# OOPS USING JAVA MOODLE EXERCISES

# WEEK 1

# 

import java.util.Scanner;

public class Odd{

public static void main(String[] args){

Scanner scanner= new Scanner(System.in);

int number=scanner.nextInt();

int result=isEven(number)?1:2;

System.out.println(result);

scanner.close();

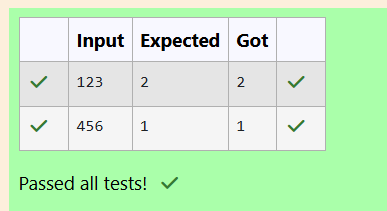
}

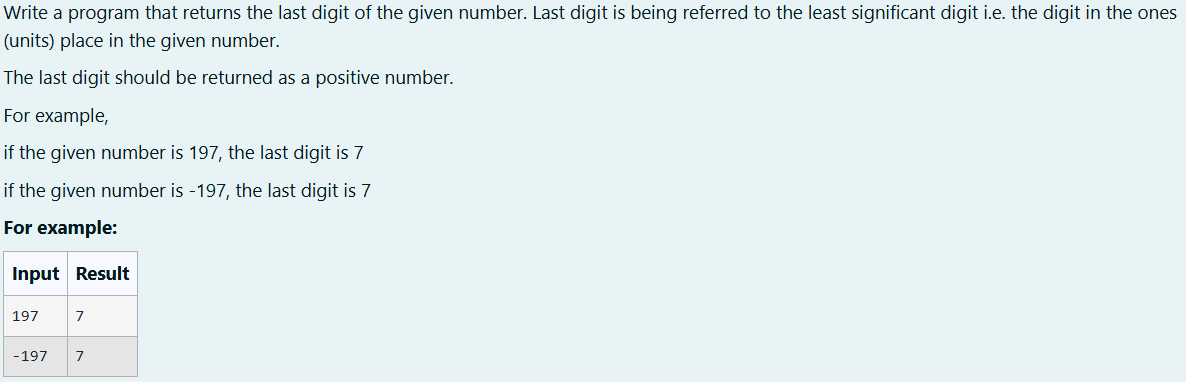
public static boolean isEven(int num){

return num%2==0;

}

}





import java.util.\*;

public class last{

public static void main(String[] args){

Scanner scanner=new Scanner(System.in);

int a=scanner.nextInt();

if(a<0){

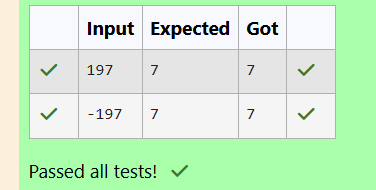
a=-(a);

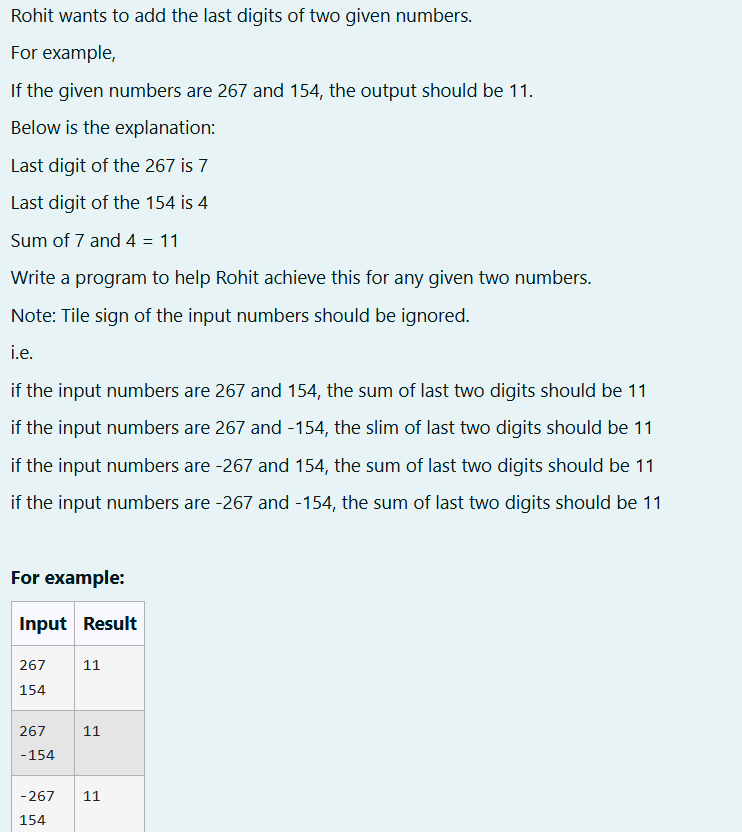
}

System.out.println(a%10);

}

}





import java.util.Scanner;

public class Sum

{

public static void main(String args[])

{

Scanner A=new Scanner (System.in);

int a=A.nextInt();

int b=A.nextInt();

int c=Math.abs(a);

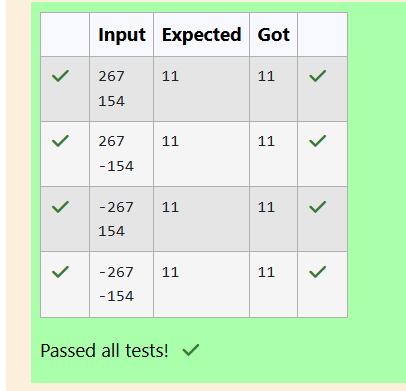
int d=Math.abs(b);

int e=(c%10)+(d%10);

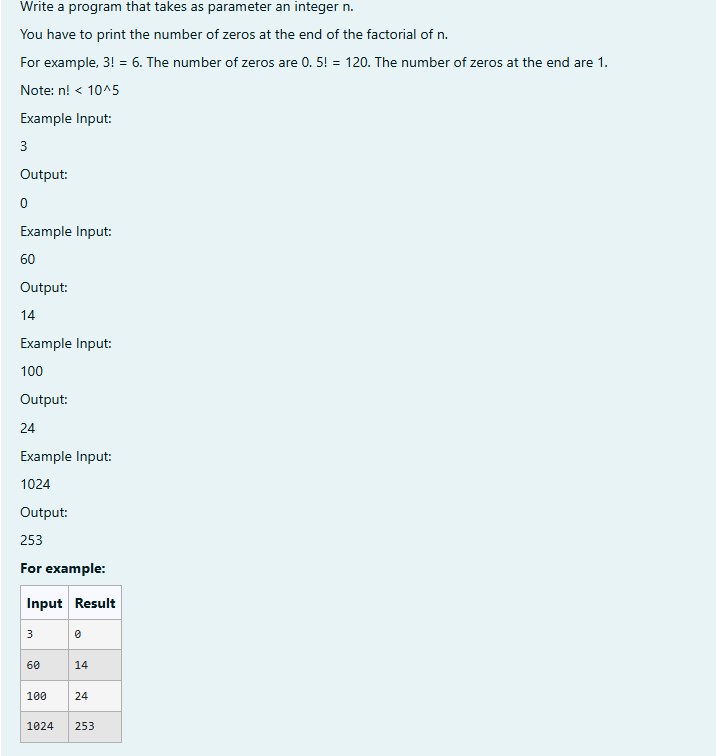
System.out.println(e);

}

}



WEEK 2



// Java program to count trailing 0s in n!

import java.io.\*;

import java.util.Scanner;

class prog {

// Function to return trailing

// 0s in factorial of n

static int findTrailingZeros(int n,int count)

{

if (n < 0) // Negative Number Edge Case

return -1;

// Initialize result

// Keep dividing n by powers

// of 5 and update count

for (int i = 5; n / i >= 1;i=i\*5)

count += n / i;

return count;

}

// Driver Code

public static void main(String[] args)

{

Scanner sc= new Scanner(System.in);

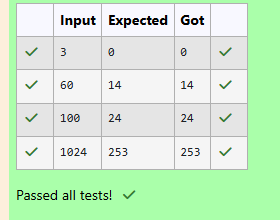
int n=sc.nextInt();

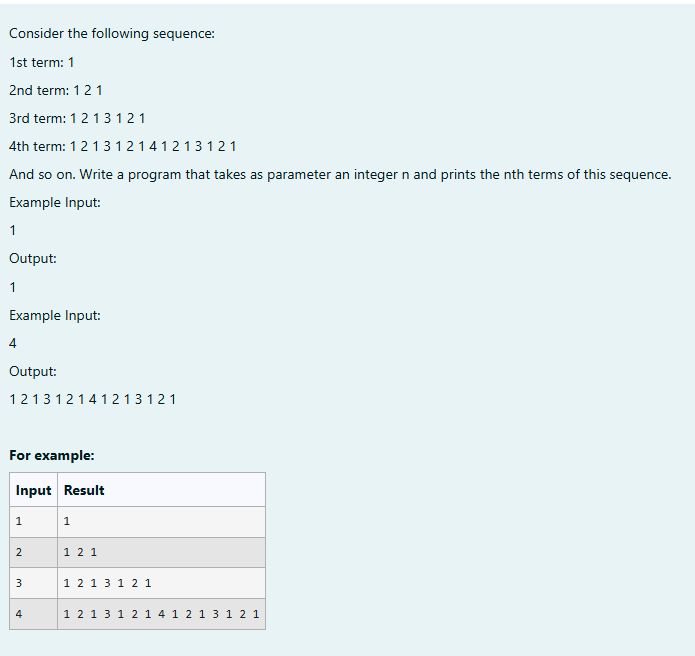
int count=0;

System.out.println(findTrailingZeros(n,count));

}

}





import java.util.Scanner;

public class Sequence{

public static void main(String args[]){

Scanner A=new Scanner(System.in);

int n=A.nextInt();

String t="1";

for(int i=2;i<=n;i++){

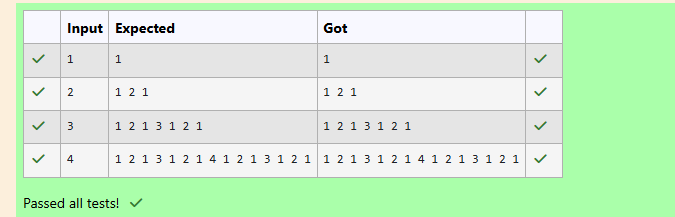
t+=" "+i+" "+t;

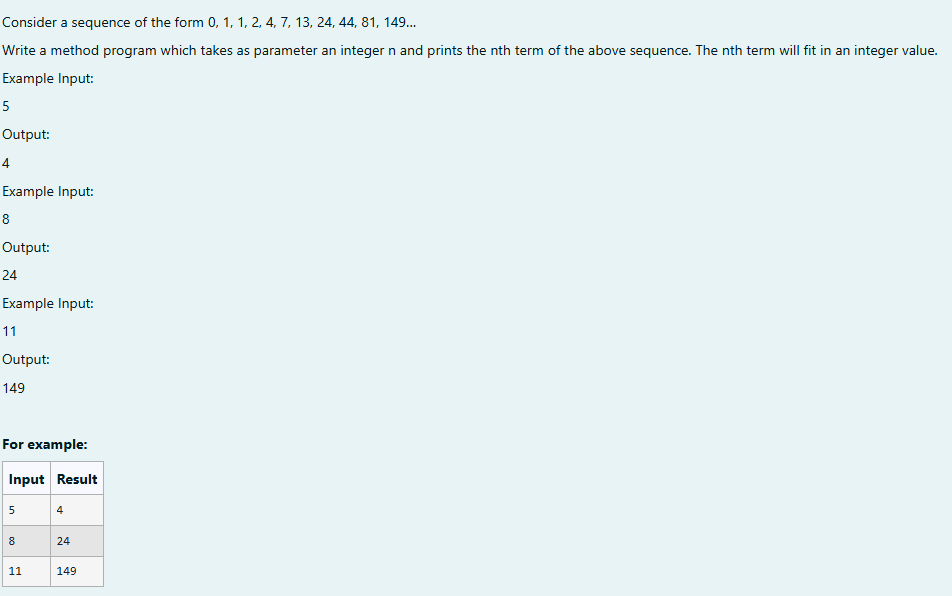
}

System.out.println(t);

}

}





import java.util.\*;

public class Fibo{

public static void main(String args[]){

Scanner s=new Scanner(System.in);

int a=s .nextInt();

int x=0;

int y=1;

int z=1;

int b=0;

for(int i=3;i<a;i++){

b=x+y+z;

x=y;

y=z;

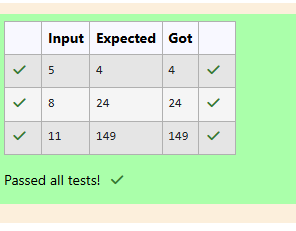
z=b;

}

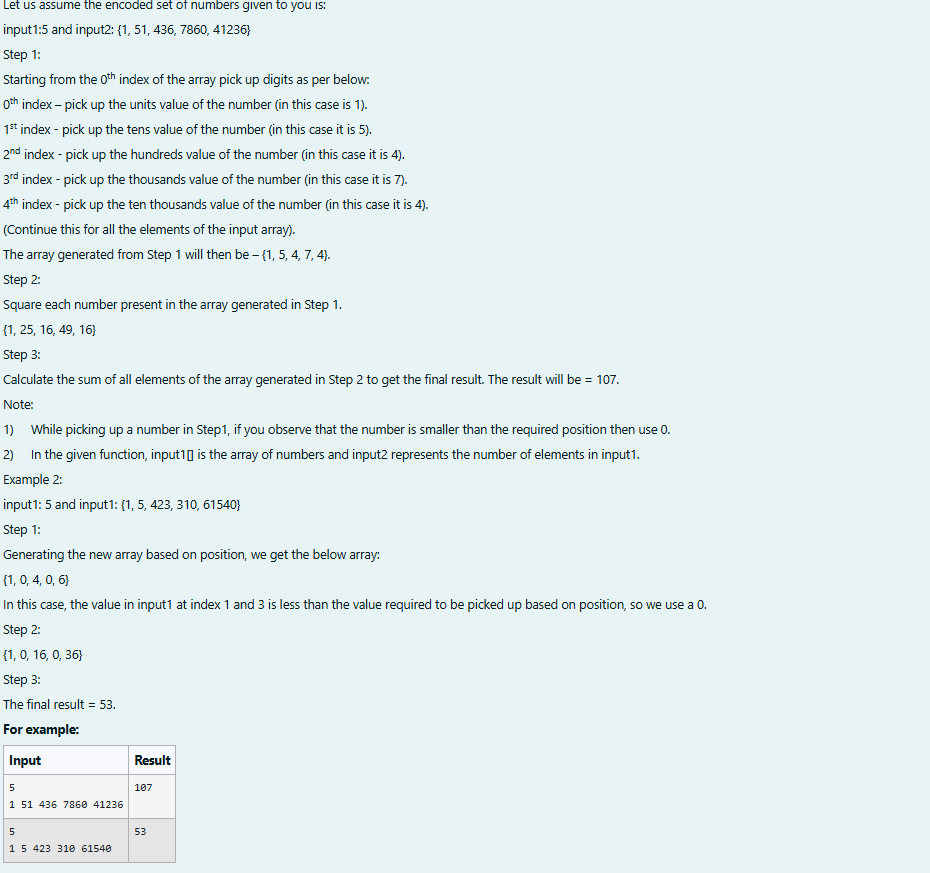
System.out.println(b);

}

}



WEEK 3



import java.util.Scanner;

public class SumOfSquaredDigits {



public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int input1 = scanner.nextInt();



int[] input2 = new int[input1];



for (int i = 0; i < input1; i++) {



input2[i] = scanner.nextInt();

}



int result = calculateSumOfSquaredDigits(input2);

System.out.println(result);

scanner.close();

}

public static int calculateSumOfSquaredDigits(int[] numbers) {

int[] extractedDigits = new int[numbers.length];

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

int number = numbers[i];

int digit = 0;

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

digit = number % 10;

number /= 10;

}

extractedDigits[i] = digit;

}

int sumOfSquares = 0;

for (int digit : extractedDigits) {

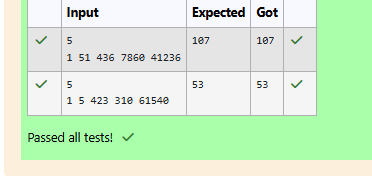
sumOfSquares += digit \* digit;

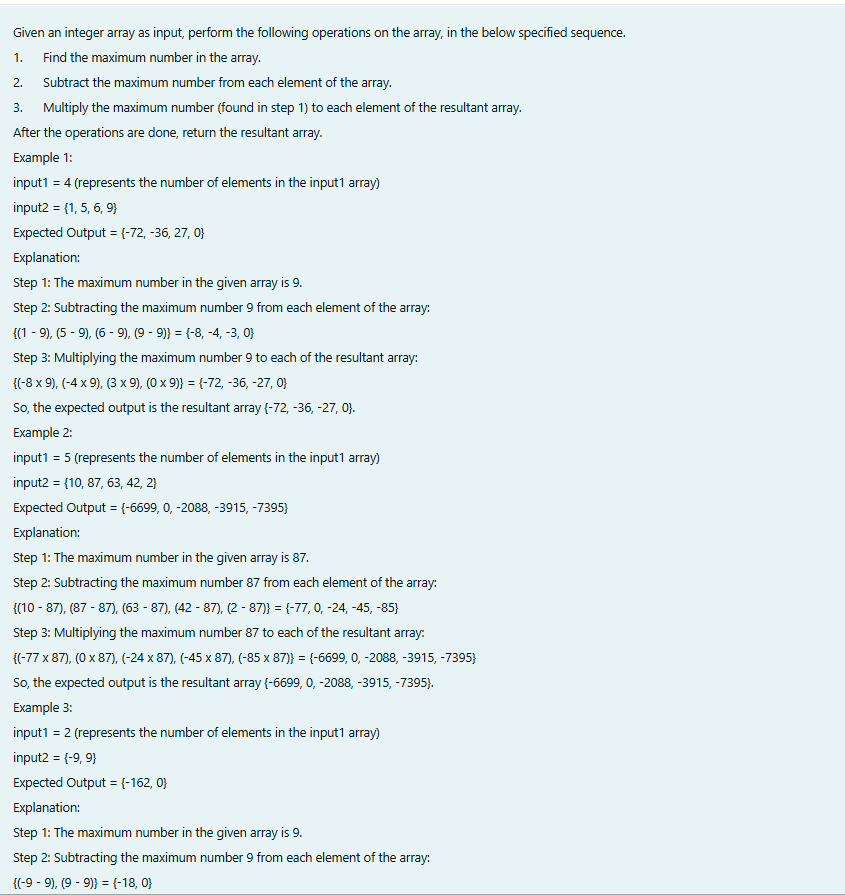
}

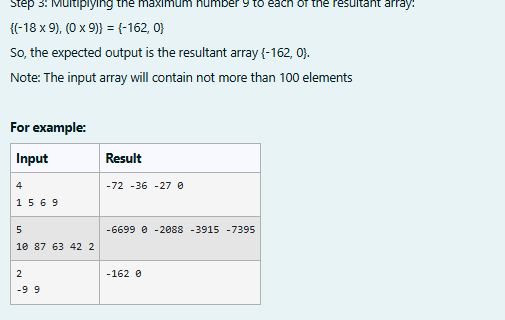
return sumOfSquares;

