1.EXECUTE IN BANK TRANSACTIONS SUCH AS VIEW BALANCE, DEPOSIT, WITHDRAWAL AND MONEY TRANSFER,ETC..... BY USING SINGLE LEVEL INHERITANCE. NOTE: INPUTS: USER INPUTS SHOULD BE RECIEVED IN BASE CLASS. PROCESS SHOULD BE DONE IN THE DERIVED CLASS. RESULT SHOULD BE DONE IN MAIN CLASS.

import java.util.Scanner;

class Bank {

int tot=10000;

int wh=0,dep=0,trans=0;

int sender=112233;

String ch;

int[] receiveraccs={12345, 12456, 13456, 167856, 19898};

Scanner sc=new Scanner(System.in);

public void input() {

System.out.println("INITIAL BALANCE : "+tot);

System.out.print("Enter your choice (dep/wh/trans/ex) : ");

ch = sc.nextLine();

if (ch.equals("wh")) {

System.out.print("Enter the withdrawal amount : ");

wh = sc.nextInt();

sc.nextLine();

if (wh > tot) {

System.out.println("INSUFFICIENT BALANCE !!");

wh = 0;

}

} else if (ch.equals("dep")) {

System.out.print("Enter the deposit amount : ");

dep = sc.nextInt();

sc.nextLine();

} else if (ch.equals("trans")) {

System.out.print("Enter the transfer amount : ");

trans = sc.nextInt();

sc.nextLine();

if (trans > tot) {

System.out.println("INSUFFICIENT BALANCE !!");

trans = 0;

} else {

System.out.print("Enter the receiver account number: ");

int receiverAccNo=sc.nextInt();

sc.nextLine();

if (!isValidAccount(receiverAccNo)) {

System.out.println("INVALID RECEIVER ACCOUNT NUMBER !!");

trans = 0;

}

}

} else if (!ch.equals("ex")) {

System.out.println("Please enter 'dep', 'wh', 'trans' or 'ex'.");

}

}

boolean isValidAccount(int accNo) {

for (int acc : receiveraccs) {

if (acc == accNo) {

return true;

}

}

return false;

}

}

class Transaction extends Bank {

public void process() {

if (ch.equals("wh")) {

if (wh > 0) {

tot = tot - wh;

System.out.println("WITHDRAWAL SUCCESS!!");

System.out.println("BALANCE AFTER WITHDRAWAL : " + tot);

}

} else if (ch.equals("dep")) {

tot = tot + dep;

System.out.println("DEPOSIT SUCCESS!!");

System.out.println("BALANCE AFTER DEPOSIT : " + tot);

} else if (ch.equals("trans")) {

if (trans > 0) {

tot = tot - trans;

System.out.println("TRANSFER SUCCESS!!");

System.out.println("BALANCE AFTER TRANSFER : " + tot);

}

}

}

}

public class Main {

public static void main(String[] args) {

Transaction t = new Transaction();

while (true) {

t.input();

if (t.ch.equals("ex")) {

System.out.println("Exiting...");

break;

