RAJALAKSHMI ENGINEERING COLLEGE

RAJALAKSHMI NAGAR, THANDALAM – 602 105

RAJALAKSHMI ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

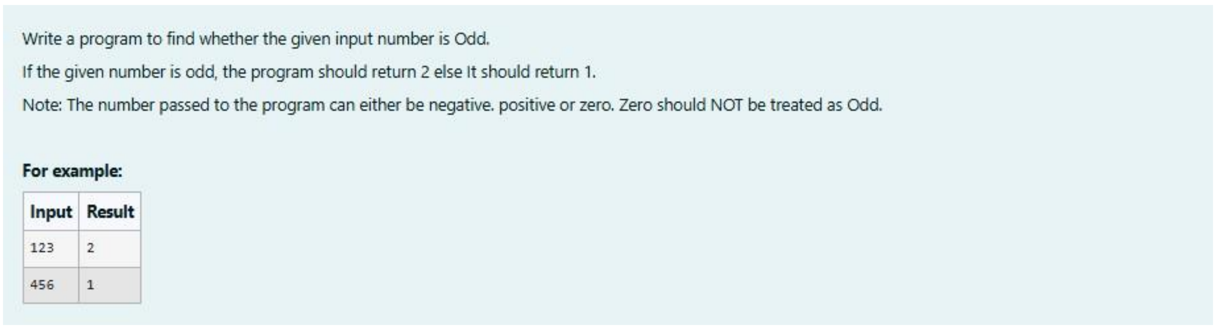
CS23333 OBJECT ORIENTED PROGRAMMING USING JAVA

Laboratory Record Notebook

NAME: ATHIENA RACHEL
YEAR/BRANCH/SECTION: 2nd YEAR/CSE/A
UNIVERSITY REGISTER NO:2116230701046
COLLEGE ROLL NO: 230701046
SEMESTER:3rd semester
ACADEMIC:2023-24

Lab-01-Java Architecture, Language Basics

1.


Write a program to find whether the given input number is Odd.

If the given number is odd, the program should return 2 else It should return 1.

Note: The number passed to the program can either be negative. positive or zero. Zero should NOT be treated as Odd.

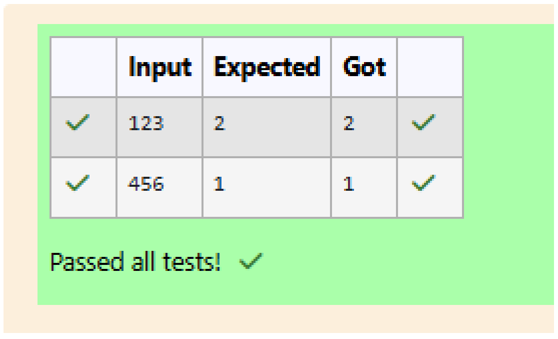
For example:

Input	Result
123	2
456	1

SOLUTION :

```
i
m
p
o
r
t

j
a
```

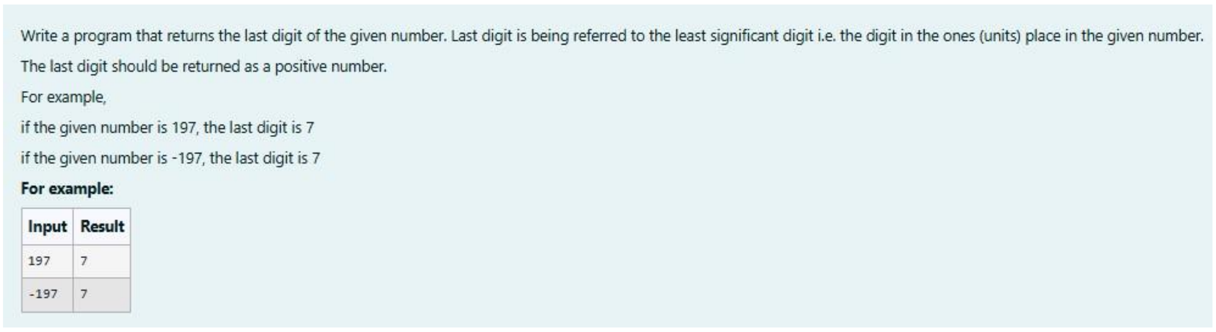
OUTPUT :



	Input	Expected	Got	
✓	123	2	2	✓
✓	456	1	1	✓

Passed all tests! ✓

2.



Write a program that returns the last digit of the given number. Last digit is being referred to the least significant digit i.e. the digit in the ones (units) place in the given number.

The last digit should be returned as a positive number.

For example,

if the given number is 197, the last digit is 7

if the given number is -197, the last digit is 7

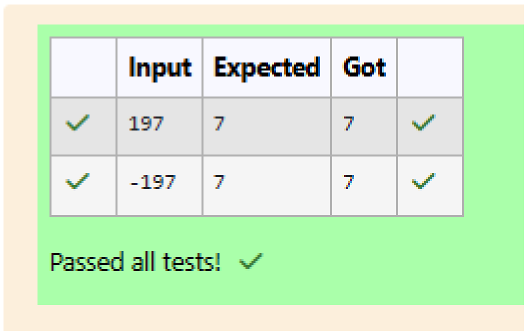
For example:

Input	Result
197	7
-197	7

SOLUTION :

```
i
m
p
o
r
t
```

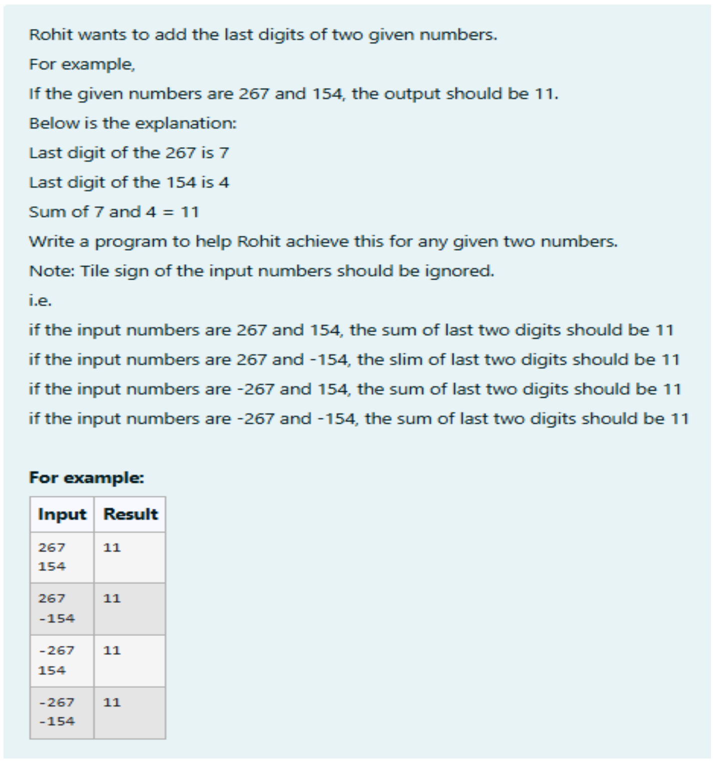
OUTPUT :



	Input	Expected	Got	
✓	197	7	7	✓
✓	-197	7	7	✓

Passed all tests! ✓

3.



Rohit wants to add the last digits of two given numbers.

For example,

If the given numbers are 267 and 154, the output should be 11.

Below is the explanation:

Last digit of the 267 is 7

Last digit of the 154 is 4

Sum of 7 and 4 = 11

Write a program to help Rohit achieve this for any given two numbers.