}

}







import java.util.Arrays;

import java.util.Scanner;

public class ArrayTransformation {

public static int[] transformArray(int[] inputArray) {

int maxNum = Integer.MIN\_VALUE;

for (int num : inputArray) {

if (num > maxNum) {

maxNum = num;

}

}

int[] subtractedArray = new int[inputArray.length];

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

subtractedArray[i] = inputArray[i] - maxNum;

}

int[] resultantArray = new int[subtractedArray.length];

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

resultantArray[i] = subtractedArray[i] \* maxNum;

}

return resultantArray;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

int[] inputArray = new int[n];

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

inputArray[i] = scanner.nextInt();

}

int[] result = transformArray(inputArray);

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

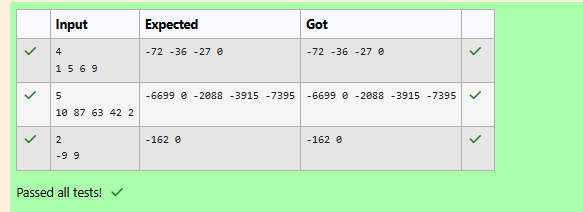
System.out.print(result[i]+" ");

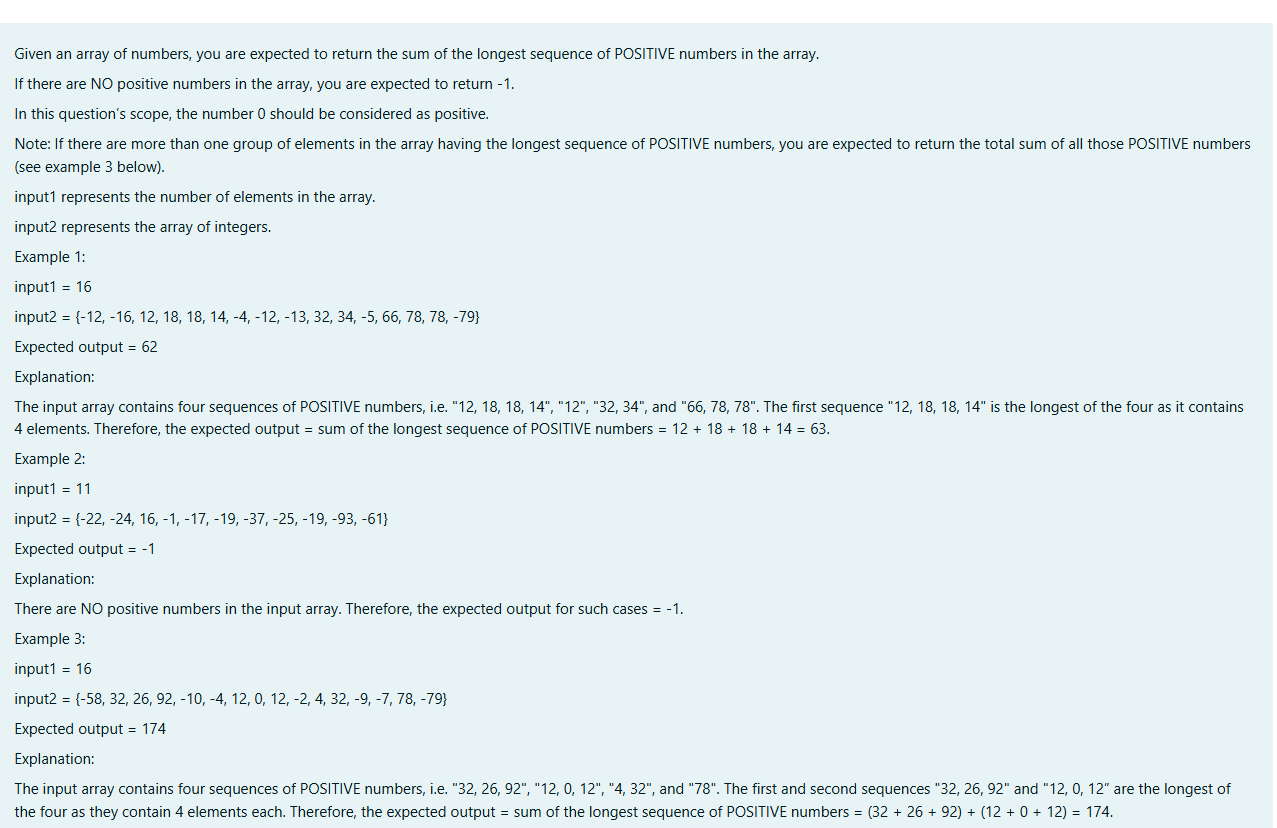
}

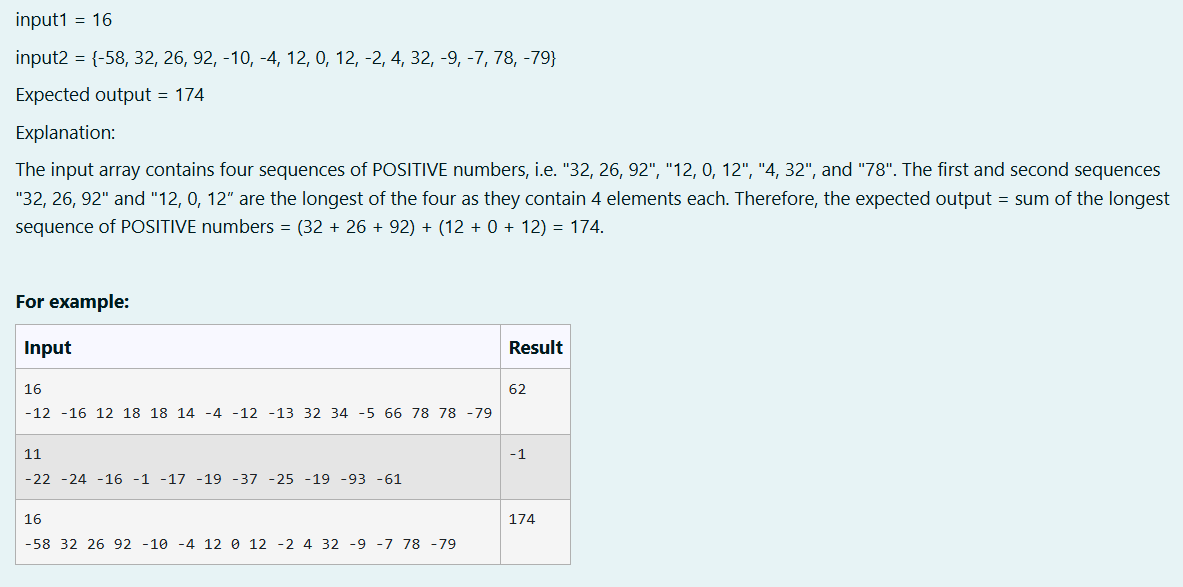
scanner.close();

}

}







import java.util.Scanner;

public class LongestPositiveSequence {

public static int sumOfLongestPositiveSequence(int n, int[] arr) {

int maxLength = 0;

int maxSum = 0;

int currentLength = 0;

int currentSum = 0;

for (int num : arr) {

if (num >= 0) { // Count 0 as positive

currentLength++;

currentSum += num;

} else {

if (currentLength > maxLength) {

maxLength = currentLength;

maxSum = currentSum;

} else if (currentLength == maxLength) {

maxSum += currentSum;

}

// Reset current sequence

currentLength = 0;

currentSum = 0;

}

}

// Final check in case the array ends with a positive sequence

if (currentLength > maxLength) {

maxLength = currentLength;

maxSum = currentSum;

} else if (currentLength == maxLength) {

maxSum += currentSum;

}

return maxLength > 0 ? maxSum : -1;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int input1 = scanner.nextInt();

int[] input2 = new int[input1];

for (int i = 0; i < input1; i++) {

input2[i] = scanner.nextInt();

}

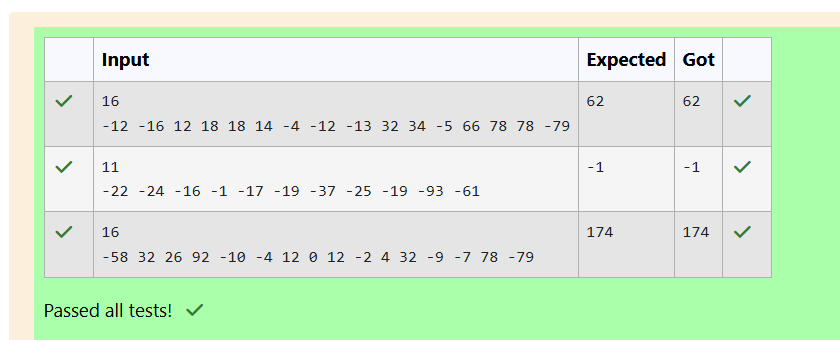
int result = sumOfLongestPositiveSequence(input1, input2);

System.out.println(result);

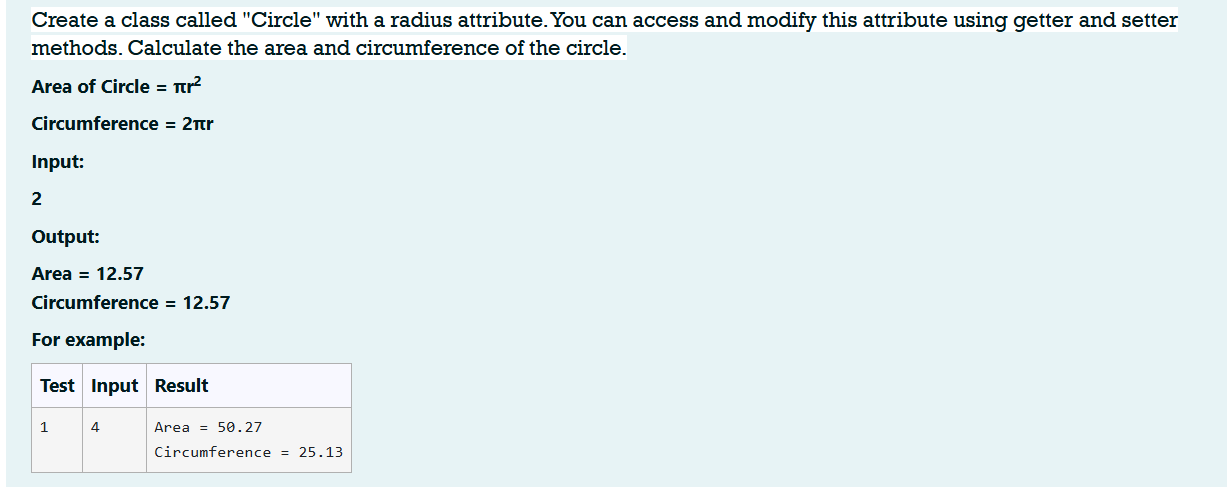
scanner.close();

}

}



WEEK 4



import java.io.\*;

import java.util.\*;

class Circle

{

private double radius;

public Circle (double radius)

{

this.radius=radius;

}

public void setRadius (double radius)

{

this.radius=radius;

}

public double getRadius()

{

return radius;

}

public double calculateArea()

{

return Math.PI\*radius\*radius;

}

public double calculateCircumference()

{

return 2\*Math.PI\*radius;

}

}

class prog

{

public static void main(String[] args)

{

Scanner sc= new Scanner(System.in);

int r=sc.nextInt();

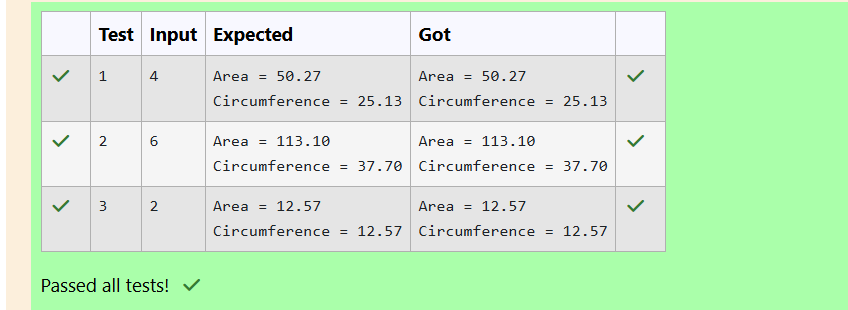
Circle c= new Circle(r);

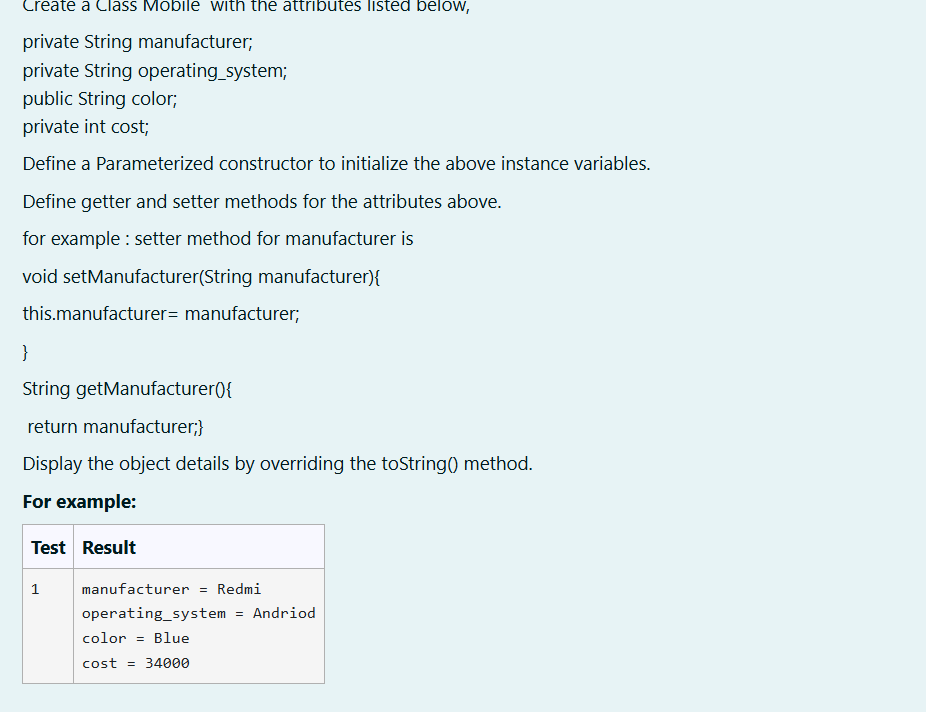
System.out.println("Area = "+String.format("%.2f", c.calculateArea()));

System.out.println("Circumference = "+String.format("%.2f", c.calculateCircumference()));

}

}





import java.util.\*;

public class Mobile

{

private String manufacturer;

private String operating\_system;

public String color;

public int cost;

Mobile (String m, String os, String col, int cost)

{

this.manufacturer=m;

this.operating\_system=os;

this.color=col;

this.cost=cost;

}

public String getM()

{

return manufacturer;

}

public String getOs()

{

return operating\_system;

}

public String getc()

{

return color;

}

public int getcost()

{

return cost;

}

public String toString()

{

return "manufacturer = "+getM()+'\n'+"operating\_system = "+getOs ()+'\n'+"color = "+getc()+'\n'+"cost = "+getcost();

}

public static void main(String[] args)

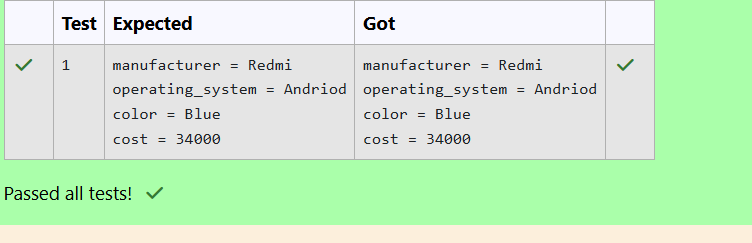
{

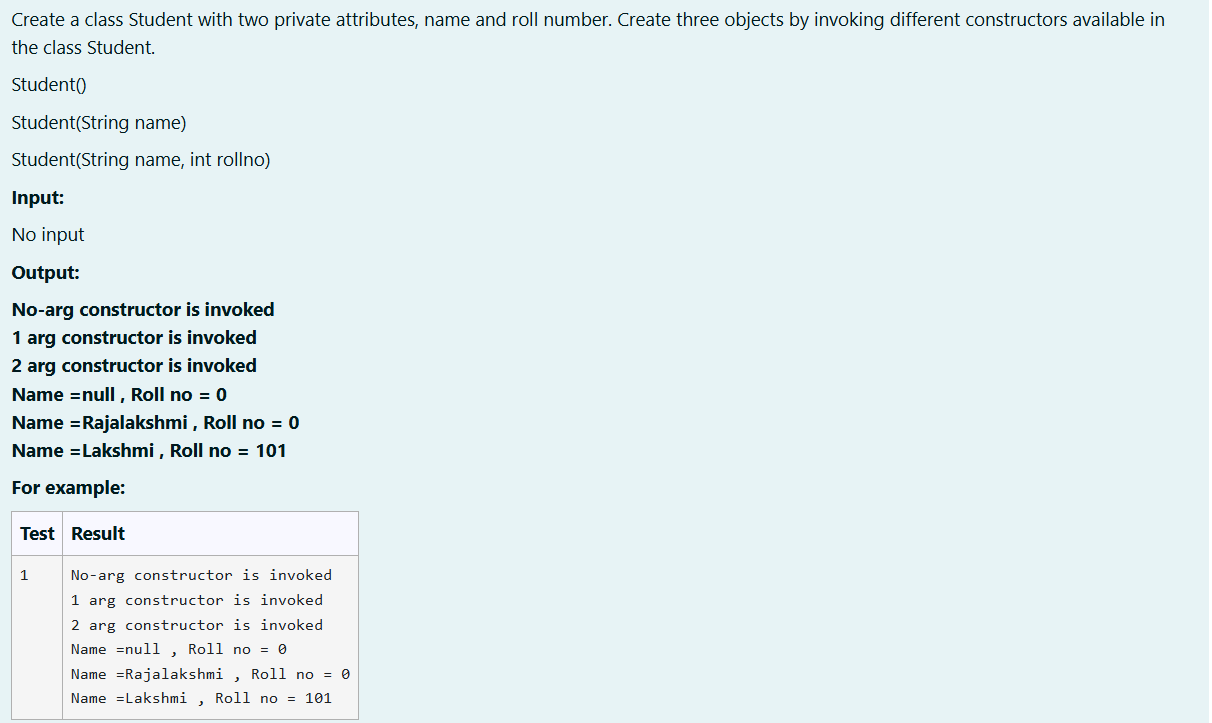
Mobile r=new Mobile ("Redmi", "Andriod", "Blue", 34000);