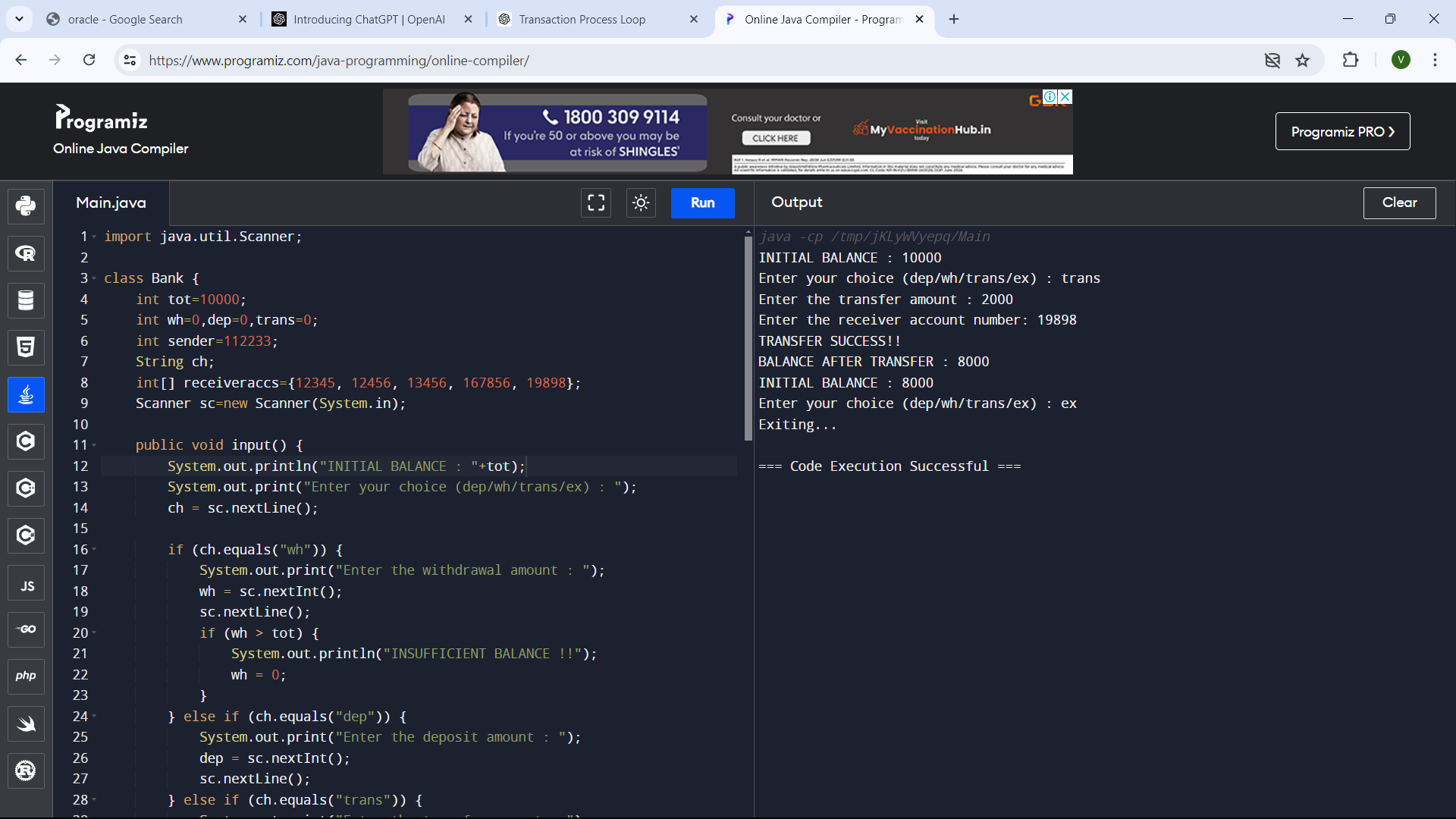
}

t.process();

}

}

}



2.CREATE A ACCOUNT WITH A MINIMUM BALANCE OF 2000 IN THE BASE CLASS.

IN ONE OF THE DERIVED CLASS, TRY TO TRANSFER THE AMOUNT 5000. TO YOUR FRIEND. IN ANOTHER DERIVED CLASS TRY TO WITHDRAW THE AMOUNT BY MAKING YOUR ACCOUNT BALANCE AS ZERO. USE MULTILEVEL INHERITENCE.

import java.util.Scanner;

class Bank {

protected int balance;

protected int minBalance = 2000;

public Bank() {

balance = 10000;

}

public void deposit(int amount) {

balance += amount;

System.out.println("DEPOSIT SUCCESS!! Deposited: " + amount);

System.out.println("BALANCE AFTER DEPOSIT : " + balance);

}

public void withdraw(int amount) {

if (balance - amount < minBalance) {

System.out.println("INSUFFICIENT BALANCE TO WITHDRAW !! Minimum balance should be " + minBalance);

} else {

balance -= amount;

System.out.println("WITHDRAWAL SUCCESS!! Withdrawn: " + amount);

System.out.println("BALANCE AFTER WITHDRAWAL : " + balance);

}

}

}

class Transfer extends Bank {

public void transferAmountToFriend(int amount) {

if (balance - amount < minBalance) {

System.out.println("INSUFFICIENT BALANCE TO TRANSFER !! Minimum balance should be " + minBalance);

} else {

balance -= amount;

System.out.println("TRANSFER TO FRIEND SUCCESS!!");

System.out.println("TRANSFERRED AMOUNT : " + amount);

System.out.println("BALANCE AFTER TRANSFER : " + balance);

}

}

}

class Withdrawal extends Transfer {

public void withdrawAll() {

int amountToWithdraw = balance - minBalance;

balance = minBalance;

System.out.println("WITHDRAWAL SUCCESS!! Withdrawn all available balance");

System.out.println("BALANCE AFTER WITHDRAWAL : " + balance);

}

}

public class Main {

public static void main(String[] args) {

Withdrawal account = new Withdrawal();

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("Current Balance: " + account.balance);