Note: Tile sign of the input numbers should be ignored.

i.e.

if the input numbers are 267 and 154, the sum of last two digits should be 11

if the input numbers are 267 and -154, the slim of last two digits should be 11

if the input numbers are -267 and 154, the sum of last two digits should be 11

if the input numbers are -267 and -154, the sum of last two digits should be 11

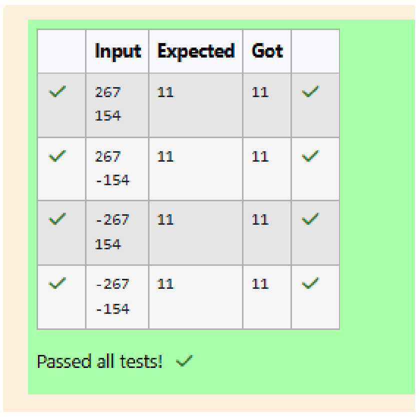
For example:

Input	Result
267 154	11
267 -154	11
-267 154	11
-267 -154	11

SOLUTION :

i
m
p
o
r
t

OUTPUT:

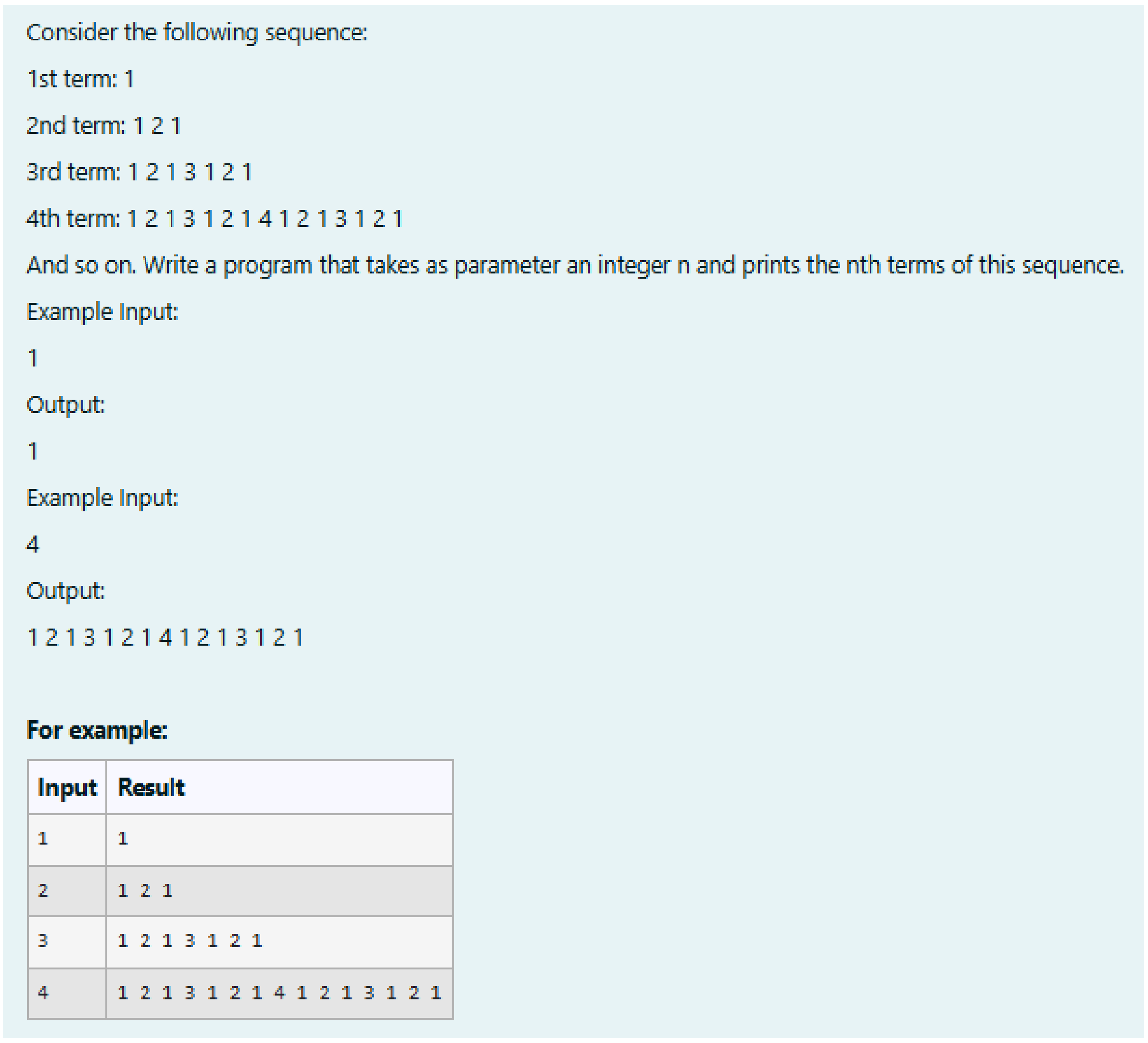


	Input	Expected	Got	
✓	267 154	11	11	✓
✓	267 -154	11	11	✓
✓	-267 154	11	11	✓
✓	-267 -154	11	11	✓

Passed all tests! ✓

Lab-02-Flow Control Statements

1.



Consider the following sequence:

1st term: 1

2nd term: 1 2 1

3rd term: 1 2 1 3 1 2 1

4th term: 1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

And so on. Write a program that takes as parameter an integer n and prints the nth terms of this sequence.

Example Input:

1

Output:

1

Example Input:

4

Output:

1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

For example:

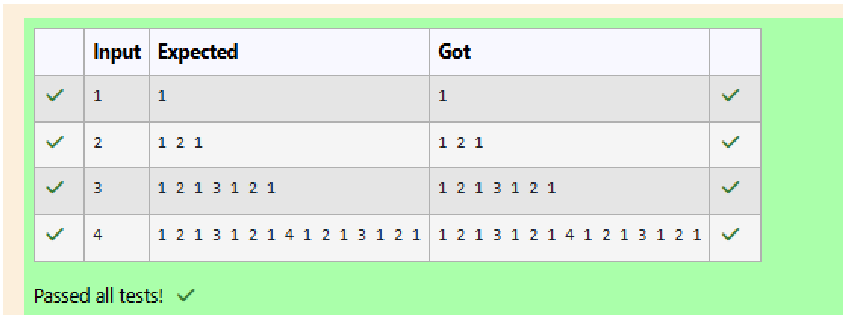
Input	Result
1	1
2	1 2 1
3	1 2 1 3 1 2 1
4	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1

SOLUTION :

```
import java.util.Scanner;
public class SequenceGenerator{
   public
      stati
      c
      voi
      d
      mai
      n(St
```

c
u
r

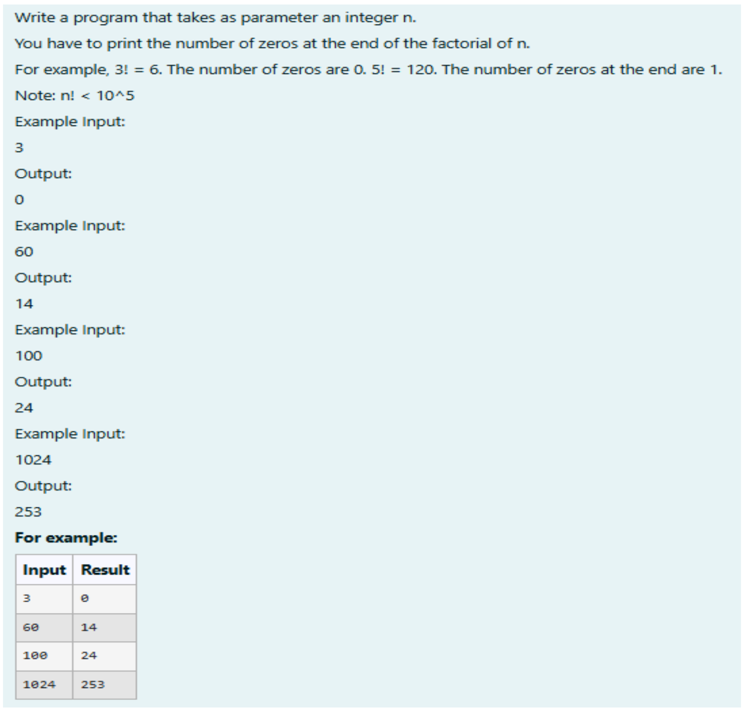
OUTPUT :



	Input	Expected	Got	
✓	1	1	1	✓
✓	2	1 2 1	1 2 1	✓
✓	3	1 2 1 3 1 2 1	1 2 1 3 1 2 1	✓
✓	4	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1	1 2 1 3 1 2 1 4 1 2 1 3 1 2 1	✓

Passed all tests! ✓

2.