System.out.println(r.toString());

}

}





public class Student{

private String name;

private int roll;

Student(){

this.name=null;

this.roll=0;

}

Student(String name) {

this.name=name;

this.roll=0;

}

Student(String name, int roll){

this.name=name;

this.roll=roll;

}

public void display(){

System.out.println("Name ="+name+" , Roll no = "+roll);

}

public static void main(String[] args) {

Student s=new Student();

Student s1=new Student("Rajalakshmi");

Student s2=new Student("Lakshmi", 101);

System.out.println("No-arg constructor is invoked\n1 arg constructor is invoked\n2 arg constructor is invoked");

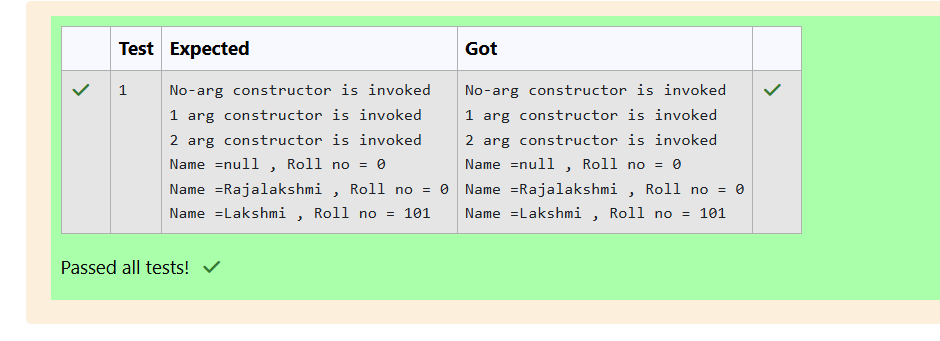
s.display();

s1.display();

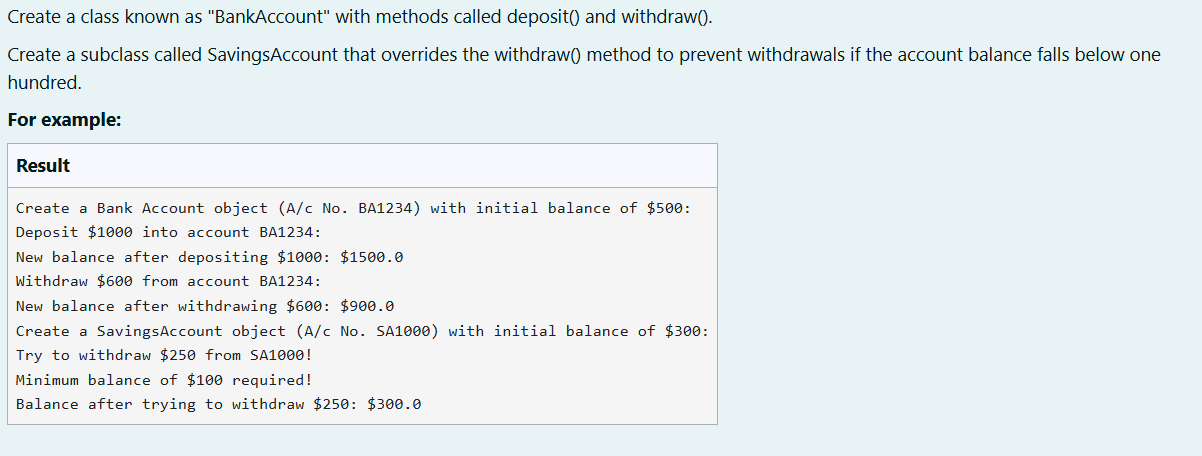
s2.display();

}

}



WEEK 5



class BankAccount {

protected String accountNumber;

protected double balance;

// Constructor to initialize account number and balance

public BankAccount(String accountNumber, double balance) {

this.accountNumber = accountNumber;

this.balance = balance;

}

// Method to deposit money

public void deposit(double amount) {

if (amount > 0) {

balance += amount;

System.out.println("Deposit $" + (int)amount + " into account " + accountNumber + ":");

System.out.println("New balance after depositing $" + (int)amount + ": $" + balance);

} else {

System.out.println("Deposit amount must be positive.");

}

}

// Method to withdraw money

public void withdraw(double amount) {

if (amount > 0 && amount <= balance) {

balance -= amount;

System.out.println("Withdraw $" + (int)amount + " from account " + accountNumber + ":");

System.out.println("New balance after withdrawing $" + (int)amount + ": $" + balance);

} else {

System.out.println("Insufficient balance for withdrawal.");

}

}

}

class SavingsAccount extends BankAccount {

private static final double MIN\_BALANCE = 100.0;

// Constructor for SavingsAccount

public SavingsAccount(String accountNumber, double balance) {

super(accountNumber, balance);

}

// Overridden withdraw method to enforce minimum balance

@Override

public void withdraw(double amount) {

System.out.println("Try to withdraw $" + (int)amount + " from " + accountNumber + "!");

if (balance - amount >= MIN\_BALANCE) {

super.withdraw(amount);

} else {

System.out.println("Minimum balance of $100 required!");

System.out.println("Balance after trying to withdraw $" + (int)amount + ": $" + balance);

}

}

}

public class Main {

public static void main(String[] args) {

// Creating a BankAccount object

BankAccount ba = new BankAccount("BA1234", 500);

System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");

ba.deposit(1000);

ba.withdraw(600);

// Creating a SavingsAccount object

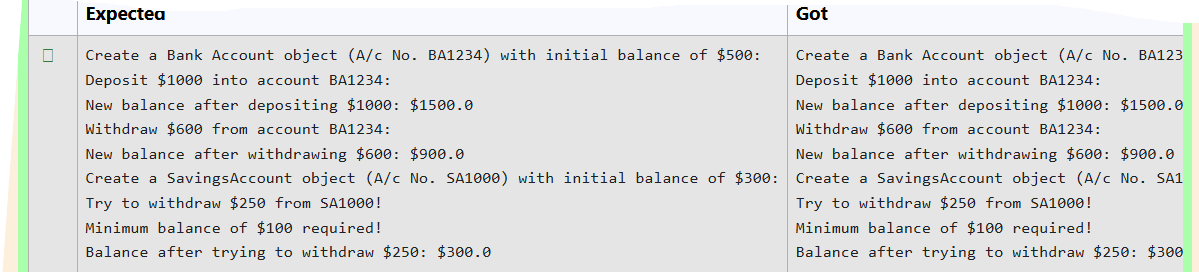
SavingsAccount sa = new SavingsAccount("SA1000", 300);

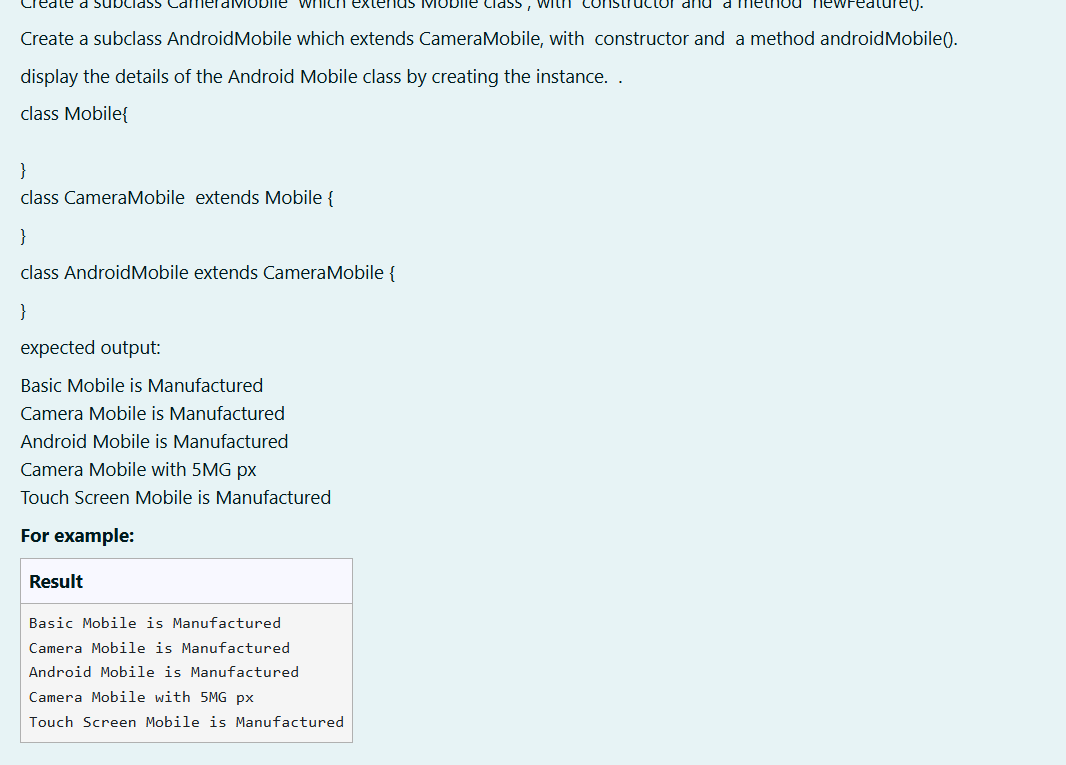
System.out.println("Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:");

sa.withdraw(250);

}

}





class Mobile {

// Constructor

public Mobile() {

System.out.println("Basic Mobile is Manufactured");

}

// Method to indicate it's a basic mobile

public void basicMobile() {

System.out.println("This is a basic mobile.");

}

}

// Derived class extending Mobile

class CameraMobile extends Mobile {

// Constructor

public CameraMobile() {

// Call the parent class constructor

super();

System.out.println("Camera Mobile is Manufactured");

}

// Method to indicate camera features

public void newFeature() {

System.out.println("Camera Mobile with 5MG px");

}

}

// Derived class extending CameraMobile

class AndroidMobile extends CameraMobile {

// Constructor

public AndroidMobile() {

// Call the parent class constructor

super();

System.out.println("Android Mobile is Manufactured");

}

// Method to indicate Android features

public void androidMobile() {

System.out.println("Touch Screen Mobile is Manufactured");

}

}

// Main class to test the inheritance

public class Main{

public static void main(String[] args) {

// Create an object of AndroidMobile

AndroidMobile myAndroid = new AndroidMobile();

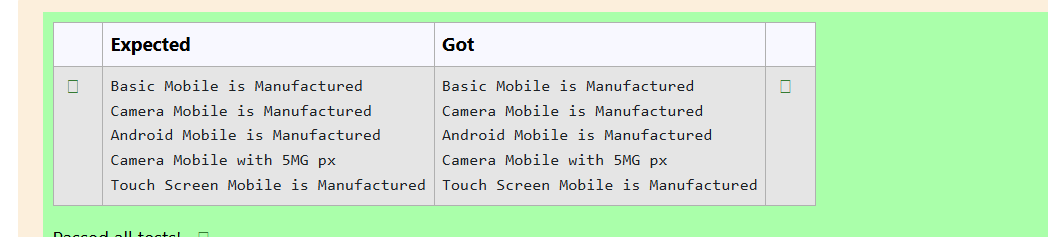
// Calling methods from the inheritance chain

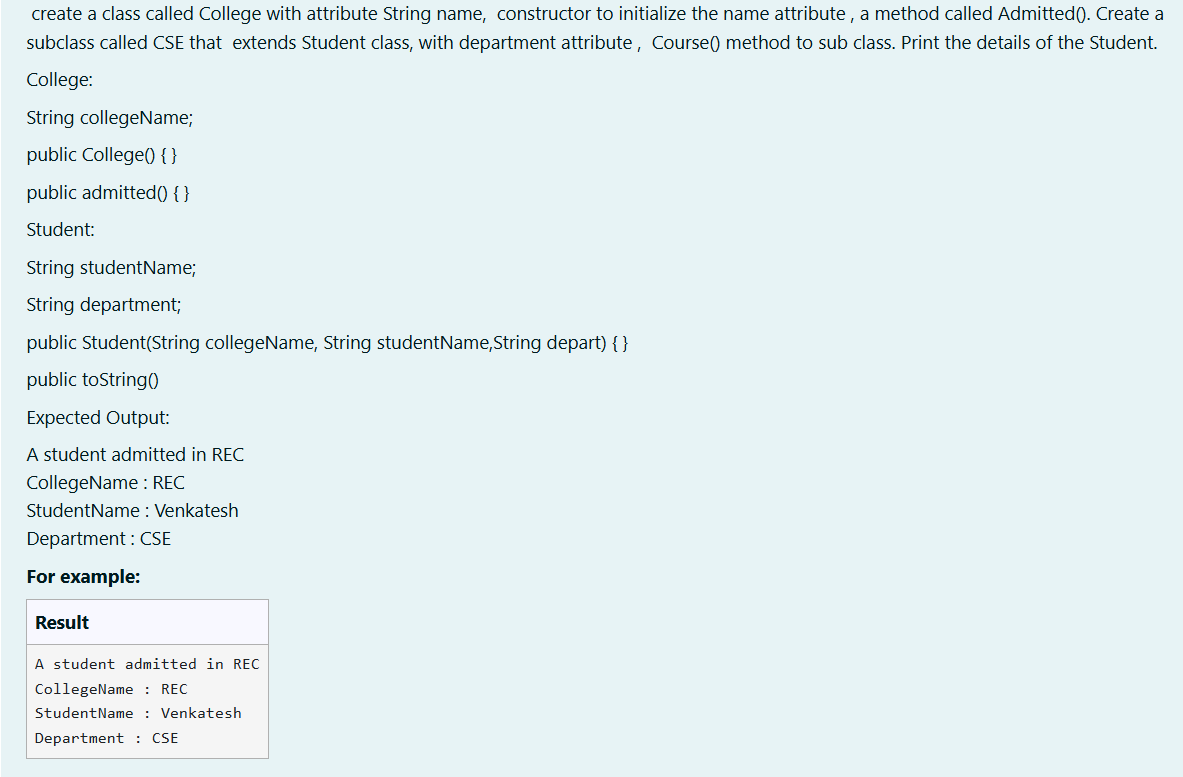
myAndroid.newFeature(); // Method from CameraMobile class

myAndroid.androidMobile(); // Method from AndroidMobile class

}

}





class College {

protected String collegeName;

public College(String collegeName) {

// Initialize the instance variable

this.collegeName = collegeName;

}

public void admitted() {

System.out.println("A student admitted in " + collegeName);

}

}

class Student extends College {

private String studentName;

private String department;

public Student(String collegeName, String studentName, String department) {

// Initialize the instance variables

super(collegeName); // Call to the parent class constructor

this.studentName = studentName;

this.department = department;

}

@Override

public String toString() {

// Return the details of the student

return "CollegeName : " + collegeName + "\n" +

"StudentName : " + studentName + "\n" +

"Department : " + department;

}

}

class prog {

public static void main(String[] args) {

Student s1 = new Student("REC", "Venkatesh", "CSE");

// Invoke the admitted() method

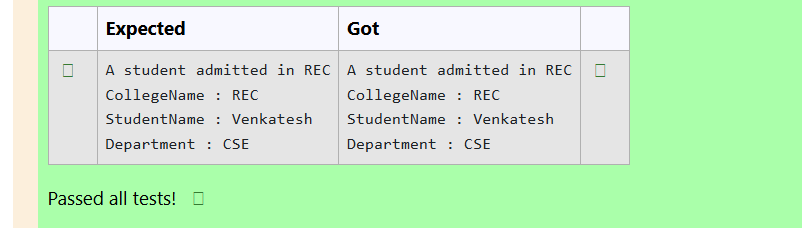
s1.admitted();