System.out.println("Choose action: ");

System.out.println("1. Withdraw all balance");

System.out.println("2. Transfer amount to friend");

System.out.println("3. Deposit amount");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

switch (choice) {

case 1:

account.withdrawAll();

break;

case 2:

System.out.print("Enter transfer amount: ");

int transferAmount = scanner.nextInt();

account.transferAmountToFriend(transferAmount);

break;

case 3:

System.out.print("Enter deposit amount: ");

int depositAmount = scanner.nextInt();

account.deposit(depositAmount);

break;

case 4:

System.out.println("Exiting...");

scanner.close();

return;

default:

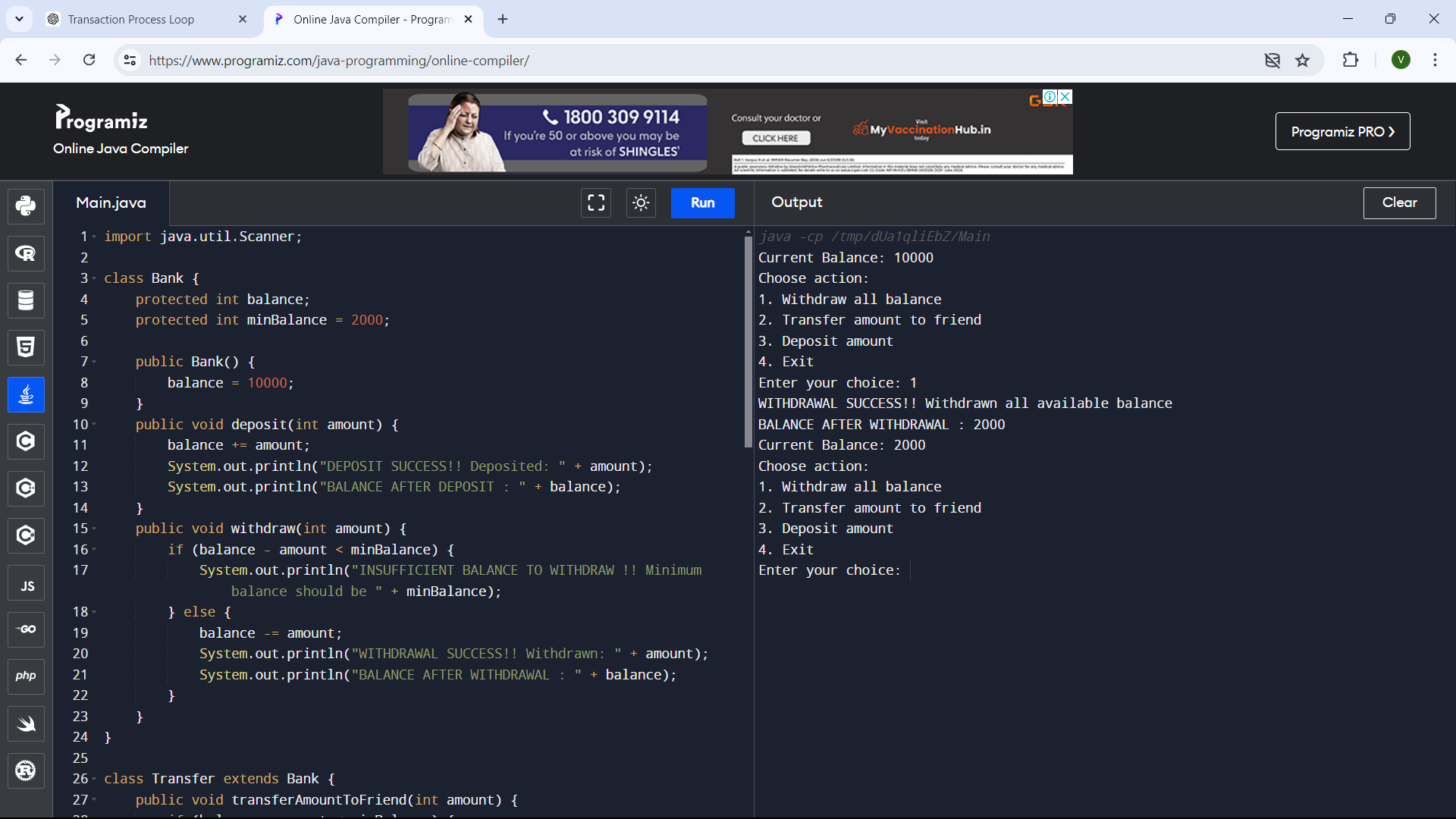
System.out.println("Invalid choice. Please enter again.");