Write a program that takes as parameter an integer n.

You have to print the number of zeros at the end of the factorial of n.

For example, 3! = 6. The number of zeros are 0. 5! = 120. The number of zeros at the end are 1.

Note: n! < 10^5

Example Input:

3

Output:

0

Example Input:

60

Output:

14

Example Input:

100

Output:

24

Example Input:

1024

Output:

253

For example:

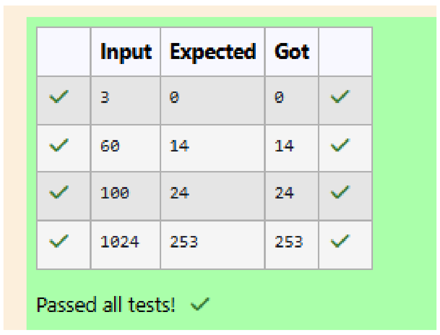
Input	Result
3	0
60	14
100	24
1024	253

SOLUTION :

// Java
progra
m to
count
trailin
g 0s in
n!

```
// Initialize result

int count=0;
// Keep dividing n by powers
// of 5 and update count
fo
   r

   (
   i
   n
   t
```

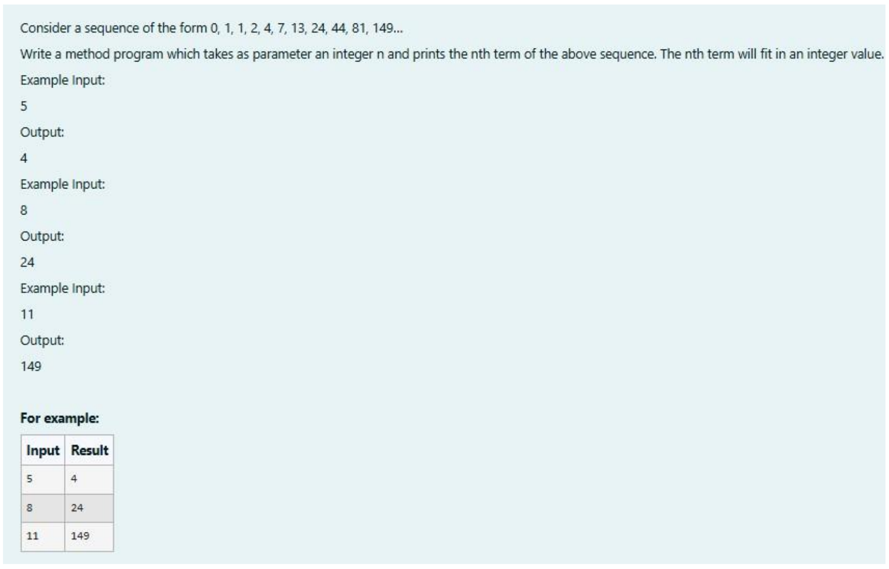
OUTPUT :



	Input	Expected	Got	
✓	3	0	0	✓
✓	60	14	14	✓
✓	100	24	24	✓
✓	1024	253	253	✓

Passed all tests! ✓

3.



Consider a sequence of the form 0, 1, 1, 2, 4, 7, 13, 24, 44, 81, 149...

Write a method program which takes as parameter an integer n and prints the nth term of the above sequence. The nth term will fit in an integer value.

Example Input:

5

Output:

4

Example Input:

8

Output:

24

Example Input:

11

Output:

149

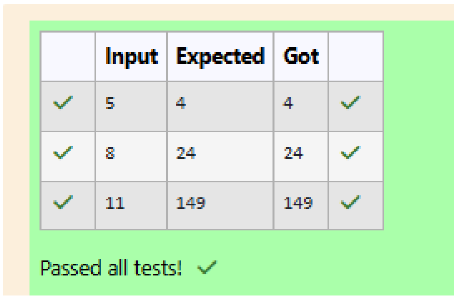
For example:

Input	Result
5	4
8	24
11	149

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
t
i
l
.
S
c
a
n
n
e
r
;
```

OUTPUT :

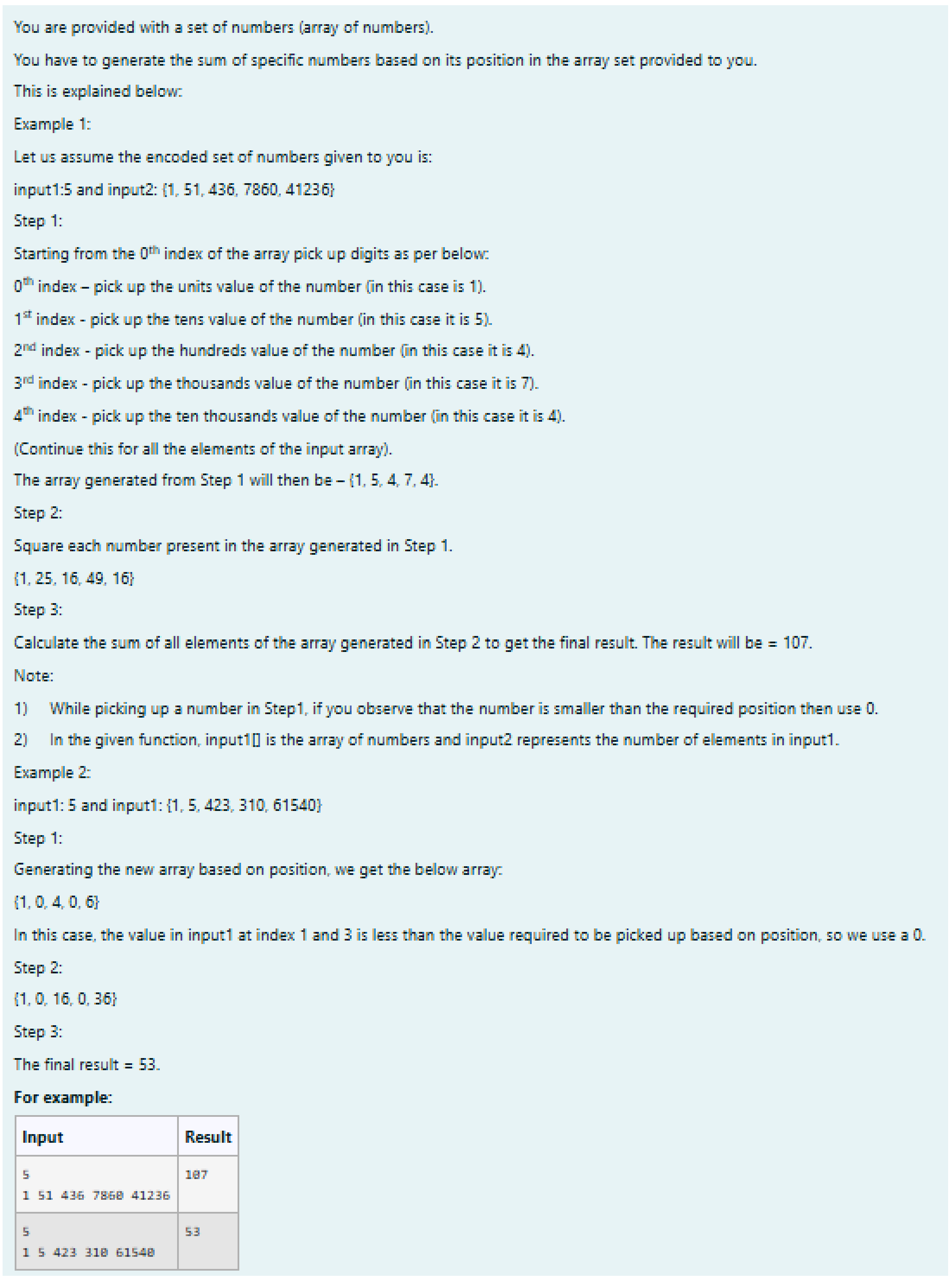


	Input	Expected	Got	
✓	5	4	4	✓
✓	8	24	24	✓
✓	11	149	149	✓

Passed all tests! ✓

Lab-03-Arrays

1.



You are provided with a set of numbers (array of numbers).

You have to generate the sum of specific numbers based on its position in the array set provided to you.

This is explained below:

Example 1:

Let us assume the encoded set of numbers given to you is:

input1:5 and input2: {1, 51, 436, 7860, 41236}

Step 1:

Starting from the 0th index of the array pick up digits as per below:

0th index – pick up the units value of the number (in this case is 1).

1st index - pick up the tens value of the number (in this case it is 5).

2nd index - pick up the hundreds value of the number (in this case it is 4).

3rd index - pick up the thousands value of the number (in this case it is 7).

4th index - pick up the ten thousands value of the number (in this case it is 4).

(Continue this for all the elements of the input array).

The array generated from Step 1 will then be – {1, 5, 4, 7, 4}.

Step 2:

Square each number present in the array generated in Step 1.

{1, 25, 16, 49, 16}

Step 3:

Calculate the sum of all elements of the array generated in Step 2 to get the final result. The result will be = 107.

Note:

1) While picking up a number in Step1, if you observe that the number is smaller than the required position then use 0.

2) In the given function, input1[] is the array of numbers and input2 represents the number of elements in input1.

Example 2:

input1: 5 and input1: {1, 5, 423, 310, 61540}

Step 1:

Generating the new array based on position, we get the below array:

{1, 0, 4, 0, 6}

In this case, the value in input1 at index 1 and 3 is less than the value required to be picked up based on position, so we use a 0.

Step 2:

{1, 0, 16, 0, 36}

Step 3:

The final result = 53.

For example:

Input	Result

`5`
`1 51 436 7860 41236`
	`107`

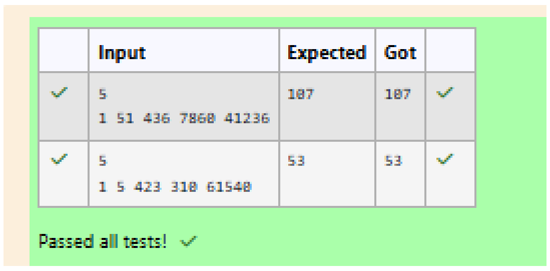
`5`
`1 5 423 310 61540`
	`53`

SOLUTION :

i
m