// Print the details of the student

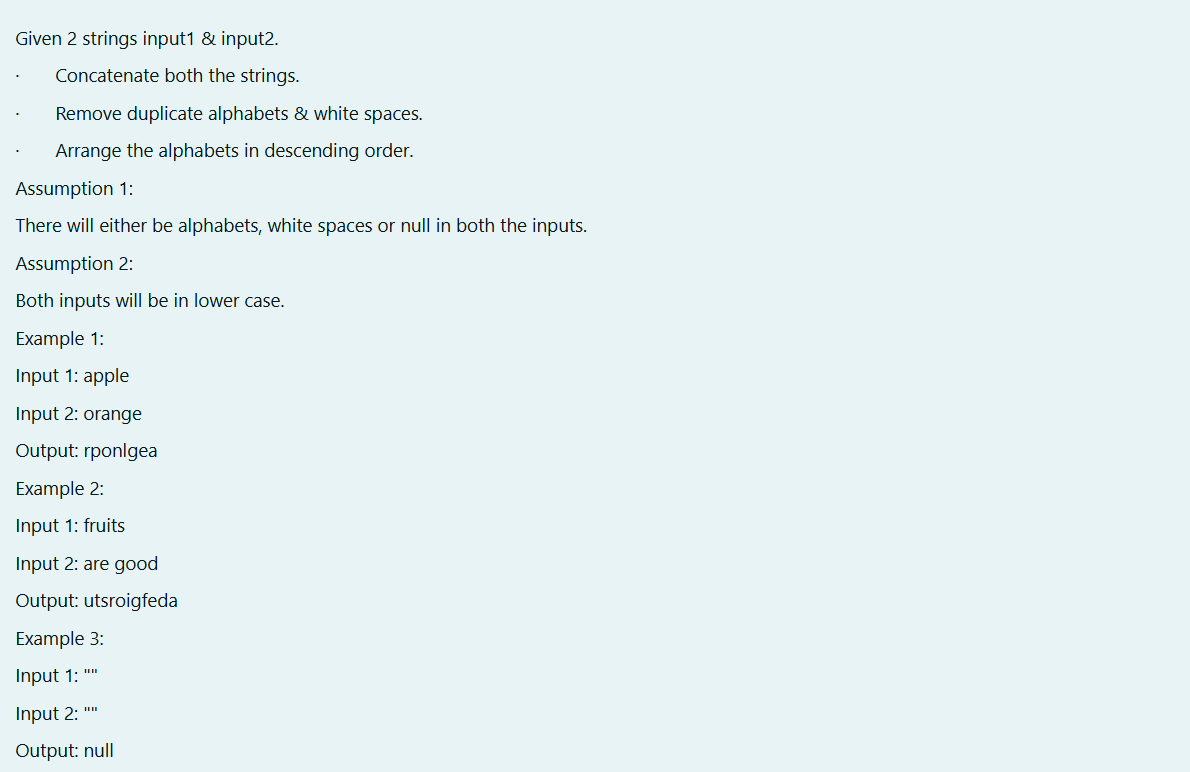
System.out.println(s1.toString());

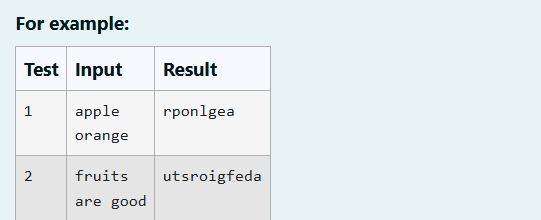
}

}



WEEK 6





import java.util.\*;

public class StringMergeSort {

public static String mergeAndSort(String input1, String input2) {

// Step 1: Concatenate both strings

String concatenated = input1 + input2;

// Step 2: Remove duplicate characters and whitespaces

Set<Character> uniqueChars = new HashSet<>();

for (char ch : concatenated.toCharArray()) {

if (ch != ' ') {

uniqueChars.add(ch);

}

}

// Step 3: Sort the characters in descending order

List<Character> sortedList = new ArrayList<>(uniqueChars);

Collections.sort(sortedList, Collections.reverseOrder());

// Step 4: Create the final string

StringBuilder result = new StringBuilder();

for (char ch : sortedList) {

result.append(ch);

}

// Step 5: If the result is empty, return "null", else return the result

return result.length() > 0 ? result.toString() : "null";

}

public static void main(String[] args) {

// Using Scanner to take input from the user

Scanner scanner = new Scanner(System.in);

//nter the first string: ");

String input1 = scanner.nextLine();

// Input 2

String input2 = scanner.nextLine();

// Calling the mergeAndSort method and printing the result

String result = mergeAndSort(input1, input2);

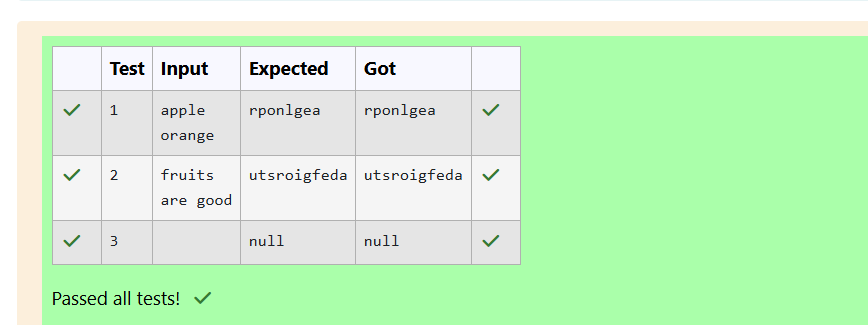
System.out.println(result);

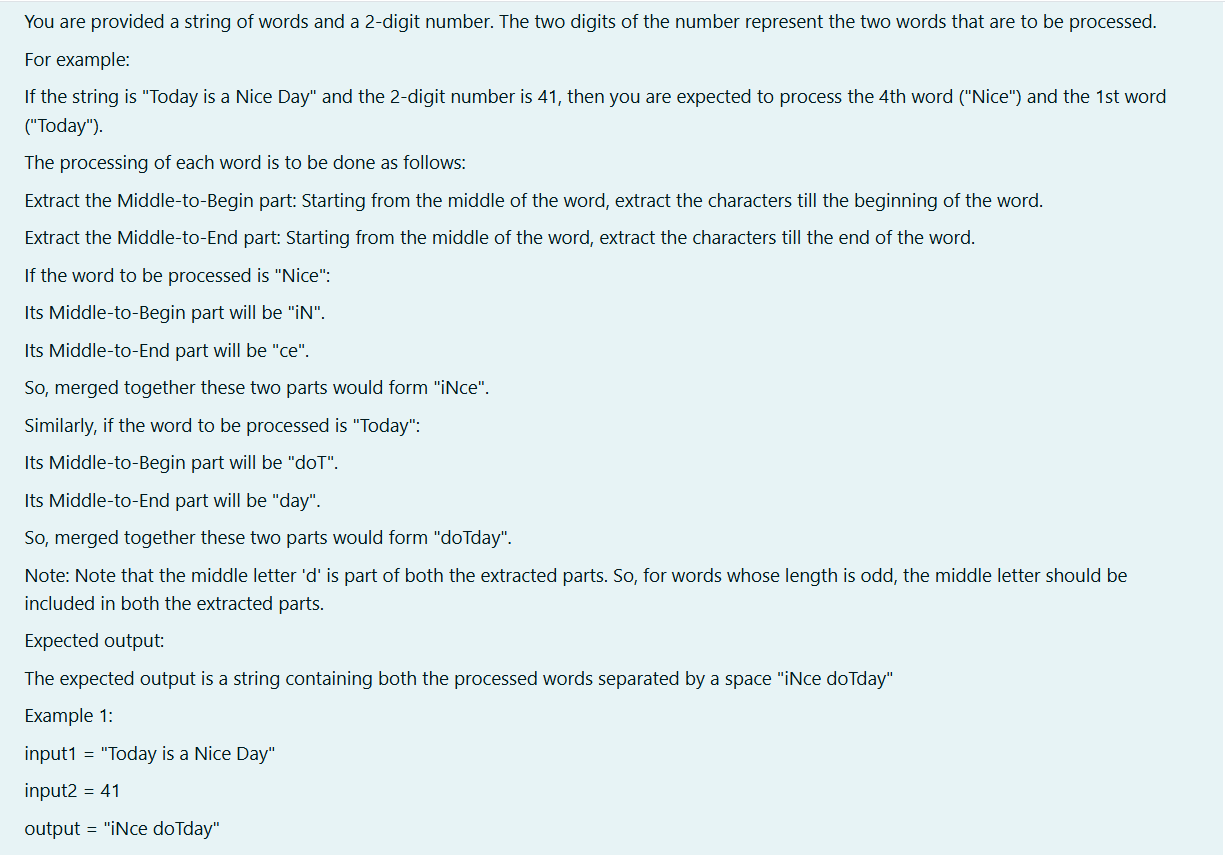
// Closing the scanner

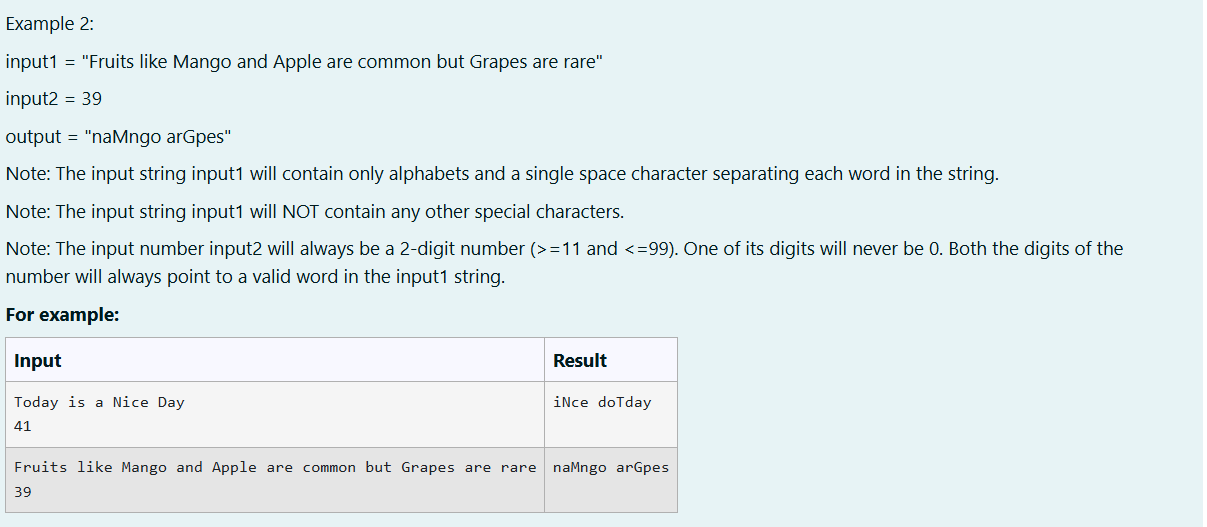
scanner.close();

}

}







import java.util.Scanner;

public class WordProcessor {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

// Input the string of words

String input = sc.nextLine();

// Input the 2-digit number

int number = sc.nextInt();

// Split the string into words

String[] words = input.split(" ");

// Extract the two positions from the 2-digit number

int pos1 = number / 10; // first digit (ten's place)

int pos2 = number % 10; // second digit (unit's place)

// Adjust positions for 0-based index

pos1--;

pos2--;

// Process both words

String result1 = processWord(words[pos1]);

String result2 = processWord(words[pos2]);

// Combine the results and print

String result = result1 + " " + result2;

System.out.println(result);

}

// Function to process each word

private static String processWord(String word) {

int len = word.length();

int mid = len / 2;

// Handle odd-length words correctly by including the middle letter in both parts

String middleToBegin;

String middleToEnd;

if (len % 2 == 0) {

// Even length

middleToBegin = new StringBuilder(word.substring(0, mid)).reverse().toString();

middleToEnd = word.substring(mid);

} else {

// Odd length

middleToBegin = new StringBuilder(word.substring(0, mid + 1)).reverse().toString();

middleToEnd = word.substring(mid);

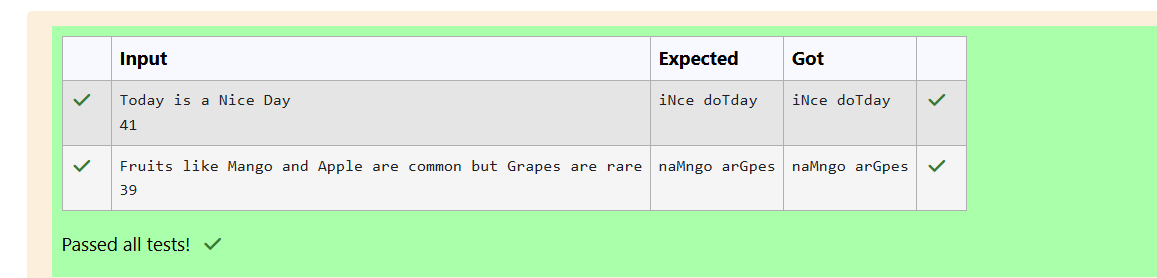
}

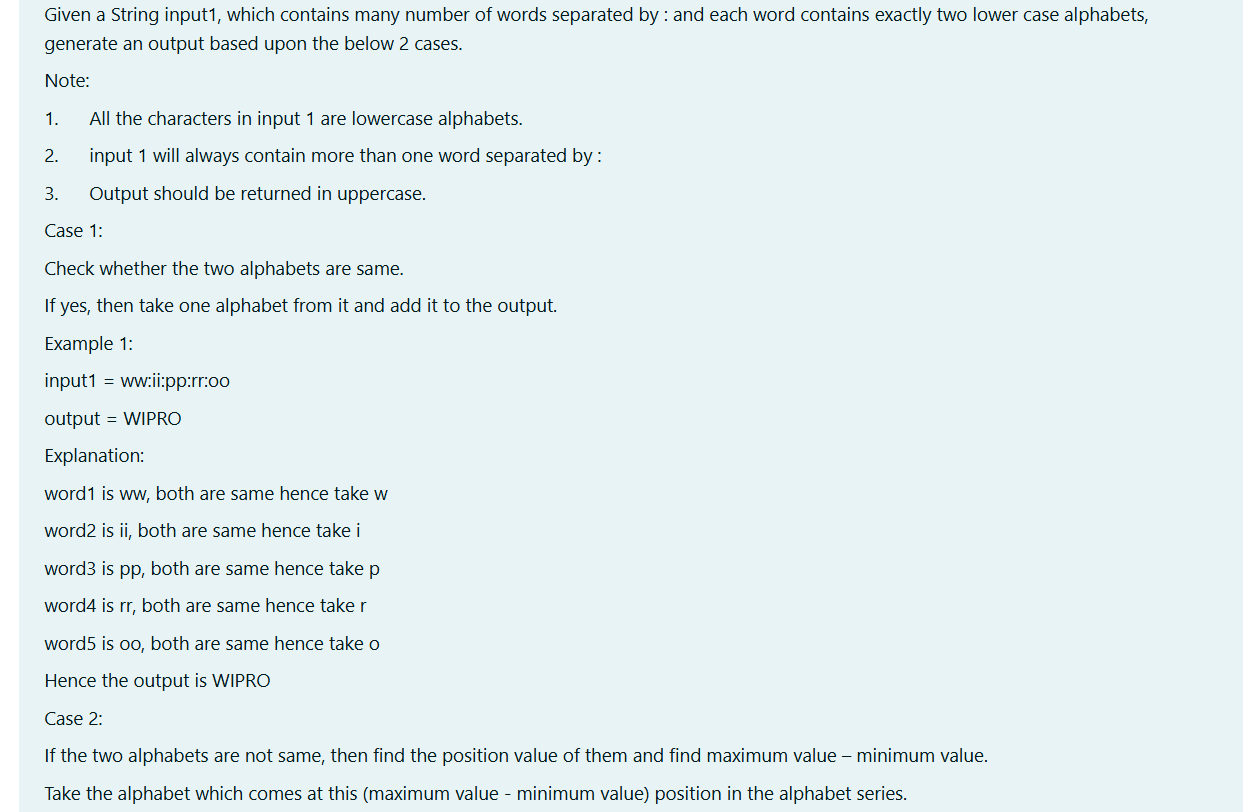
// Combine both parts

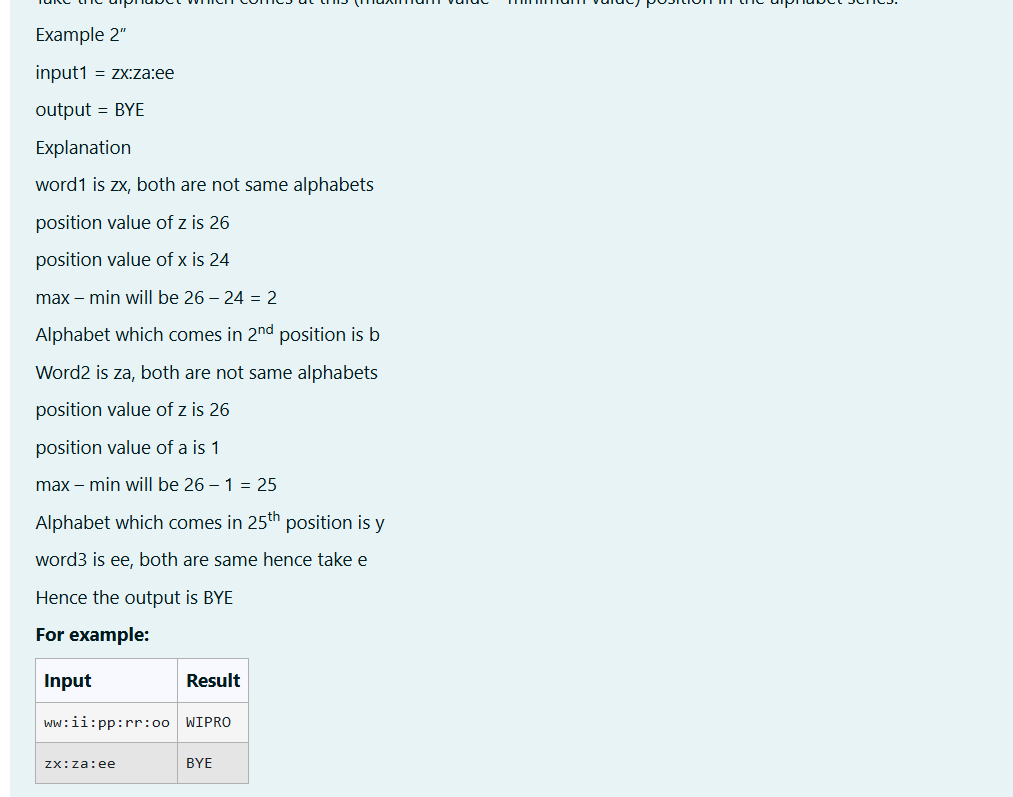
return middleToBegin + middleToEnd;

}

}







import java.util.Scanner;

public class StringManipulation {

// Method to find the alphabet based on the given logic

public static char findChar(char ch1, char ch2) {

if (ch1 == ch2) {

// Case 1: If both characters are the same

return ch1;

} else {

// Case 2: If both characters are different

int max = Math.max(ch1 - 'a' + 1, ch2 - 'a' + 1);

int min = Math.min(ch1 - 'a' + 1, ch2 - 'a' + 1);

int pos = max - min;

return (char) ('a' + pos - 1); // Position starts at 1, so adjust by -1

}

}

// Method to process the input string and generate the output

public static String processString(String input) {

// Split the input string by ":"

String[] pairs = input.split(":");

StringBuilder result = new StringBuilder();

// Process each pair of characters

for (String pair : pairs) {

char ch1 = pair.charAt(0);

char ch2 = pair.charAt(1);