}

}

}

}



3.ASSUME THAT YOU ARE GOING TO START A BANK WITH, FOUR BRANCHES. BRANCH 1: CREATE ACCOUNT. BRANCH2: DEPOSIT. BRANCH 3:WITHDRAWAL, BRANCH 4 : TRANFER.

import java.util.Arrays;

import java.util.Scanner;

class Bank {

int inputDeposit() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the Deposit amount: ");

int da = sc.nextInt();

return da;

}

int inputWithdraw() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the Withdraw amount: ");

int wa = sc.nextInt();

return wa;

}

int inputTransfer() {

Scanner sc = new Scanner(System.in);

System.out.print("Enter the Transfer amount: ");

int ta = sc.nextInt();

return ta;

}

}

class Transaction extends Bank {

int deposit(int ta, int da) {

return ta + da;

}

int withdraw(int ta, int wa) {

return ta - wa;

}

int transfer(int ta, int tra) {

return ta - tra;

}

}

class Deposit extends Transaction {

void performDeposit(int total) {

int depAmou = inputDeposit();

total = deposit(total, depAmou);

System.out.println("Before Deposit: " + (total - depAmou));

System.out.println("After Deposit: " + total);

}

}

class Withdraw extends Transaction {

void performWithdraw(int total) {

int witAmou = inputWithdraw();

if (witAmou > total) {

System.out.println("Insufficient amount");

} else {

total = withdraw(total, witAmou);

System.out.println("Before Withdraw: " + (total + witAmou));

System.out.println("After Withdraw: " + total);

}

}

}

class Transfer extends Transaction {

void performTransfer(int total, int[] accountNumbers) {

Scanner sc = new Scanner(System.in);

System.out.print("Enter account number to transfer to: ");

int accNum = sc.nextInt();

if (Arrays.binarySearch(accountNumbers, accNum) >= 0) {

int traAmou = inputTransfer();

if (traAmou > total) {

System.out.println("Insufficient amount");

} else {

total = transfer(total, traAmou);

System.out.println("Before Transfer: " + (total + traAmou));

System.out.println("After Transfer: " + total);

}

} else {

System.out.println("Invalid account number");

}

}

}

class ShowBalance extends Transaction {

void displayBalance(int total) {

System.out.println("\nBank balance: " + total);

}

}

public class Main {

public static void main(String[] args) {

int total = 1000;

int[] accountNumbers = {1001, 1002, 1011, 1012};

Arrays.sort(accountNumbers);

Scanner sc = new Scanner(System.in);

Deposit deposit = new Deposit();

Withdraw withdraw = new Withdraw();

Transfer transfer = new Transfer();

ShowBalance showBalance = new ShowBalance();

int choice;

while (true) {

showBalance.displayBalance(total);

System.out.println("Enter choice: Deposit = '1', Withdraw = '2', Transfer = '3', Exit = '4'");

choice = sc.nextInt();

if (choice == 1) {

deposit.performDeposit(total);

} else if (choice == 2) {

withdraw.performWithdraw(total);

} else if (choice == 3) {

transfer.performTransfer(total, accountNumbers);

} else if (choice == 4) {

System.out.println("Exiting");

break;