```
i
n
t

s
i
z
e

=
s
c
a
n
n
e
r
.
n
e
x
t
```

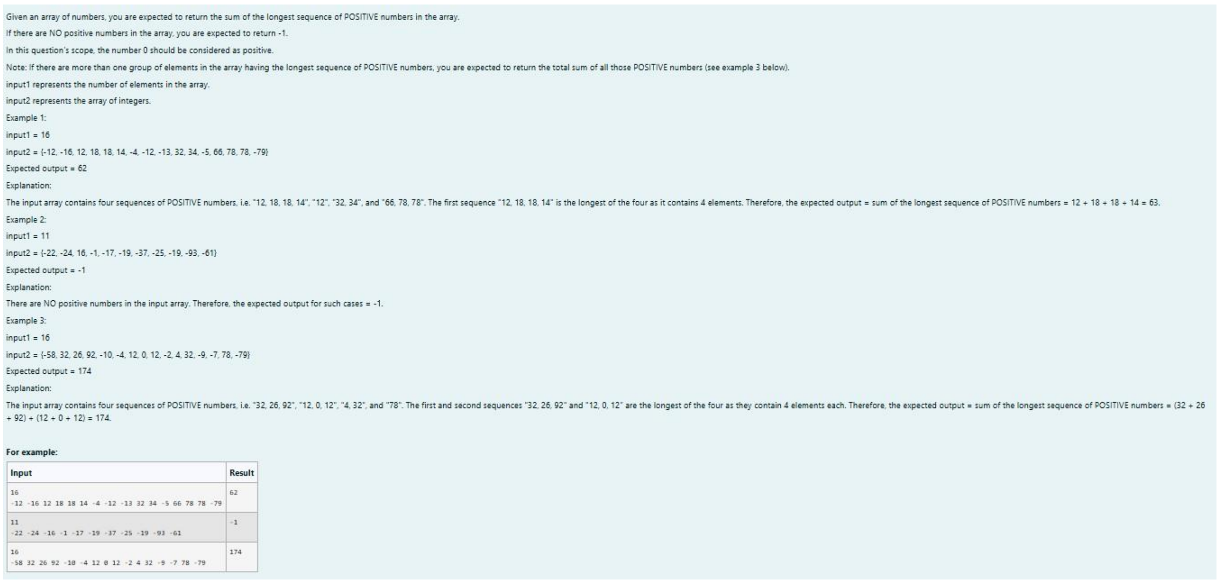
OUTPUT :



	Input	Expected	Got	
✓	5 1 51 436 7860 41236	107	107	✓
✓	5 1 5 423 310 61540	53	53	✓

Passed all tests! ✓

2.



Given an array of numbers, you are expected to return the sum of the longest sequence of POSITIVE numbers in the array.

If there are NO positive numbers in the array, you are expected to return -1.

In this question's scope, the number 0 should be considered as positive.

Note: If there are more than one group of elements in the array having the longest sequence of POSITIVE numbers, you are expected to return the total sum of all those POSITIVE numbers (see example 3 below).

input1 represents the number of elements in the array.

input2 represents the array of integers.

Example 1:

input1 = 16

input2 = {-12, -16, 12, 18, 18, 14, -4, -12, -13, 32, 34, -5, 66, 78, 78, -79}

Expected output = 62

Explanation:

The input array contains four sequences of POSITIVE numbers, i.e. "12, 18, 18, 14", "12", "32, 34", and "66, 78, 78". The first sequence "12, 18, 18, 14" is the longest of the four as it contains 4 elements. Therefore, the expected output = sum of the longest sequence of POSITIVE numbers = 12 + 18 + 18 + 14 = 63.

Example 2:

input1 = 11

input2 = {-22, -24, 16, -1, -17, -19, -37, -25, -19, -93, -61}

Expected output = -1

Explanation:

There are NO positive numbers in the input array. Therefore, the expected output for such cases = -1.

Example 3:

input1 = 16

input2 = {-58, 32, 26, 92, -10, -4, 12, 0, 12, -2, 4, 32, -9, -7, 78, -79}

Expected output = 174

Explanation:

The input array contains four sequences of POSITIVE numbers, i.e. "32, 26, 92", "12, 0, 12", "4, 32", and "78". The first and second sequences "32, 26, 92" and "12, 0, 12" are the longest of the four as they contain 4 elements each. Therefore, the expected output = sum of the longest sequence of POSITIVE numbers = (32 + 26 + 92) + (12 + 0 + 12) = 174.

For example:

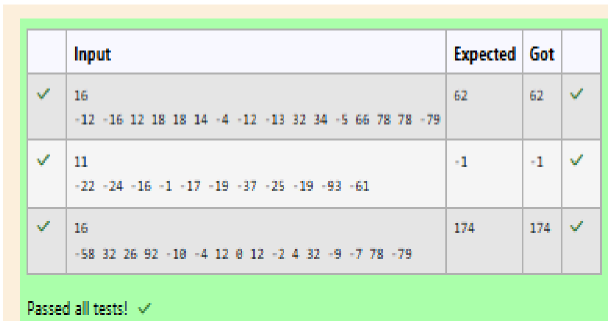
Input	Result
16 -12 -16 12 18 18 14 -4 -12 -13 32 34 -5 66 78 78 -79	62
11 -22 -24 -16 -1 -17 -19 -37 -25 -19 -93 -61	-1
16 -58 32 26 92 -10 -4 12 0 12 -2 4 32 -9 -7 78 -79	174

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
t
i
l
.
S
c
```

```
    }
}
```