// Add the corresponding character to the result

result.append(findChar(ch1, ch2));

}

// Convert result to uppercase as specified

return result.toString().toUpperCase();

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Input from use

//System.out.print("Enter the string (e.g., ww:ii:pp:rr:oo): ");

String input = scanner.nextLine();

// Process the string and get the result

String result = processString(input);

// Print the result

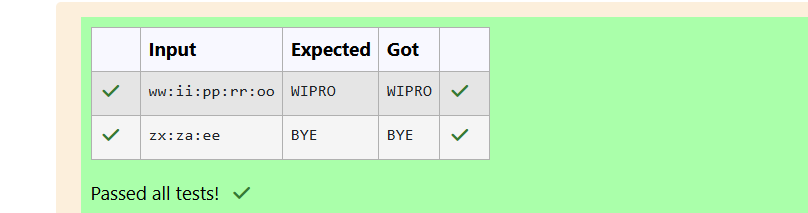
System.out.println( result);

// Close the scanner

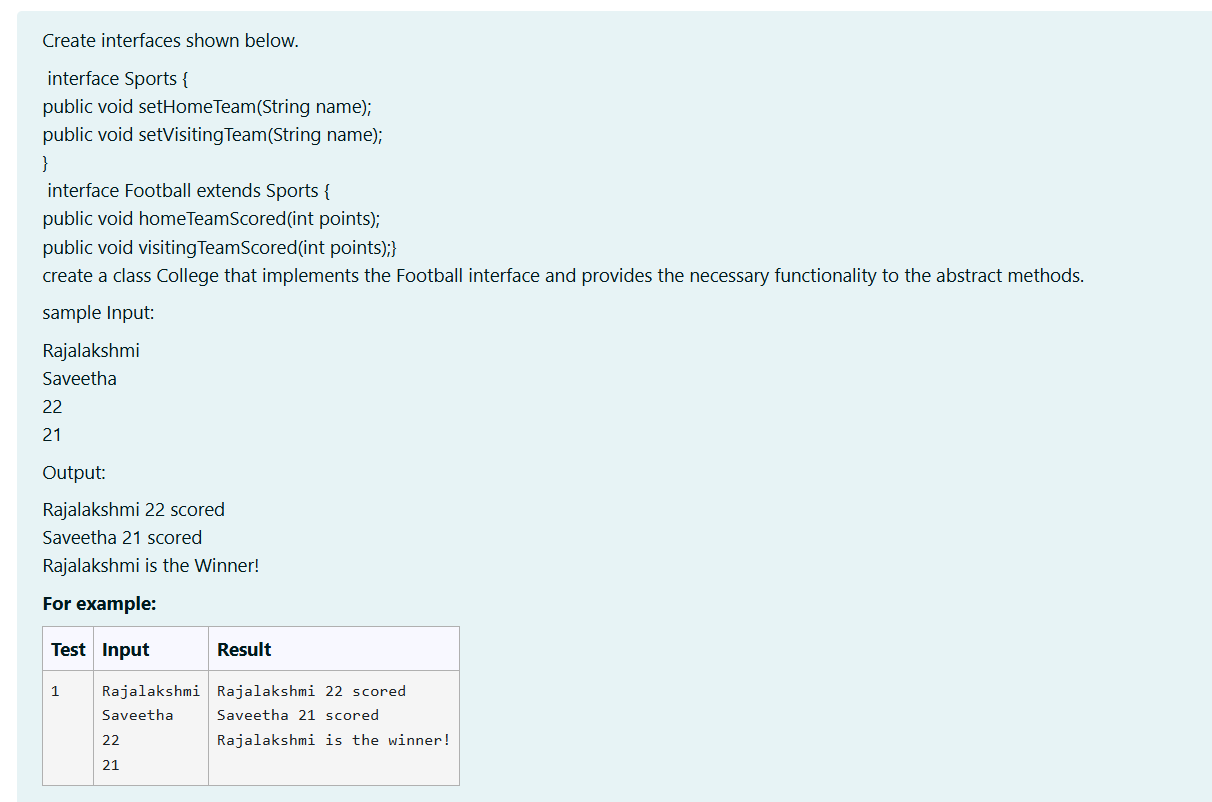
scanner.close();

}

}



WEEK 7



import java.util.Scanner;

interface Sports {

public void setHomeTeam(String name);

public void setVisitingTeam(String name);

}

interface Football extends Sports {

public void homeTeamScored(int points);

public void visitingTeamScored(int points);

}

class College implements Football {

String homeTeam;

String visitingTeam;

int points;

public void setHomeTeam(String name){

homeTeam=name;

}

public void setVisitingTeam(String name){

visitingTeam=name;

}

public void homeTeamScored(int points){

System.out.println(homeTeam+" "+points+" scored");

}

public void visitingTeamScored(int points){

System.out.println(visitingTeam+" "+points+" scored");

}

public void winningTeam(int p1, int p2){

if(p1>p2){

System.out.println(homeTeam+ " is the winner!");}

else if(p1<p2){

System.out.println(visitingTeam+ " is the winner!");}

else{

System.out.println("It's a tie match.");

}

}}

public class Main{

public static void main(String[] args){

String hname;

Scanner sc= new Scanner(System.in);

hname=sc.nextLine();

String vteam=sc.nextLine();

int a = sc.nextInt();

int b = sc.nextInt();

int htpoints=a;

int vtpoints=b;

College s= new College();

s.setHomeTeam(hname);

s.setVisitingTeam(vteam);

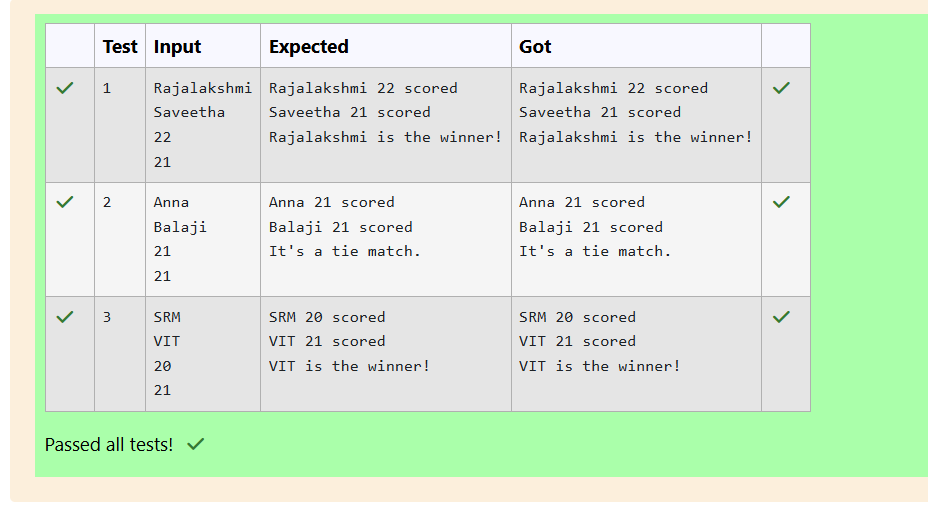
s.homeTeamScored(htpoints);

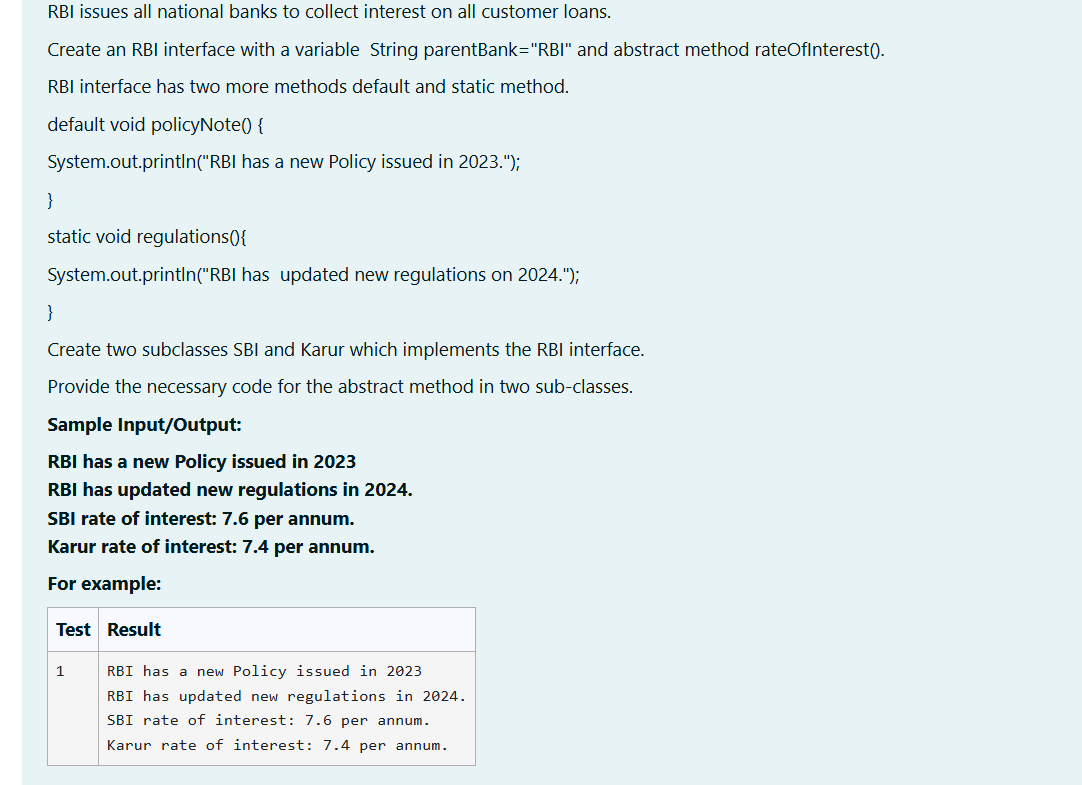
s.visitingTeamScored(vtpoints);

s.winningTeam(htpoints, vtpoints);

}

}





interface RBI{

String parentBank="RBI";

void rateofinterest();

default void policyNote(){

System.out.println("RBI has a new Policy issued in 2023");

}

static void regulation(){

System.out.println("RBI has updated new regulations in 2024.");

}

}

class SBI implements RBI{

public void rateofinterest(){

System.out.println("SBI rate of interest: 7.6 per annum.");

}

}

class Karur implements RBI{

public void rateofinterest(){

System.out.println("Karur rate of interest: 7.4 per annum.");

}

}

public class Main{

public static void main(String[] args){

SBI s = new SBI();

Karur k = new Karur();

s.policyNote();

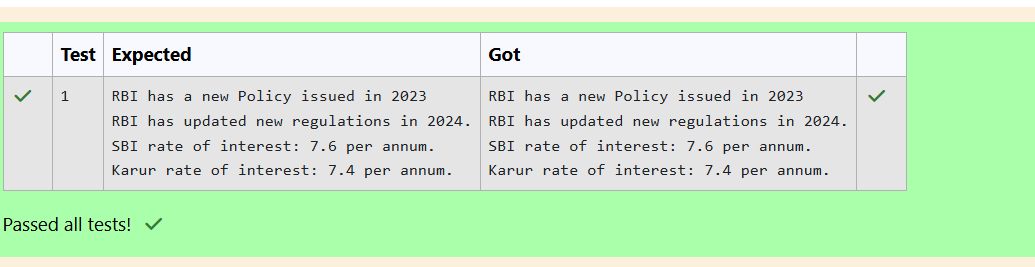
RBI.regulation();

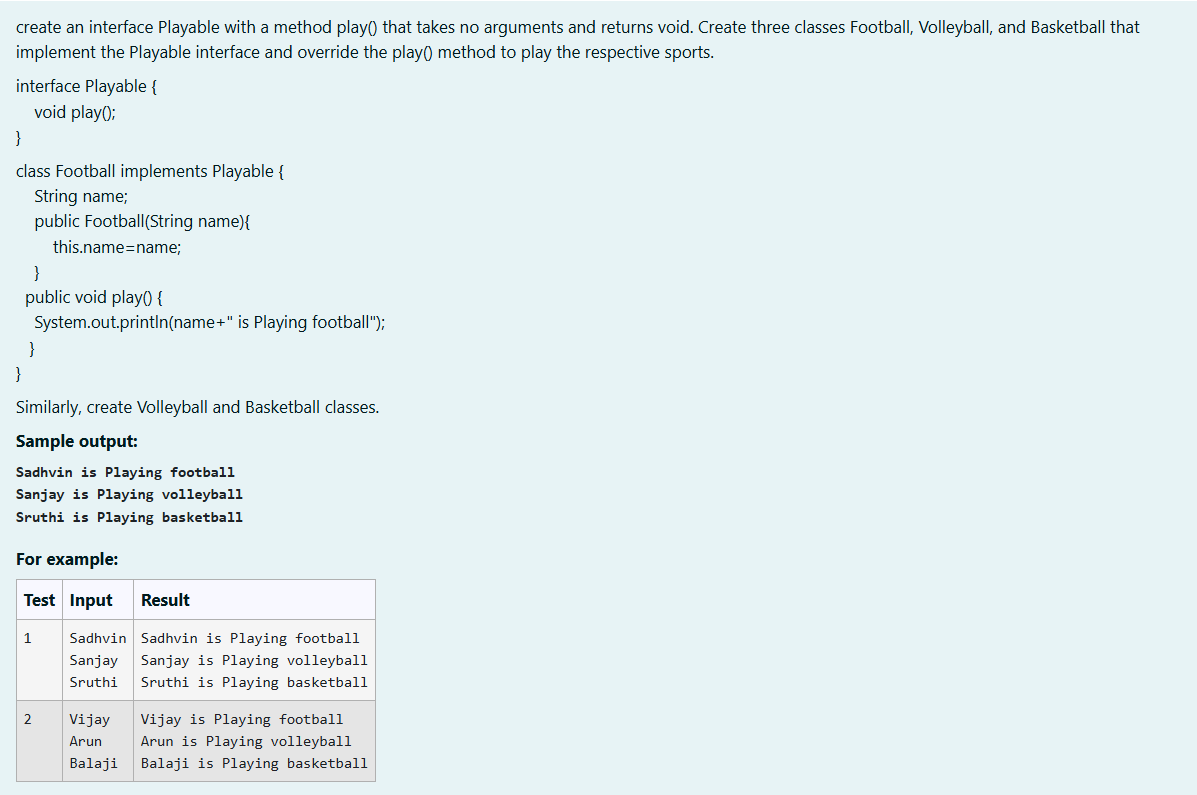
s.rateofinterest();

k.rateofinterest();

}

}





import java.util.\*;

interface playable{

void play();

}

class football implements playable{

String name;

public football(String name){

this.name=name;

}

public void play(){

System.out.println(name+" is Playing football");

}

}

class volleyball implements playable{

String name;

public volleyball(String name){

this.name=name;

}

public void play(){

System.out.println(name+" is Playing volleyball");

}

}

class basketball implements playable{

String name;

public basketball(String name){

this.name=name;

}

public void play(){

System.out.println(name + " is Playing basketball");

}

}

public class Main{

public static void main(String[] args){

Scanner inp = new Scanner(System.in);

String a= inp.nextLine();

football f= new football(a);

f.play();

String b=inp.nextLine();

volleyball v=new volleyball(b);

v.play();

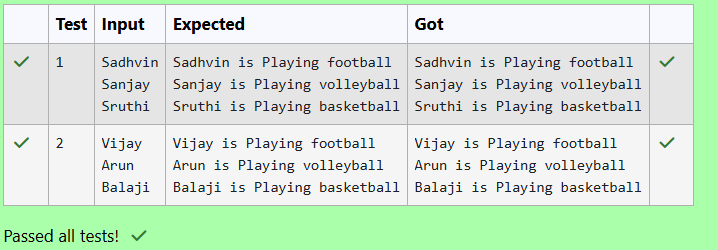
String c=inp.nextLine();

basketball b1= new basketball(c);

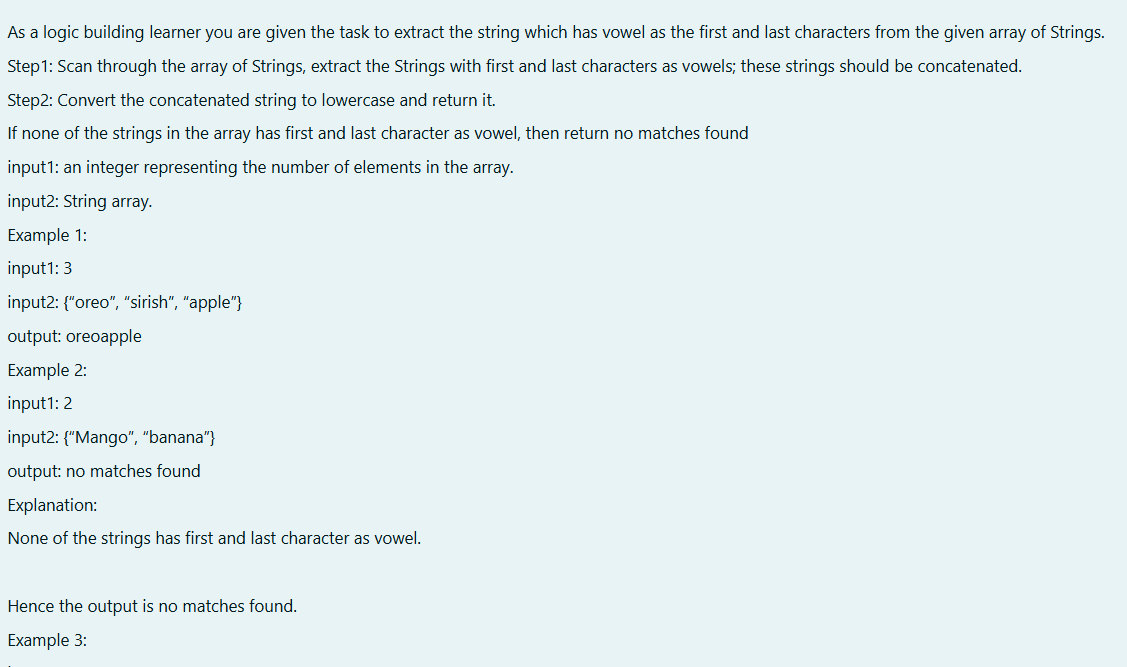
b1.play();