} else {

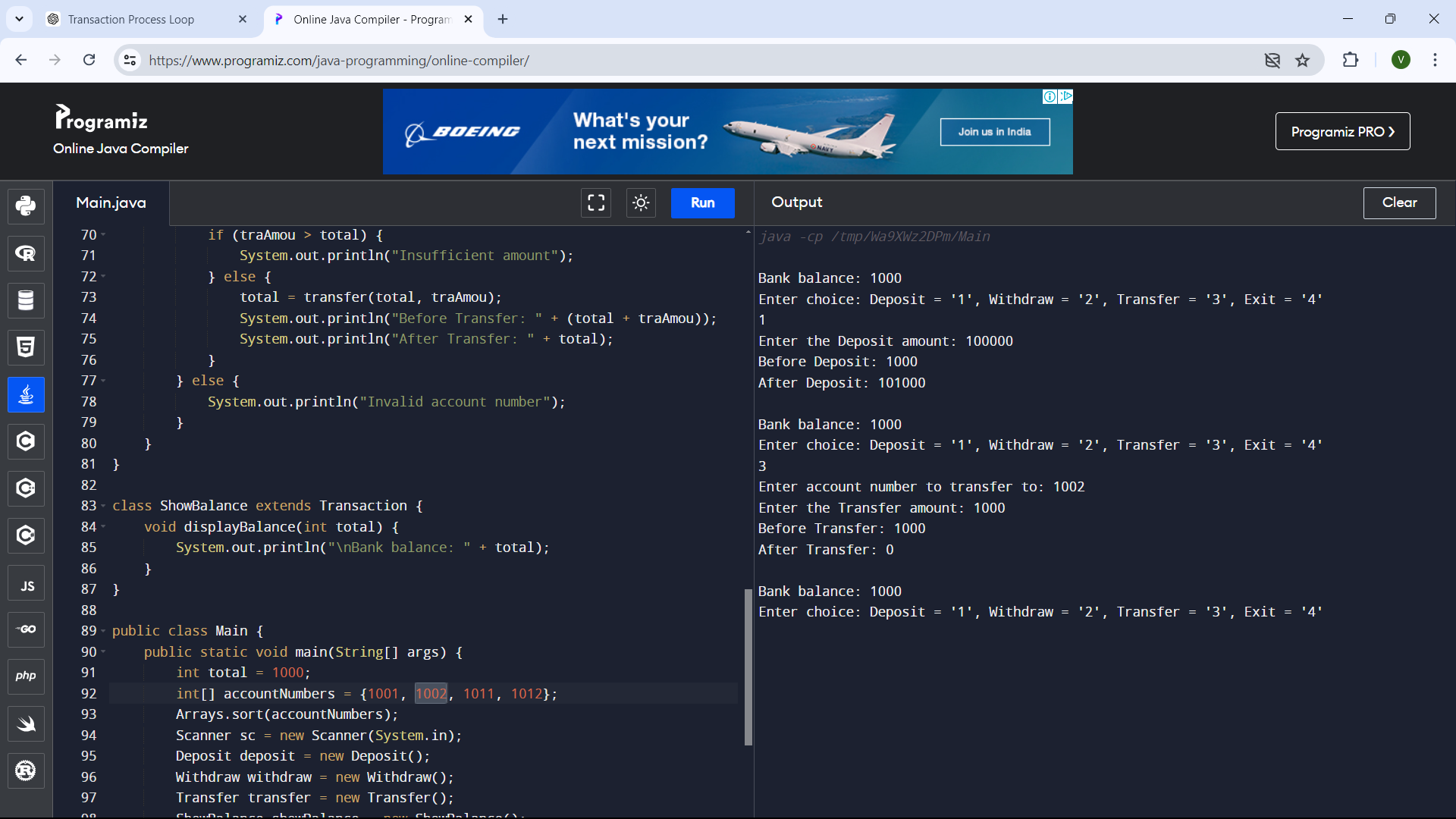
System.out.println("Invalid choice");

}

}

}

}



4.CONSIDER A STUDENT WHO COMPLETED 10TH STANDARD AND COMPLETED 12TH. DISPLAY THE MARKS, AND STU DETAILS.(NAME, REG NO, TOTAL AVG. ) 12 IN ONE BASE CLASS AND 10TH IN ANOTHER BASE CLASS. USE INTERFACE.

interface TenthStandard {

void display10thDetails();

}

interface TwelfthStandard {

void display12thDetails();

}

class Student {

String name;

int regNo;

Student(String name, int regNo) {

this.name = name;

this.regNo = regNo;

}

}

class TenthStandardStudent extends Student implements TenthStandard {

int[] marks;

TenthStandardStudent(String name, int regNo, int[] marks) {

super(name, regNo);

this.marks = marks;

}

public void display10thDetails() {

System.out.println("Name: " + name);

System.out.println("Registration No: " + regNo);

System.out.println("10th Standard Marks:");

for (int i = 0; i < marks.length; i++) {

System.out.println("Subject " + (i+1) + ": " + marks[i]);

}

}

}

class TwelfthStandardStudent extends Student implements TwelfthStandard {

int[] marks;

TwelfthStandardStudent(String name, int regNo, int[] marks) {

super(name, regNo);

this.marks = marks;

}

public void display12thDetails() {

System.out.println("Name: " + name);

System.out.println("Registration No: " + regNo);

System.out.println("12th Standard Marks:");

for (int i = 0; i < marks.length; i++) {

System.out.println("Subject " + (i+1) + ": " + marks[i]);

}

}

}

public class Main {

public static void main(String[] args) {

TenthStandardStudent student10th = new TenthStandardStudent("vishnu.u", 1001, new int[]{85, 90, 92, 88, 87});

TwelfthStandardStudent student12th = new TwelfthStandardStudent("vishnu.u", 2001, new int[]{78, 85, 80, 82, 88});

student10th.display10thDetails();

System.out.println();

student12th.display12thDetails();

}

}