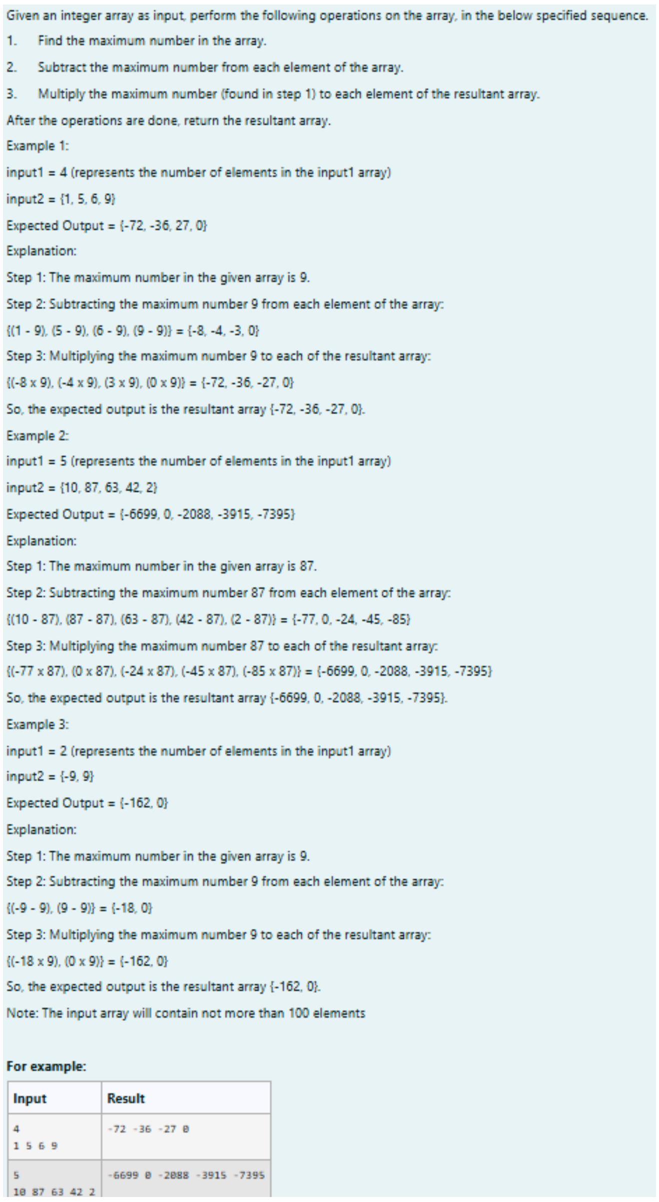
OUTPUT :



	Input	Expected	Got	
✓	16 -12 -16 12 18 18 14 -4 -12 -13 32 34 -5 66 78 78 -79	62	62	✓
✓	11 -22 -24 -16 -1 -17 -19 -37 -25 -19 -93 -61	-1	-1	✓
✓	16 -58 32 26 92 -10 -4 12 0 12 -2 4 32 -9 -7 78 -79	174	174	✓

Passed all tests! ✓

3.



Given an integer array as input, perform the following operations on the array, in the below specified sequence.

1. Find the maximum number in the array.
2. Subtract the maximum number from each element of the array.
3. Multiply the maximum number (found in step 1) to each element of the resultant array.

After the operations are done, return the resultant array.

Example 1:

input1 = 4 (represents the number of elements in the input1 array)

input2 = {1, 5, 6, 9}

Expected Output = {-72, -36, 27, 0}

Explanation:

Step 1: The maximum number in the given array is 9.

Step 2: Subtracting the maximum number 9 from each element of the array:

{(1 - 9), (5 - 9), (6 - 9), (9 - 9)} = {-8, -4, -3, 0}

Step 3: Multiplying the maximum number 9 to each of the resultant array:

{(-8 x 9), (-4 x 9), (3 x 9), (0 x 9)} = {-72, -36, -27, 0}

So, the expected output is the resultant array {-72, -36, -27, 0}.

Example 2:

input1 = 5 (represents the number of elements in the input1 array)

input2 = {10, 87, 63, 42, 2}

Expected Output = {-6699, 0, -2088, -3915, -7395}

Explanation:

Step 1: The maximum number in the given array is 87.

Step 2: Subtracting the maximum number 87 from each element of the array:

{(10 - 87), (87 - 87), (63 - 87), (42 - 87), (2 - 87)} = {-77, 0, -24, -45, -85}

Step 3: Multiplying the maximum number 87 to each of the resultant array:

{(-77 x 87), (0 x 87), (-24 x 87), (-45 x 87), (-85 x 87)} = {-6699, 0, -2088, -3915, -7395}

So, the expected output is the resultant array {-6699, 0, -2088, -3915, -7395}.

Example 3:

input1 = 2 (represents the number of elements in the input1 array)

input2 = {-9, 9}

Expected Output = {-162, 0}

Explanation:

Step 1: The maximum number in the given array is 9.

Step 2: Subtracting the maximum number 9 from each element of the array:

{(-9 - 9), (9 - 9)} = {-18, 0}

Step 3: Multiplying the maximum number 9 to each of the resultant array:

{(-18 x 9), (0 x 9)} = {-162, 0}

So, the expected output is the resultant array {-162, 0}.

Note: The input array will contain not more than 100 elements

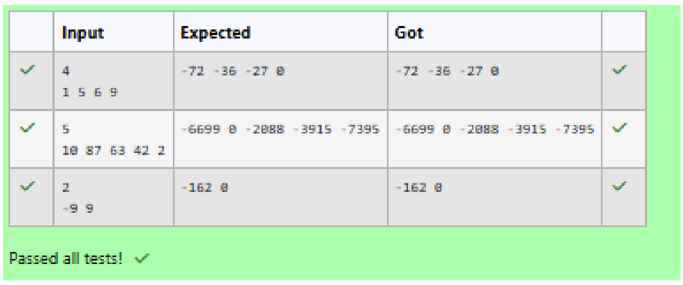
For example:

Input	Result
4 1 5 6 9	-72 -36 -27 0
5 10 87 63 42 2	-6699 0 -2088 -3915 -7395

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
t
i
l
.
S
```

OUTPUT :



	Input	Expected	Got	

✓	`4`
	`1 5 6 9`
		`-72 -36 -27 0`	`-72 -36 -27 0`	✓

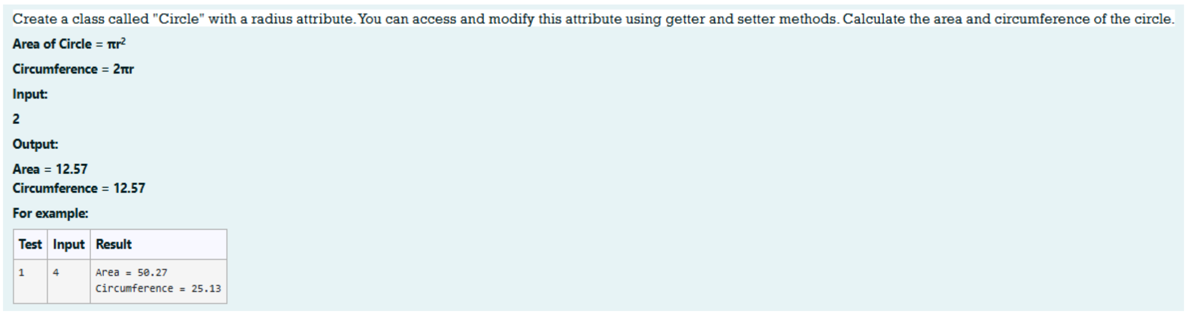
✓	`5`
	`10 87 63 42 2`
		`-6699 0 -2088 -3915 -7395`	`-6699 0 -2088 -3915 -7395`	✓

✓	`2`
	`-9 9`
		`-162 0`	`-162 0`	✓

Passed all tests! ✓

Lab-04-Classes and Objects

1.



Create a class called "Circle" with a radius attribute. You can access and modify this attribute using getter and setter methods. Calculate the area and circumference of the circle.

Area of Circle = πr^2

Circumference = $2\pi r$

Input:

2

Output:

Area = 12.57
Circumference = 12.57

For example:

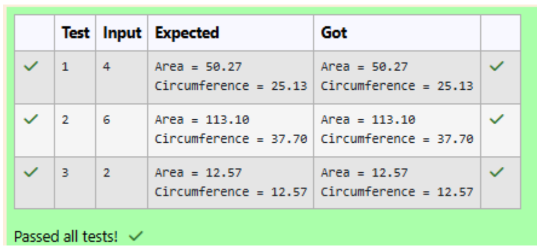
Test	Input	Result

1	4	`Area = 50.27`
		`Circumference = 25.13`

SOLUTION :