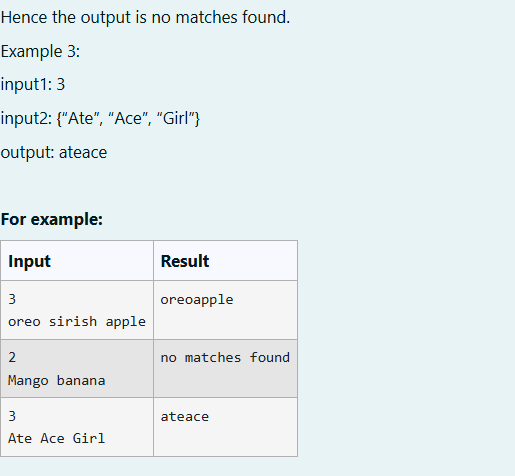
}

}



WEEK 8





import java.util.\*;

public class hello

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int k=0;

String arr[]=new String[n];

for(int i=0;i<n;i++)

{

arr[i]=sc.next();

arr[i]=arr[i].toLowerCase();

char ch=arr[i].charAt(0);

if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u')

{

int z=arr[i].length();

char x=arr[i].charAt(z-1);

if (x=='a' || x=='e' || x=='i' || x=='o'|| x=='u')

{

k=1;

System.out.print(arr[i]);

}

}

}

if(k==0)

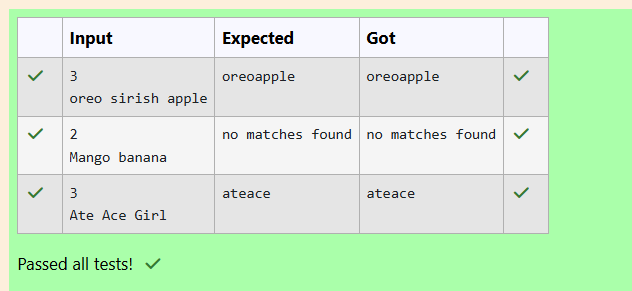
{

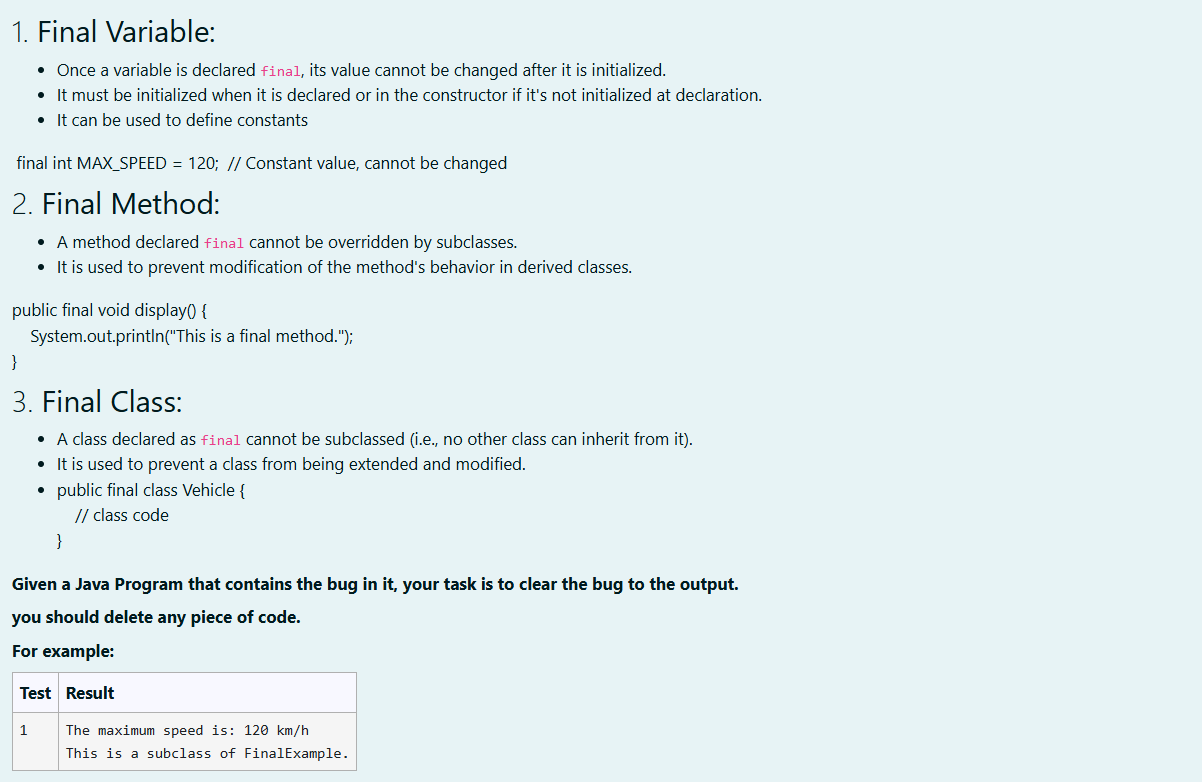
System.out.println("no matches found");

}

}

}





class FinalExample

{

int maxSpeed = 120;

public final void displayMaxSpeed()

{

System.out.println("The maximum speed is: " + maxSpeed + " km/h");

}

}

class SubClass extends FinalExample

{

public void showDetails()

{

System.out.println("This is a subclass of FinalExample.");

}

}

class prog

{

public static void main(String[] args)

{

FinalExample obj = new FinalExample();

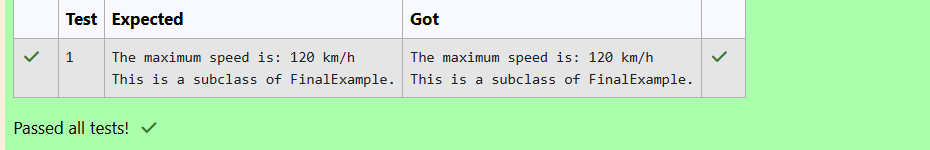
obj.displayMaxSpeed();

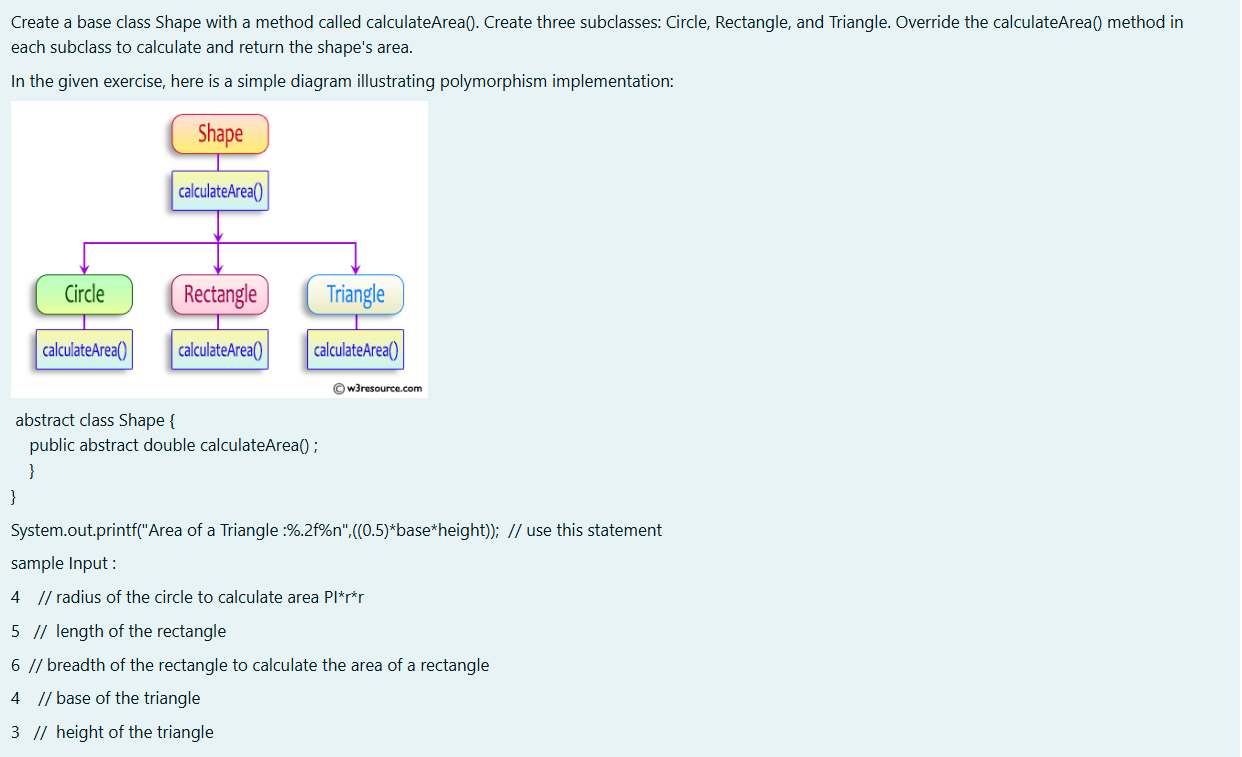
SubClass subObj = new SubClass();

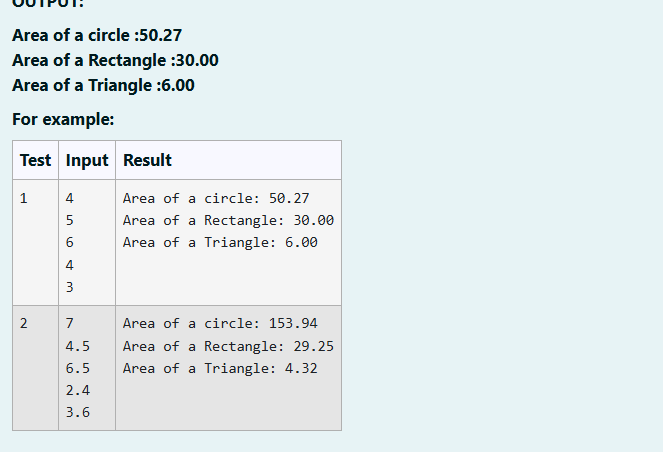
subObj.showDetails();

}

}







import java.util.\*;

abstract class s

{

public abstract double calculateArea();

}

class c extends s

{

double r;

c(double r)

{

this.r=r;

}

public double calculateArea()

{

double a=Math.PI\*r\*r;

System.out.printf("Area of a circle: %.2f\n",a);

return a;

}

}

class r extends s

{

double l;

double b;

r(double l,double b)

{

this.l=l;

this.b=b;

}

public double calculateArea()

{

double a=l\*b;

System.out.printf("Area of a Rectangle: %.2f\n",a);

return a;

}

}

class t extends s

{

double b;

double h;

t(double b,double h)

{

this.b=b;

this.h=h;

}

public double calculateArea()

{

double a=b\*h\*0.5;

System.out.printf("Area of a Triangle: %.2f\n",a);

return a;

}

}

public class hello

{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

double r1=sc.nextDouble();

c c1=new c(r1);

double l1=sc.nextDouble();

double b1=sc.nextDouble();

r r2=new r(l1,b1);

double b2=sc.nextDouble();

double h2=sc.nextDouble();

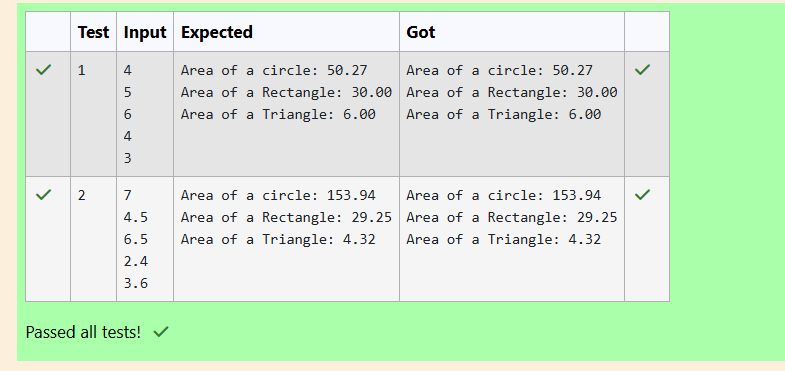
t t1=new t(b2,h2);

c1.calculateArea();

r2.calculateArea();

t1.calculateArea();

}}



WEEK 9



import java.util.Scanner;

public class ExceptionHandlingExample {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

int[] array = new int[n];

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

array[i] = scanner.nextInt();

}

try {

int result = array[0] / array[1];

} catch (ArithmeticException e) {

System.out.println("java.lang.ArithmeticException: " + e.getMessage());

}

try {

int outOfBoundsAccess = array[3];

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("java.lang.ArrayIndexOutOfBoundsException: " + e.getMessage());

} finally {

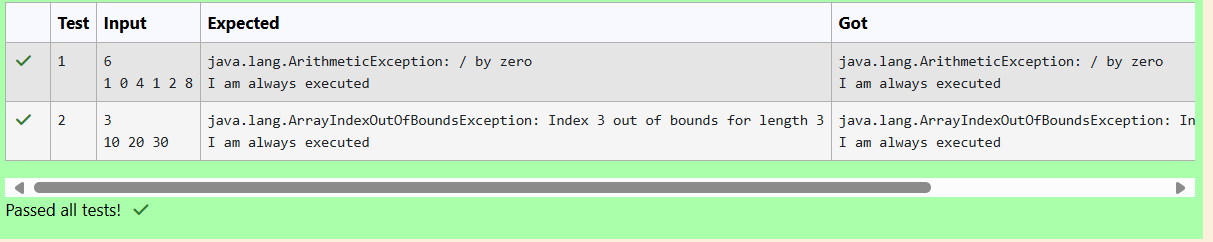
System.out.println("I am always executed");

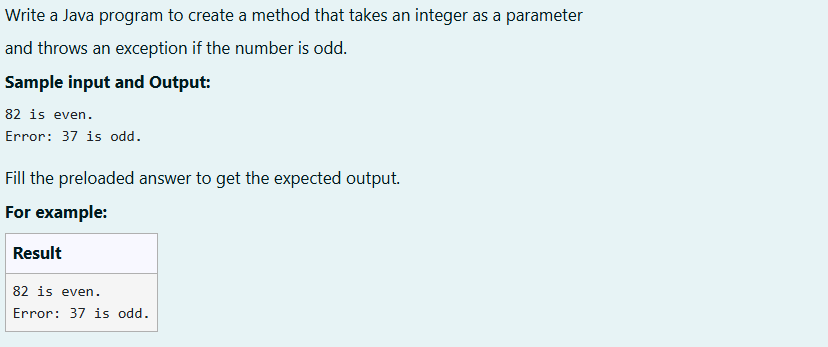
}

scanner.close();

}

}





class prog {

public static void main(String[] args) {

int n = 82;

trynumber(n);

n = 37;

// call the trynumber(n);

trynumber(n);

}

public static void trynumber(int n) {

try {

//call the checkEvenNumber()

checkEvenNumber(n);

System.out.println(n + " is even.");

} catch (Exception e) {

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

}

}

public static void checkEvenNumber(int number)throws Exception {

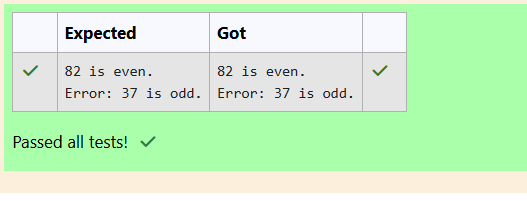
if (number % 2 != 0) {

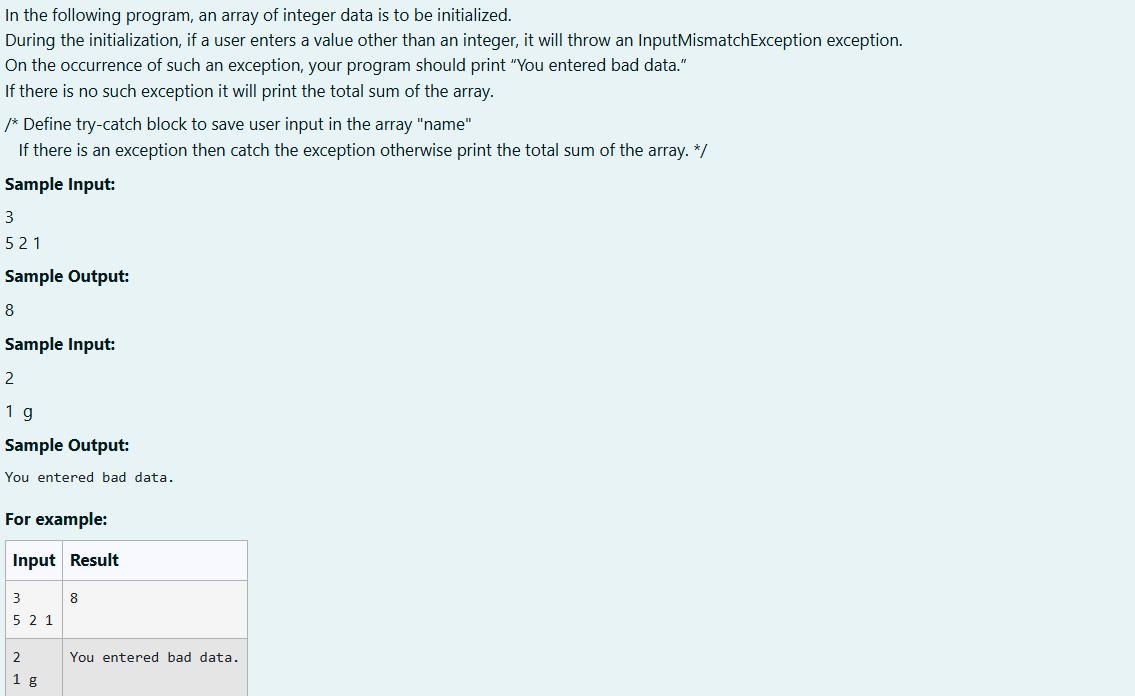
throw new Exception("Error: "+number + " is odd.");

}

}

}





import java.util.Scanner;

import java.util.InputMismatchException;

class prog {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int length = sc.nextInt();

// create an array to save user input

int[] name = new int[length];

int sum=0;//save the total sum of the array.

