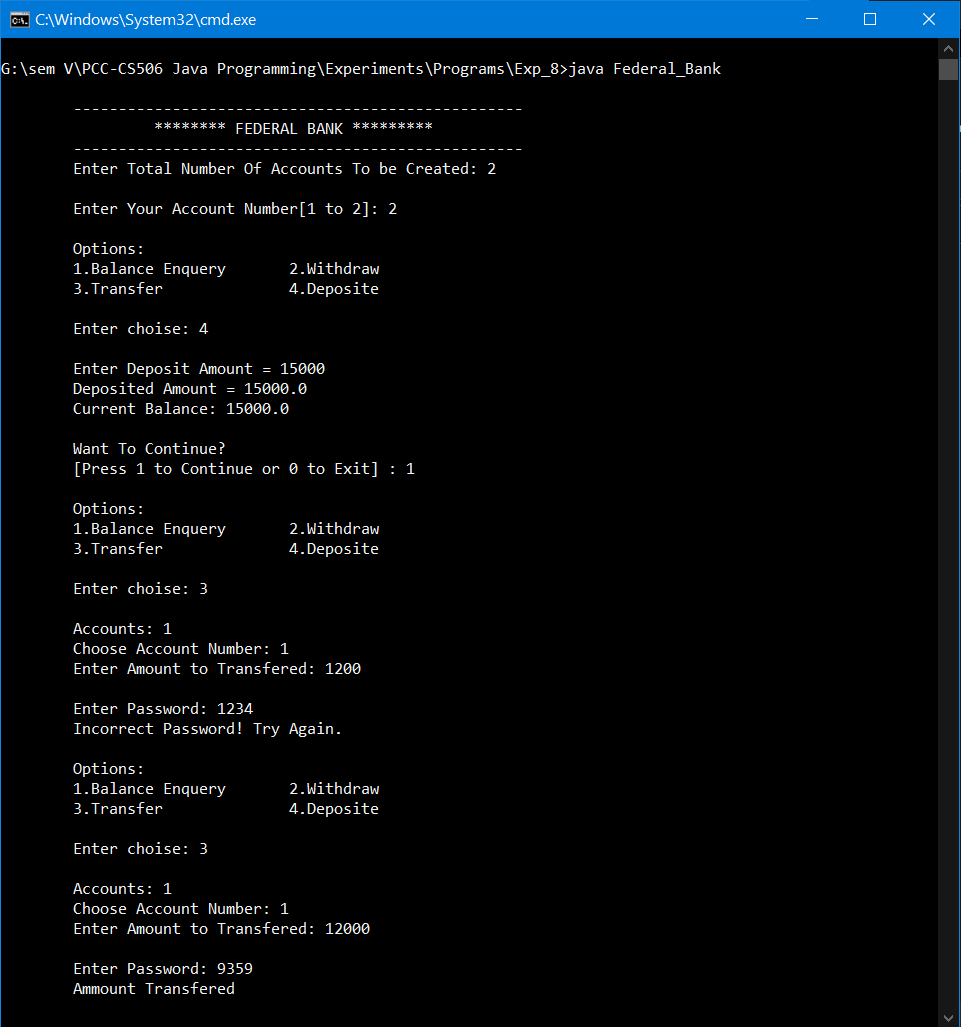
}

**Output: -**

**1. Create required bank account you want and select what process you want to do………**



**2. Deposit and Transfer the amount and password mismatch exception**



**3. Negative number exception and Low balance exception**