```
import java.io.*;
i
m
p
o
r
t

j
a
v
a
.
u
t
i
l
.
S
c
a
n
n
e
r
;
```

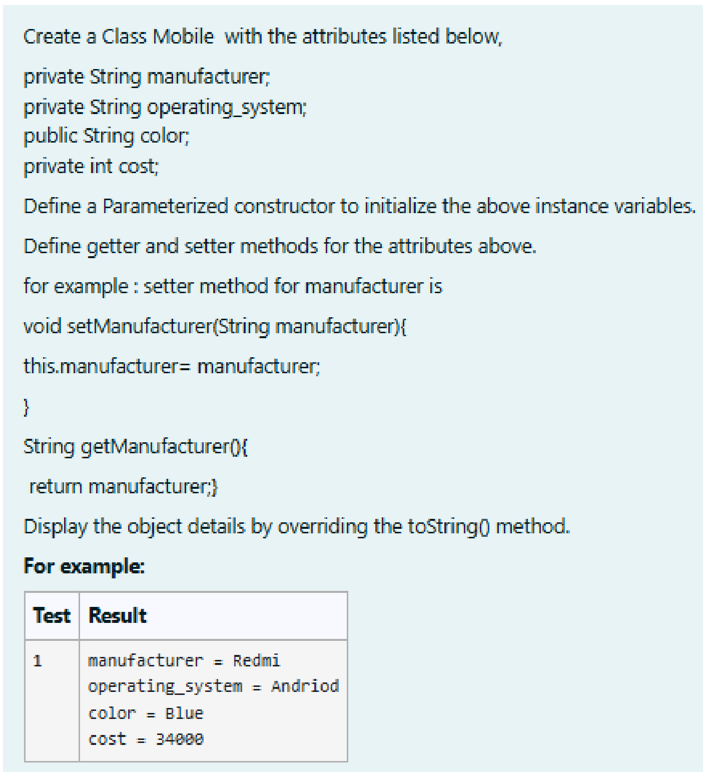
OUTPUT :



	Test	Input	Expected	Got	
✓	1	4	Area = 50.27 Circumference = 25.13	Area = 50.27 Circumference = 25.13	✓
✓	2	6	Area = 113.10 Circumference = 37.70	Area = 113.10 Circumference = 37.70	✓
✓	3	2	Area = 12.57 Circumference = 12.57	Area = 12.57 Circumference = 12.57	✓

Passed all tests! ✓

2.



Create a Class Mobile with the attributes listed below,

private String manufacturer;
private String operating_system;
public String color;
private int cost;

Define a Parameterized constructor to initialize the above instance variables.

Define getter and setter methods for the attributes above.

for example : setter method for manufacturer is

void setManufacturer(String manufacturer){

this.manufacturer= manufacturer;

}

String getManufacturer(){

 return manufacturer;}

Display the object details by overriding the toString() method.

For example:

Test	Result

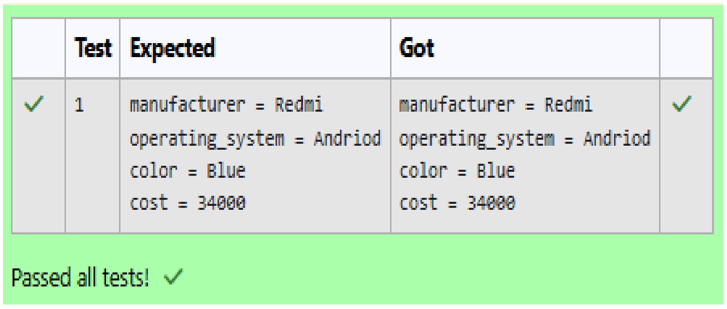
1	manufacturer = Redmi
	operating_system = Andriod
	color = Blue
	cost = 34000

SOLUTION :

```
p
  u
  b
  l
  i
  c

  c
  l
  a
```

```
mobile mobile=new
mobile("Redmi","Andriod","
Blue",34000);
```

OUTPUT :

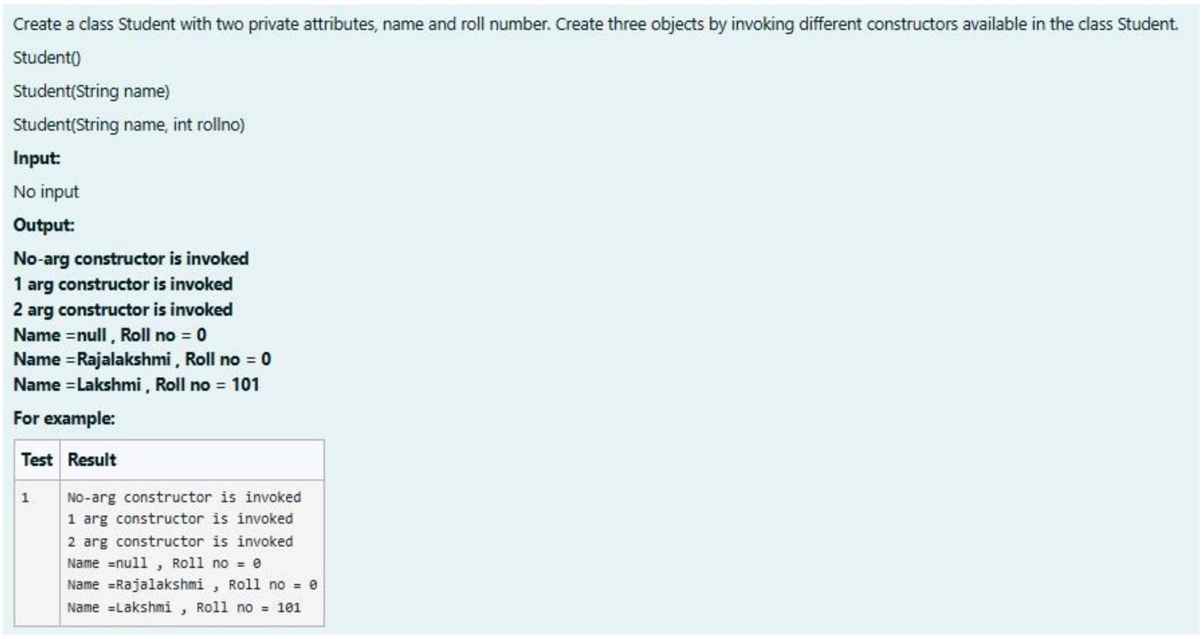


	Test	Expected	Got	

✓	1	manufacturer = Redmi
		operating_system = Andriod
		color = Blue
		cost = 34000
			manufacturer = Redmi
			operating_system = Andriod
			color = Blue
			cost = 34000
				✓

Passed all tests! ✓

3.



Create a class Student with two private attributes, name and roll number. Create three objects by invoking different constructors available in the class Student.

Student()

Student(String name)

Student(String name, int rollno)

Input:

No input

Output:

No-arg constructor is invoked
1 arg constructor is invoked
2 arg constructor is invoked
Name =null , Roll no = 0
Name =Rajalakshmi , Roll no = 0
Name =Lakshmi , Roll no = 101

For example:

Test	Result

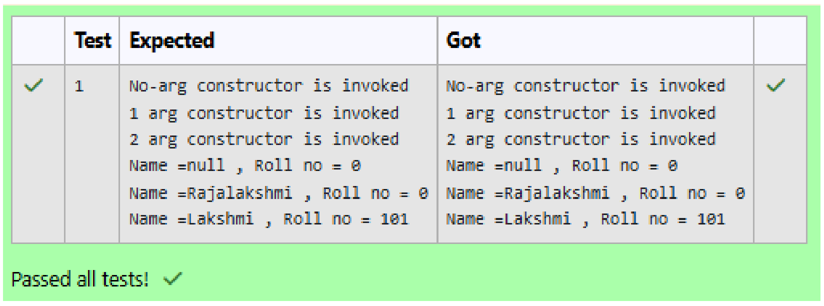
1	No-arg constructor is invoked
	1 arg constructor is invoked
	2 arg constructor is invoked
	Name =null , Roll no = 0
	Name =Rajalakshmi , Roll no = 0
	Name =Lakshmi , Roll no = 101

SOLUTION :