/\* Define try-catch block to save user input in the array "name"

If there is an exception then catch the exception otherwise print

the total sum of the array. \*/

try

{

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

name[i] = sc.nextInt();

}

for (int num : name) {

sum += num;

}

System.out.println(sum);

}

catch(InputMismatchException e )

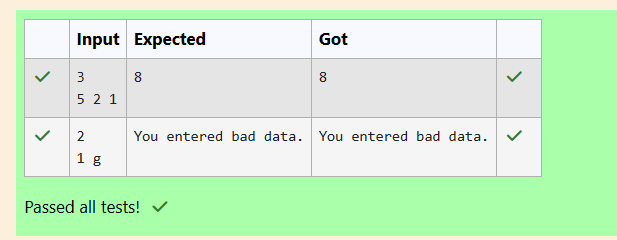
{

System.out.println("You entered bad data.");

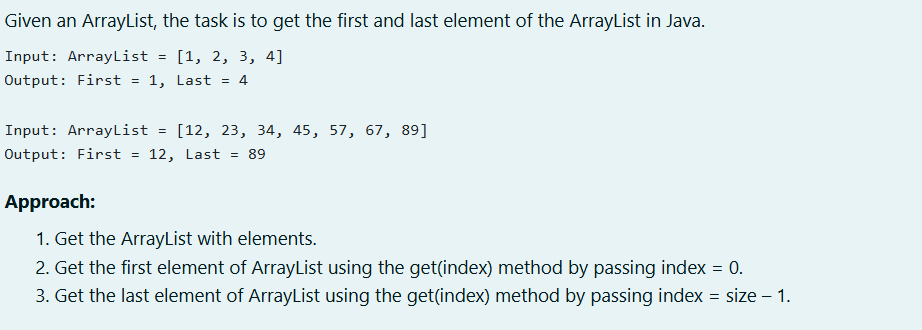
}finally {

sc.close();

}}}



WEEK 10



import java.util.\*;

public class main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

int n=obj.nextInt();

ArrayList<Integer> list=new ArrayList<>();

for(int i=0;i<n;i++)

{

list.add(obj.nextInt());

}

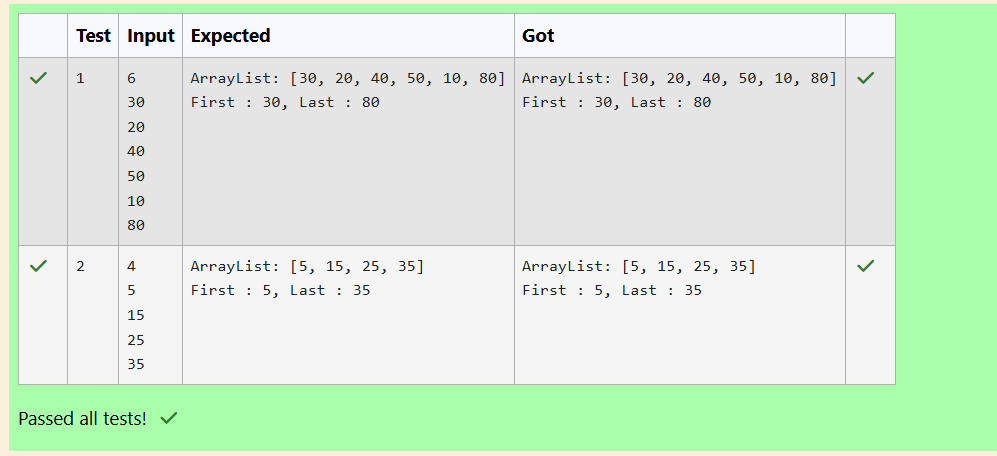
System.out.println("ArrayList: "+list);

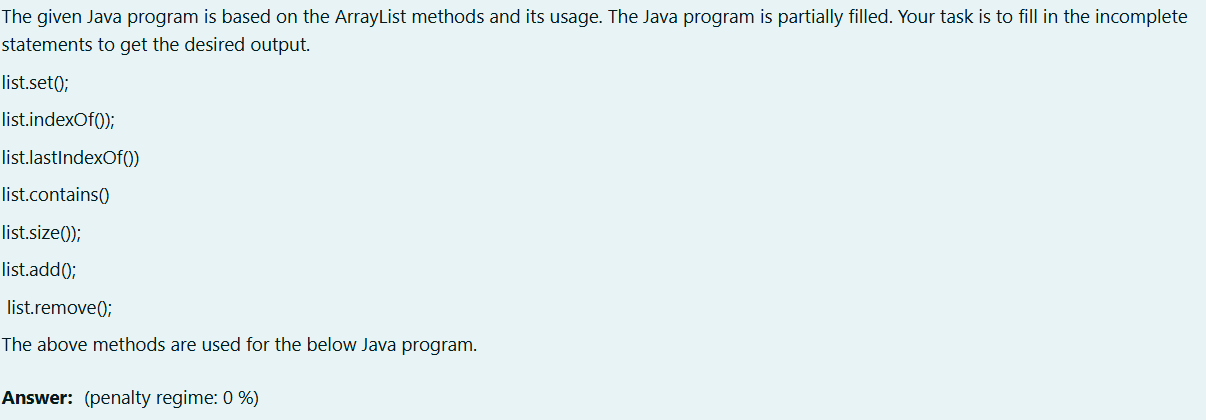
System.out.println("First : "+list.get(0)+", "+"Last : "+list.get(n-1));

obj.close();

}

}





import java.util.ArrayList;

import java.util.Scanner;

public class Prog {

public static void main(String[] args)

{

Scanner sc= new Scanner(System.in);

int n = sc.nextInt();

ArrayList<Integer> list = new ArrayList<Integer>();

for(int i = 0; i<n;i++)

list.add(sc.nextInt());

// printing initial value ArrayList

System.out.println("ArrayList: " + list);

//Replacing the element at index 1 with 100

list.set(1,100);

//Getting the index of first occurrence of 100

System.out.println("Index of 100 = "+ list.indexOf(100) );

//Getting the index of last occurrence of 100

System.out.println("LastIndex of 100 = "+ list.lastIndexOf(100) );

// Check whether 200 is in the list or not

System.out.println( list.contains(200) ); //Output : false

// Print ArrayList size

System.out.println("Size Of ArrayList = "+ list.size() );

//Inserting 500 at index 1

list.add(1,500) ; // code here

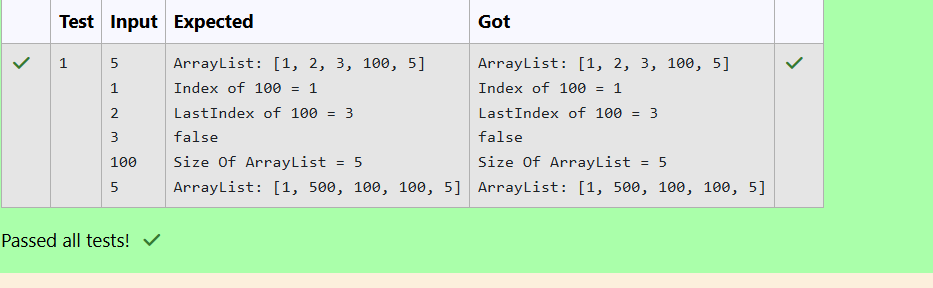
//Removing an element from position 3

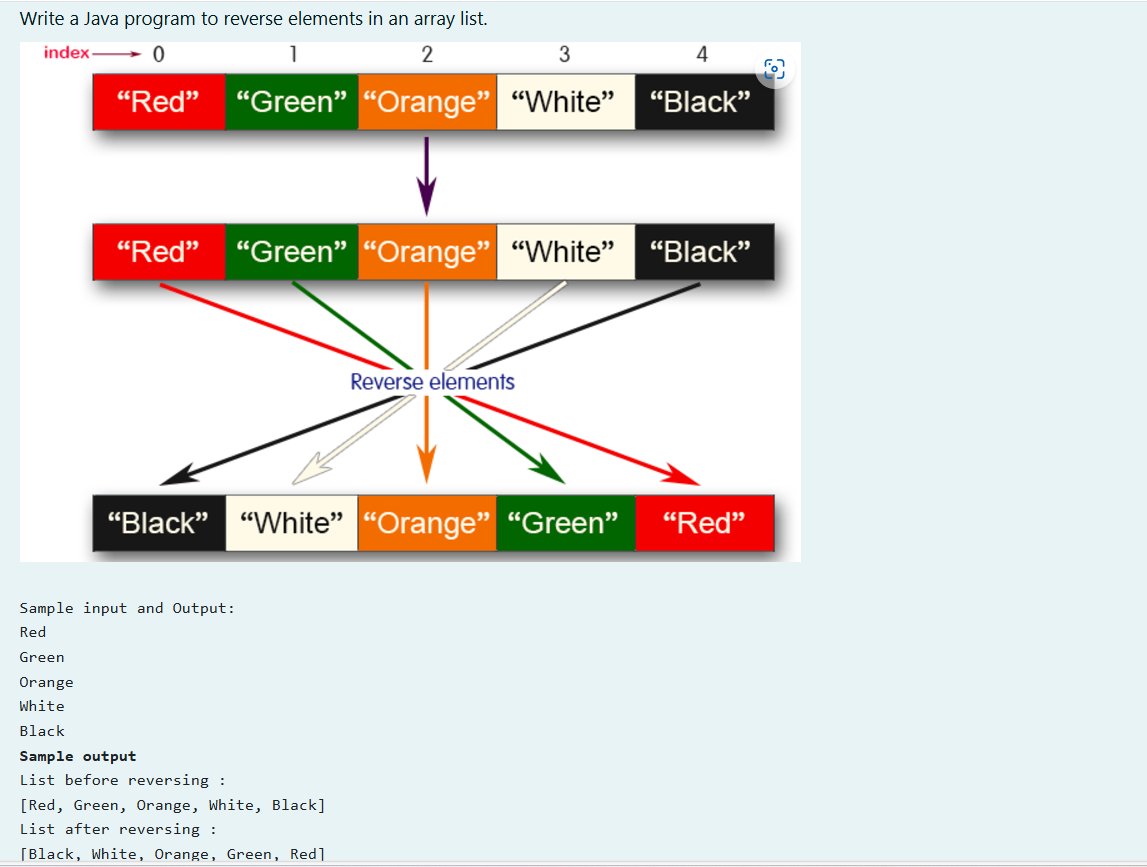
list.remove(list.get(3)); // code here

System.out.print("ArrayList: " + list);

}

}





//Reverse in arraylist

import java.util.\*;

public class Main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

int n=obj.nextInt();

ArrayList<String> list=new ArrayList<>();

for(int i=0;i<n;i++)

{

list.add(obj.next());

}

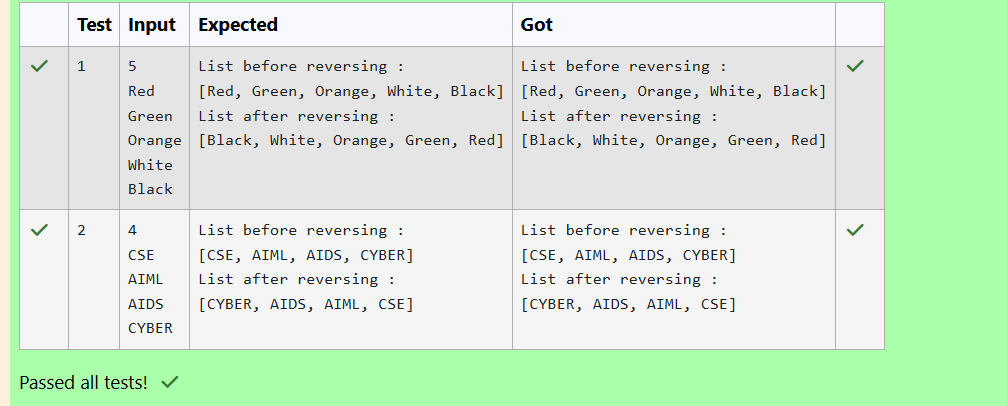
System.out.println("List before reversing :"+"\n"+list);

Collections.reverse(list);

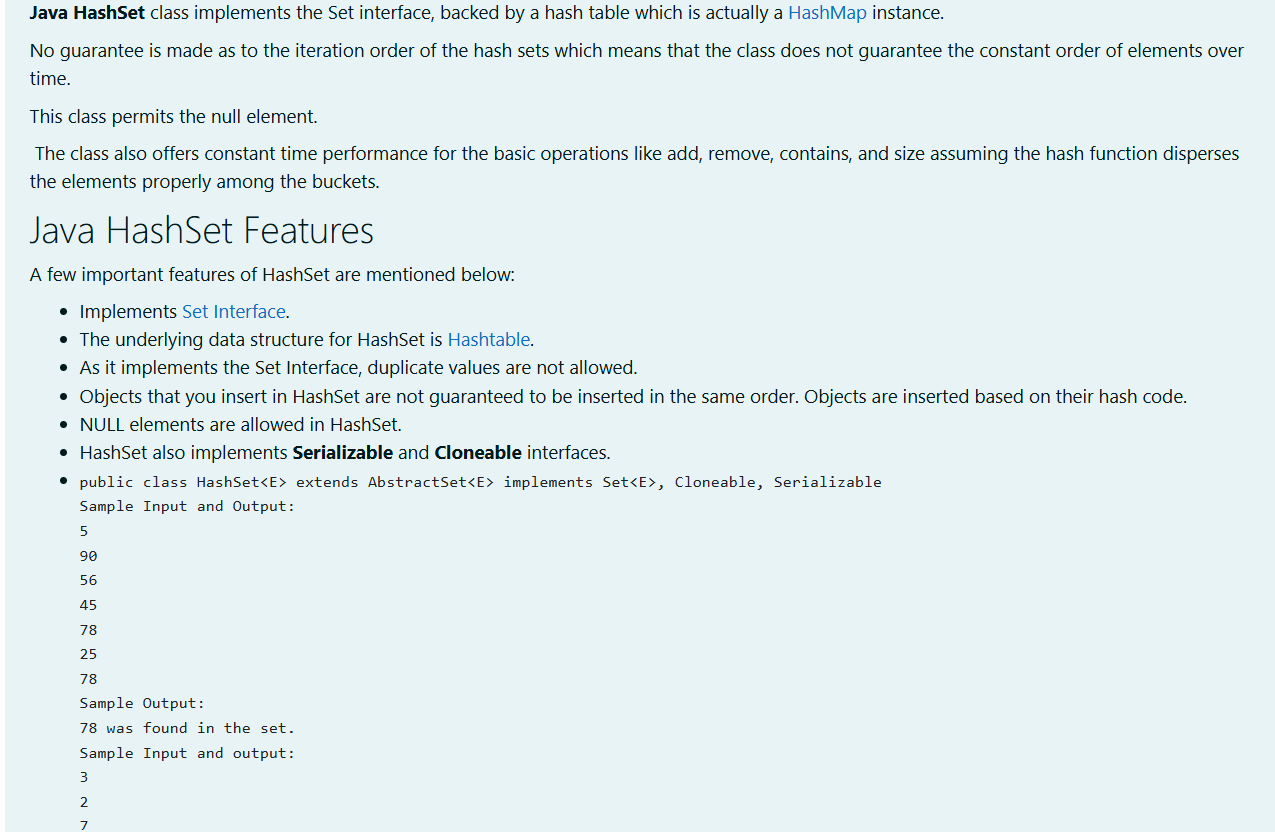
System.out.println("List after reversing :"+"\n"+list);

obj.close();

}

}  


WEEK 11



import java.util.HashSet;

import java.util.Scanner;

class prog {

public static void main(String[] args) {

Scanner sc= new Scanner(System.in);

int n = sc.nextInt();

// Create a HashSet object called numbers

HashSet<Integer> numbers=new HashSet<>();

// Add values to the set

for(int i=0;i<n;i++)

numbers.add(sc.nextInt());

int skey=sc.nextInt();

// Show which numbers between 1 and 10 are in the set

if(numbers.contains(skey))

System.out.println( skey + " was found in the set.");

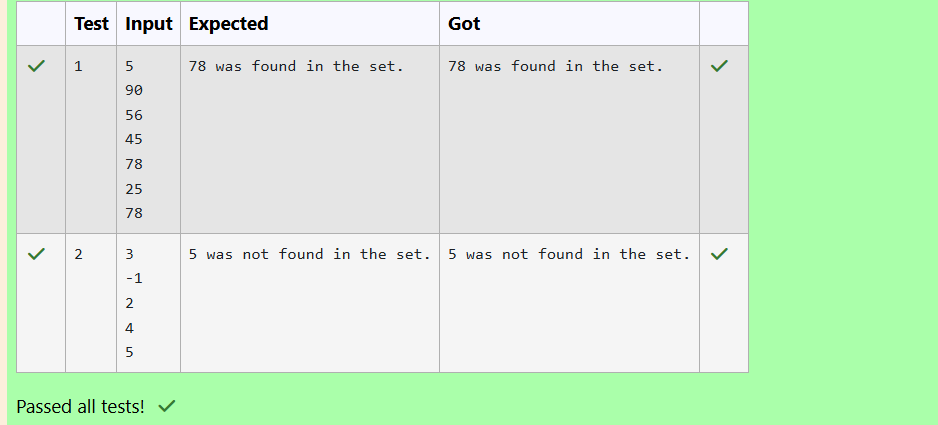
else {

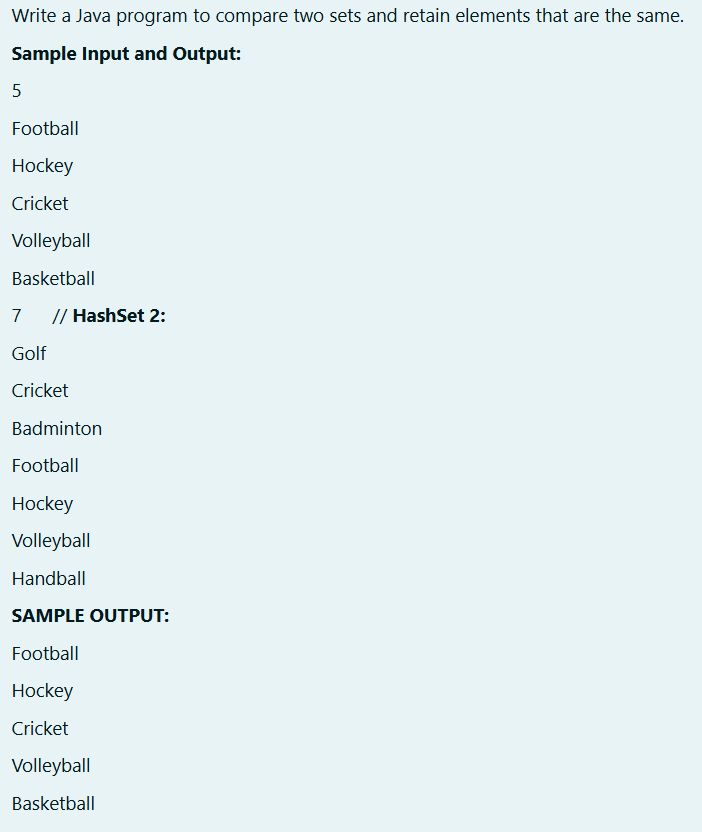
System.out.println(skey + " was not found in the set.");