```
p
  u
  b
  l
  i
  c

  c
```

```
}
public stud(String
   name,int roll){
   System.out.print
   ln("2 arg
   constructor is
   invoked");
   this.name=name
   ;
   this.roll=roll;
```

OUTPUT :



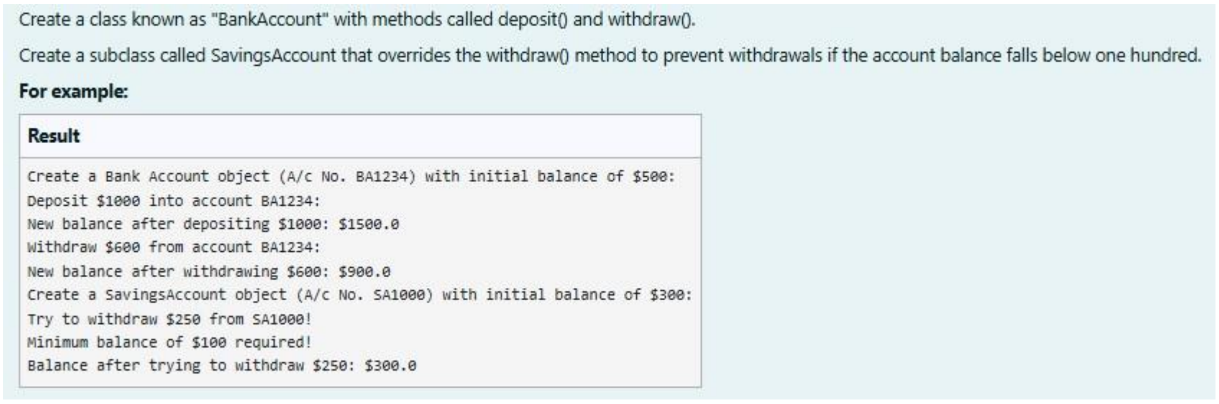
	Test	Expected	Got	

✓	1	No-arg constructor is invoked
		1 arg constructor is invoked
		2 arg constructor is invoked
		Name =null , Roll no = 0
		Name =Rajalakshmi , Roll no = 0
		Name =Lakshmi , Roll no = 101
			No-arg constructor is invoked
			1 arg constructor is invoked
			2 arg constructor is invoked
			Name =null , Roll no = 0
			Name =Rajalakshmi , Roll no = 0
			Name =Lakshmi , Roll no = 101
				✓

Passed all tests! ✓

Lab-05-Inheritance

1.



Create a class known as "BankAccount" with methods called deposit() and withdraw().

Create a subclass called SavingsAccount that overrides the withdraw() method to prevent withdrawals if the account balance falls below one hundred.

For example:

Result

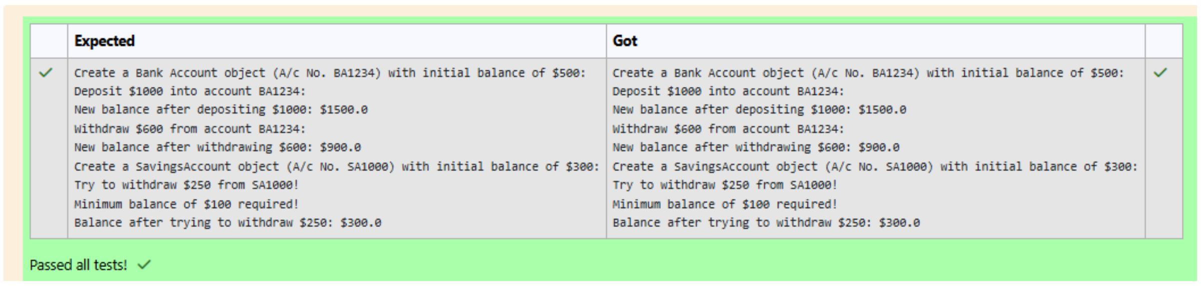
```
Create a Bank Account object (A/c No. BA1234) with initial balance of $500:
Deposit $1000 into account BA1234:
New balance after depositing $1000: $1500.0
Withdraw $600 from account BA1234:
New balance after withdrawing $600: $900.0
Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:
Try to withdraw $250 from SA1000!
Minimum balance of $100 required!
Balance after trying to withdraw $250: $300.0
```

SOLUTION :

```
class BankAccount {
   // Private
   field to
```

```
    private double balance;

    // Constructor to initialize account number and balance
    public BankAccount(String accountNumber,double balance){
        this.accountNumber=accountNumber;
        this.balance=balance;
    }




    // Method to deposit an amount into the account
    public void deposit(double amount) {
        // Increase the balance by the deposit amount
     balance+=amount;
    }

    // Method to withdraw an amount from the account
    public void withdraw(double amount) {
        // Check if the balance is sufficient for the withdrawal
        if (balance >= amount) {
            // Decrease the balance by the withdrawal amount
            balance -= amount;
        } else {
            // Print a message if the balance is insufficient
            System.out.println("Insufficient balance");
        }
    }

    // Method to get the current balance
    public double getBalance() {
        // Return the current balance
        return balance;
    }
    public String getAccountNumber(){
        return accountNumber;
    }
}
class SavingsAccount extends BankAccount {
    // Constructor to initialize account number and balance
    public SavingsAccount(String accountNumber, double balance) {
        // Call the parent class constructor
        super(accountNumber,balance);
    }

    // Override the withdraw method from the parent class
    @Override
    public void withdraw(double amount) {
        // Check if the withdrawal would cause the balance to drop below $100
        if (getBalance() - amount < 100) {
            // Print a message if the minimum balance requirement is not met
            System.out.println("Minimum balance of $100 required!");
        } else {
            // Call the parent class withdraw method
            super.withdraw(amount);
        }
    }
}

public class Main {

    public static void main(String[] args) {
        // Print message to indicate creation of a BankAccount object
        System.out.println("Create a Bank Account object (A/c No. BA1234) with initial
balance of $500:");
        // Create a BankAccount object (A/c No. "BA1234") with initial balance of $500
        BankAccount BA1234 = new BankAccount("BA1234", 500);
        // Print message to indicate deposit action
        System.out.println("Deposit $1000 into account BA1234:");
        // Deposit $1000 into account BA1234
       BA1234.deposit(1000);
        // Print the new balance after deposit
       System.out.println("New balance after depositing $1000: $"+BA1234.getBalance());

        // Print message to indicate withdrawal action
        System.out.println("Withdraw $600 from account BA1234:");
        // Withdraw $600 from account BA1234
      BA1234.withdraw(600);
        // Print the new balance after withdrawal
        System.out.println("New balance after withdrawing $600: $" +
BA1234.getBalance());

        // Print message to indicate creation of another SavingsAccount object
        System.out.println("Create a SavingsAccount object (A/c No. SA1000) with initial
balance of $300:");
        // Create a SavingsAccount object (A/c No. "SA1000") with initial balance of $300
        SavingsAccount SA1000 = new SavingsAccount("SA1000", 300);