}

}

}





import java.util.\*;

public class Main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

Set<String> h1=new HashSet<>();

Set<String> h2=new HashSet<>();

int n1=obj.nextInt();

for(int i=0;i<n1;i++)

{

h1.add(obj.next());

}

int n2=obj.nextInt();

for(int i=0;i<n2;i++)

{

h2.add(obj.next());

}

h1.retainAll(h2);

for(String s:h1)

{

System.out.println(s);

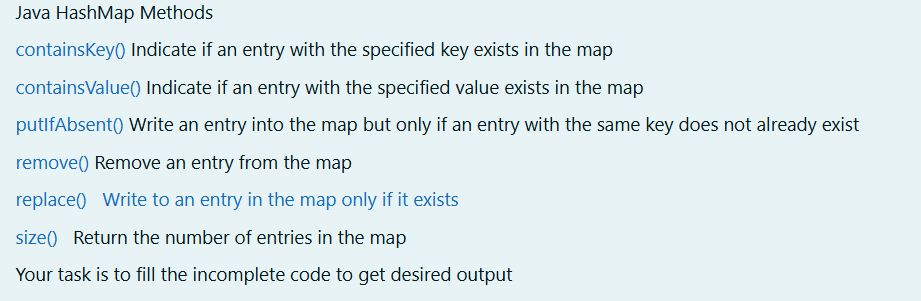
}

obj.close();

}

}





import java.util.HashMap;

import java.util.Map.Entry;

import java.util.Set;

import java.util.Scanner;

class prog

{

public static void main(String[] args)

{

//Creating HashMap with default initial capacity and load factor

HashMap<String, Integer> map = new HashMap<String, Integer>();

String name;

int num;

Scanner sc= new Scanner(System.in);

int n=sc.nextInt();

for(int i =0;i<n;i++)

{

name=sc.next();

num= sc.nextInt();

map.put(name,num);

}

//Printing key-value pairs

Set<Entry<String, Integer>> entrySet = map.entrySet();

for (Entry<String, Integer> entry : entrySet)

{

System.out.println(entry.getKey()+" : "+entry.getValue());

}

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

//Creating another HashMap

HashMap<String, Integer> anotherMap = new HashMap<String, Integer>();

//Inserting key-value pairs to anotherMap using put() method

anotherMap.put("SIX", 6);

anotherMap.put("SEVEN", 7);

//Inserting key-value pairs of map to anotherMap using putAll() method

anotherMap.putAll (map); // code here

//Printing key-value pairs of anotherMap

entrySet = anotherMap.entrySet();

for (Entry<String, Integer> entry : entrySet)

{

System.out.println(entry.getKey()+" : "+entry.getValue());

}

//Adds key-value pair 'FIVE-5' only if it is not present in map

map.putIfAbsent("FIVE", 5);

//Retrieving a value associated with key 'TWO'

int value = map.get("TWO");

System.out.println(value);

//Checking whether key 'ONE' exist in map

System.out.println( map.containsKey("ONE") );

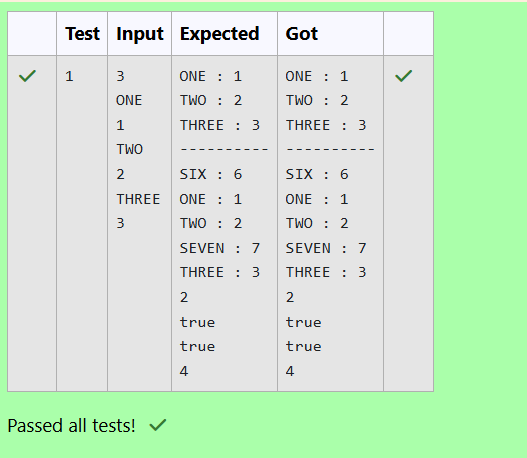
//Checking whether value '3' exist in map

System.out.println( map.containsValue(3) );

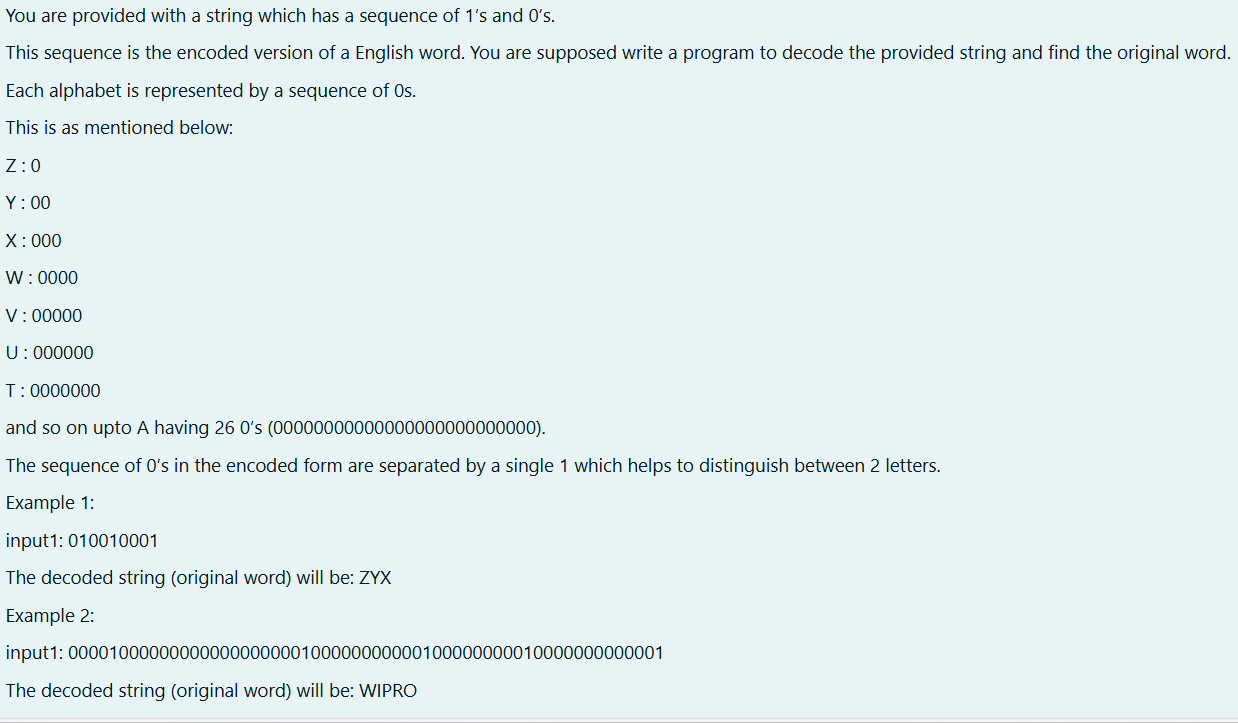
//Retrieving the number of key-value pairs present in map

System.out.println( map.size() );

}

}  


WEEK 12



import java.util.Scanner;

public class main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

String s=obj.next();

String[] arr=s.split("1");

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

{

int c=91-(arr[i].length());

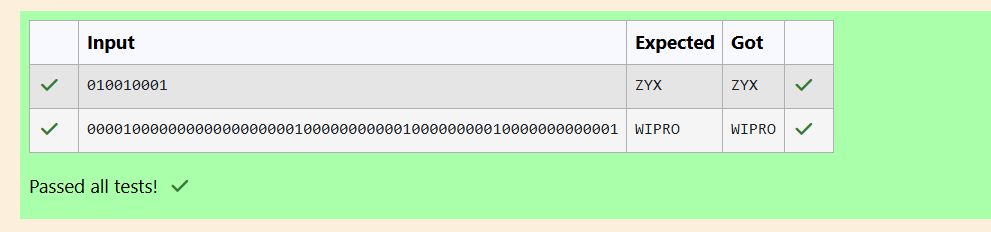
System.out.print((char)c);

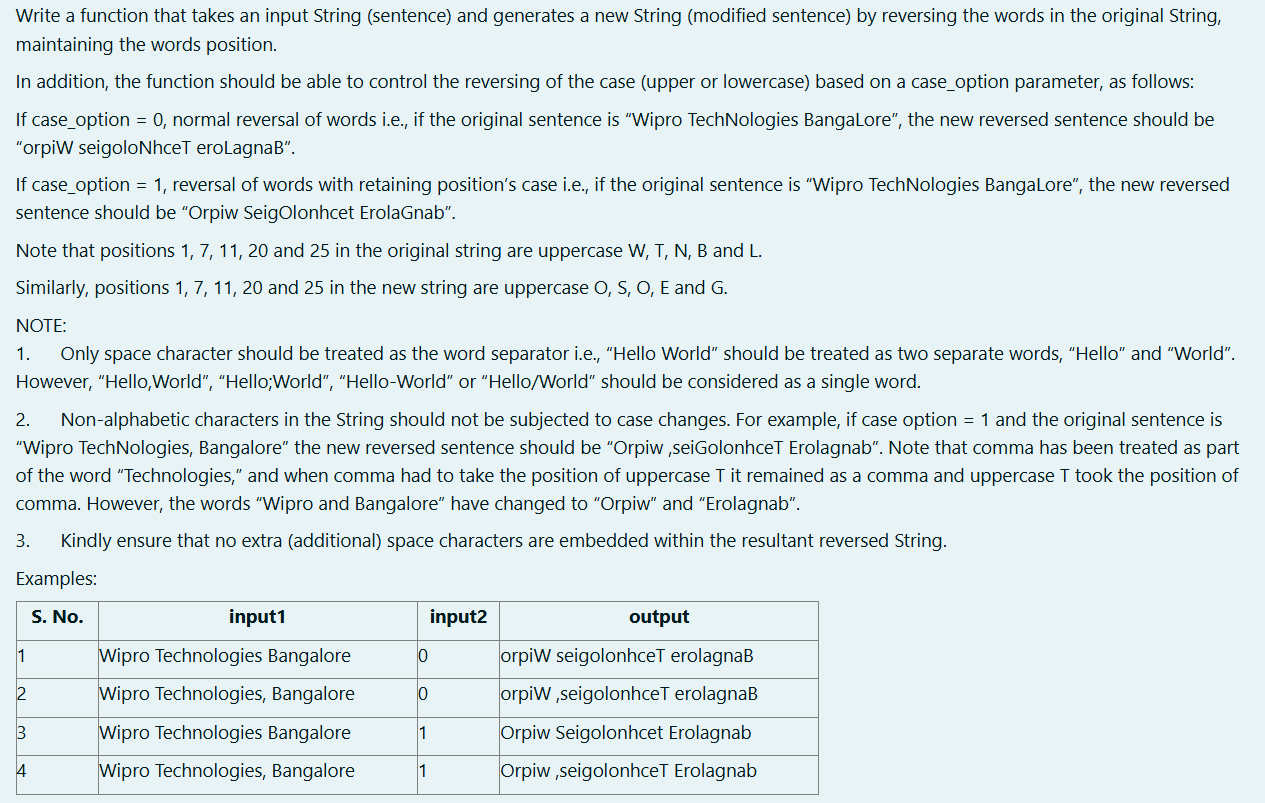
}

obj.close();

}

}





//case and reverse

import java.util.\*;

public class main

{

public static void main(String[] args)

{

Scanner obj=new Scanner(System.in);

String s=obj.nextLine();

int c=obj.nextInt();

String[] arr=s.split(" ");

if(c==0)

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

StringBuffer sb=new StringBuffer(arr[i]);

System.out.print(sb.reverse()+" ");}

}

else

{

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

ArrayList<Integer> list=new ArrayList<>();

for(int j=0;j<arr[i].length();j++)

{

if(Character.isUpperCase(arr[i].charAt(j)))

{

list.add(j);

}

}

String help=arr[i];

StringBuffer sb=new StringBuffer(arr[i].toLowerCase());

char[] carr=sb.reverse().toString().toCharArray();

for(int j=0;j<carr.length;j++)

{

if(Character.isLetter(carr[j]) && list.contains(j))

{

System.out.print(Character.toUpperCase(carr[j]));

}

else if(Character.isLetter(arr[i].charAt(j))==false)

{

//System.out.print(Character.toUpperCase(carr[j]));

System.out.print(help.charAt(help.length()-1-j));

}

else

{

System.out.print(carr[j]);

}

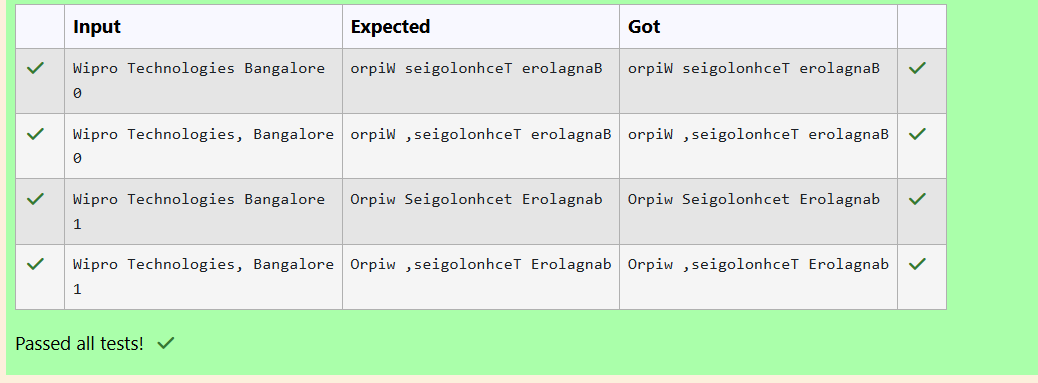
}

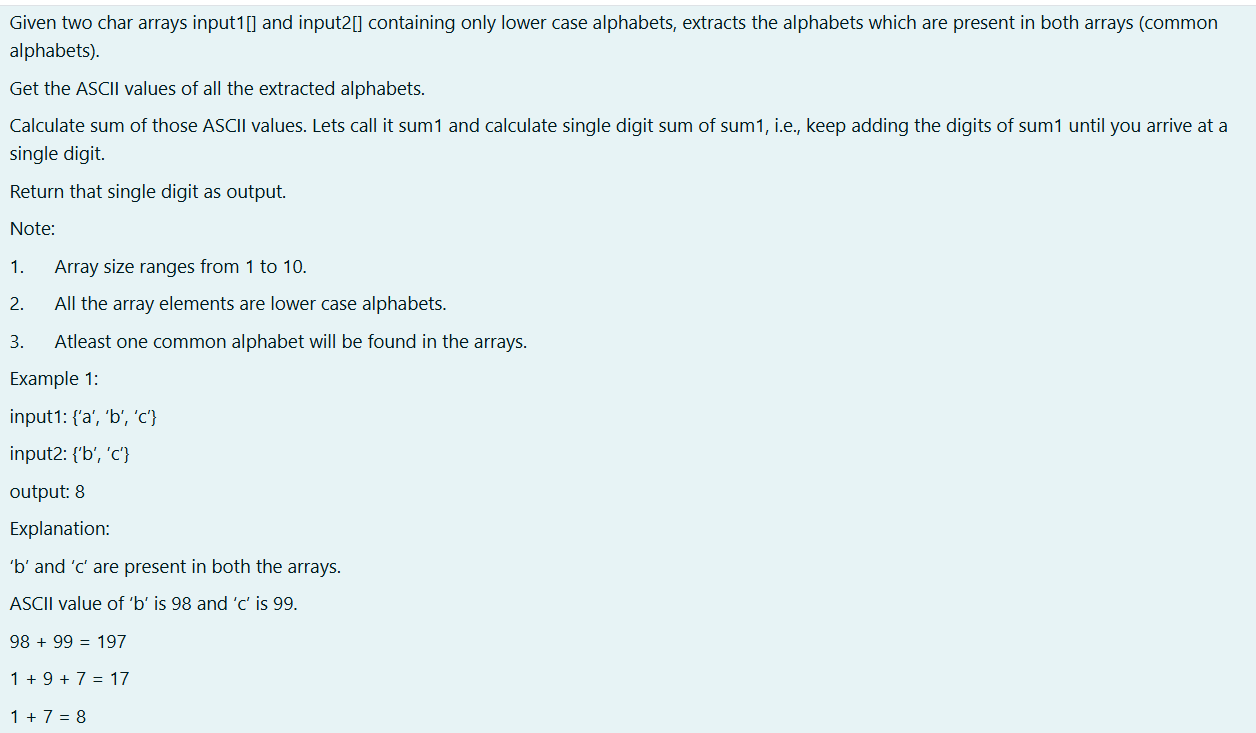
System.out.print(" ");}

}

}

}





import java.util.HashSet;

public class CommonAsciiSum {

public static int singleDigitSum(char[] input1, char[] input2) {

// Use HashSet to store unique characters from both arrays

HashSet<Character> set1 = new HashSet<>();

HashSet<Character> commonSet = new HashSet<>();

// Add all characters from input1 to set1

for (char c : input1) {

set1.add(c);

}

// Find common characters in input1 and input2

for (char c : input2) {

if (set1.contains(c)) {

commonSet.add(c);

}

}

// Calculate the sum of ASCII values of common characters

int sum1 = 0;

for (char c : commonSet) {

sum1 += (int) c;

}

// Calculate the single digit sum

while (sum1 >= 10) {

int tempSum = 0;

while (sum1 > 0) {

tempSum += sum1 % 10;

sum1 /= 10;

}

sum1 = tempSum;

}

return sum1;

}

public static void main(String[] args) {

char[] input1 = {'a', 'b', 'c'};

char[] input2 = {'b', 'c'};

System.out.println(singleDigitSum(input1, input2)); // Only prints the result

}

}