        // Print message to indicate withdrawal action
        System.out.println("Try to withdraw $250 from SA1000!");
        // Withdraw $250 from SA1000 (balance falls below $100)
        SA1000.withdraw(250);
        // Print the balance after attempting to withdraw $250
        System.out.println("Balance after trying to withdraw $250: $" +
SA1000.getBalance());
    }
}
```

OUTPUT :



✓ **Expected**

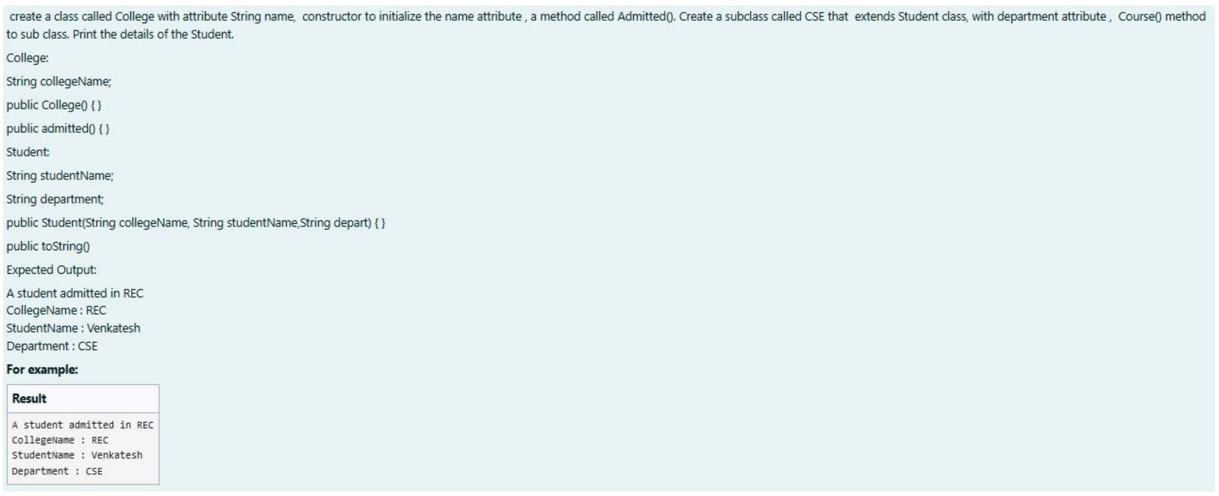
```
Create a Bank Account object (A/c No. BA1234) with initial balance of $500:
Deposit $1000 into account BA1234:
New balance after depositing $1000: $1500.0
Withdraw $600 from account BA1234:
New balance after withdrawing $600: $900.0
Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:
Try to withdraw $250 from SA1000!
Minimum balance of $100 required!
Balance after trying to withdraw $250: $300.0
```

Got ✓

```
Create a Bank Account object (A/c No. BA1234) with initial balance of $500:
Deposit $1000 into account BA1234:
New balance after depositing $1000: $1500.0
Withdraw $600 from account BA1234:
New balance after withdrawing $600: $900.0
Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:
Try to withdraw $250 from SA1000!
Minimum balance of $100 required!
Balance after trying to withdraw $250: $300.0
```

Passed all tests! ✓

2.



create a class called College with attribute String name, constructor to initialize the name attribute , a method called Admitted(). Create a subclass called CSE that extends Student class, with department attribute , Course() method to sub class. Print the details of the Student.

College:

String collegeName;

public College() { }

public admitted() { }

Student:

String studentName;

String department;

public Student(String collegeName, String studentName,String depart) { }

public toString()

Expected Output:

A student admitted in REC
CollegeName : REC
StudentName : Venkatesh
Department : CSE

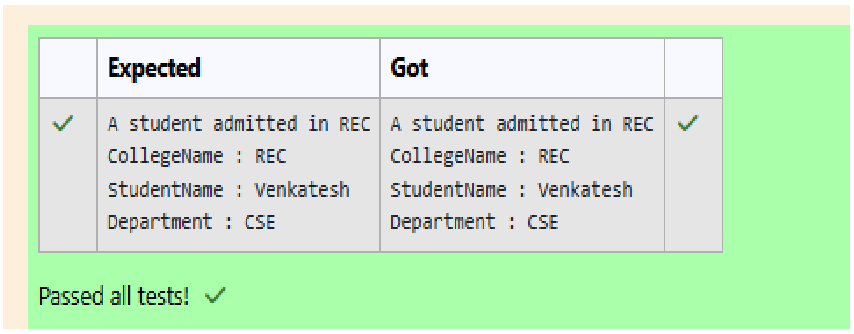
For example:

Result
A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE

SOLUTION :

```
class College
{
public  String collegeName;

publ
 ic
Coll
ege(
 Stri
  ng
 coll
 ege
  Na
 me)
   {
}

public String toString(){
    // return the details of the student
    return "CollegeName :
"+collegeName+"\n"+"Studen
tName :
"+studentName+"\n"+"Depart
ment : "+department;
}
```

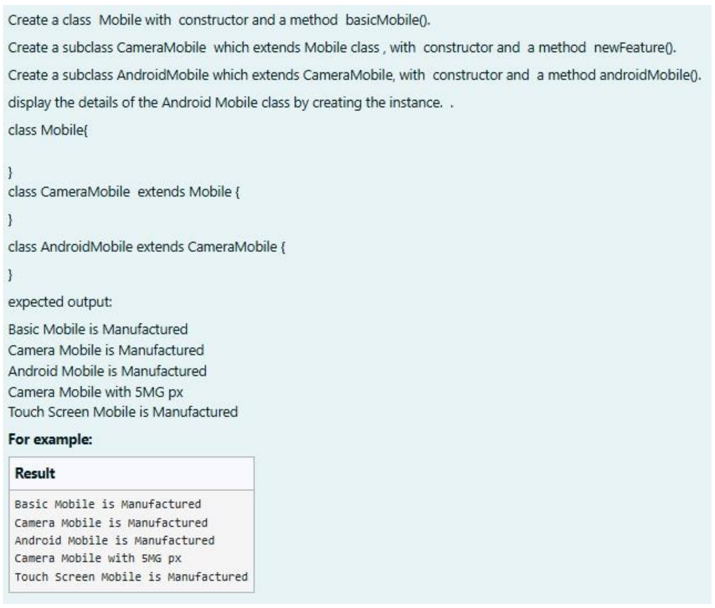
OUTPUT :



	Expected	Got	
✓	A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE	A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE	✓

Passed all tests! ✓

3.



Create a class Mobile with constructor and a method basicMobile().

Create a subclass CameraMobile which extends Mobile class , with constructor and a method newFeature().

Create a subclass AndroidMobile which extends CameraMobile, with constructor and a method androidMobile().

display the details of the Android Mobile class by creating the instance. .

```
class Mobile{

}
class CameraMobile extends Mobile {
}
class AndroidMobile extends CameraMobile {
}
```

expected output:

```
Basic Mobile is Manufactured
Camera Mobile is Manufactured
Android Mobile is Manufactured
Camera Mobile with 5MG px
Touch Screen Mobile is Manufactured
```

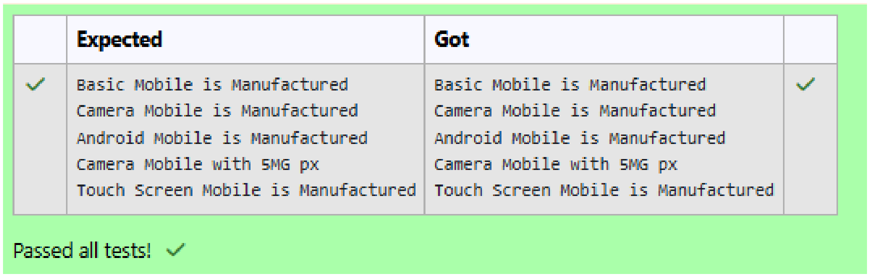
For example:

Result
Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured

SOLUTION :

```
c
    l
    }
    void basmob(){
        System.out.println("Basic Mobile is Manufactured");
    }
}
c
    l
    a
    s
    s

    c
    a
    m

    e
    x
    t
    e
    n
```

OUTPUT :

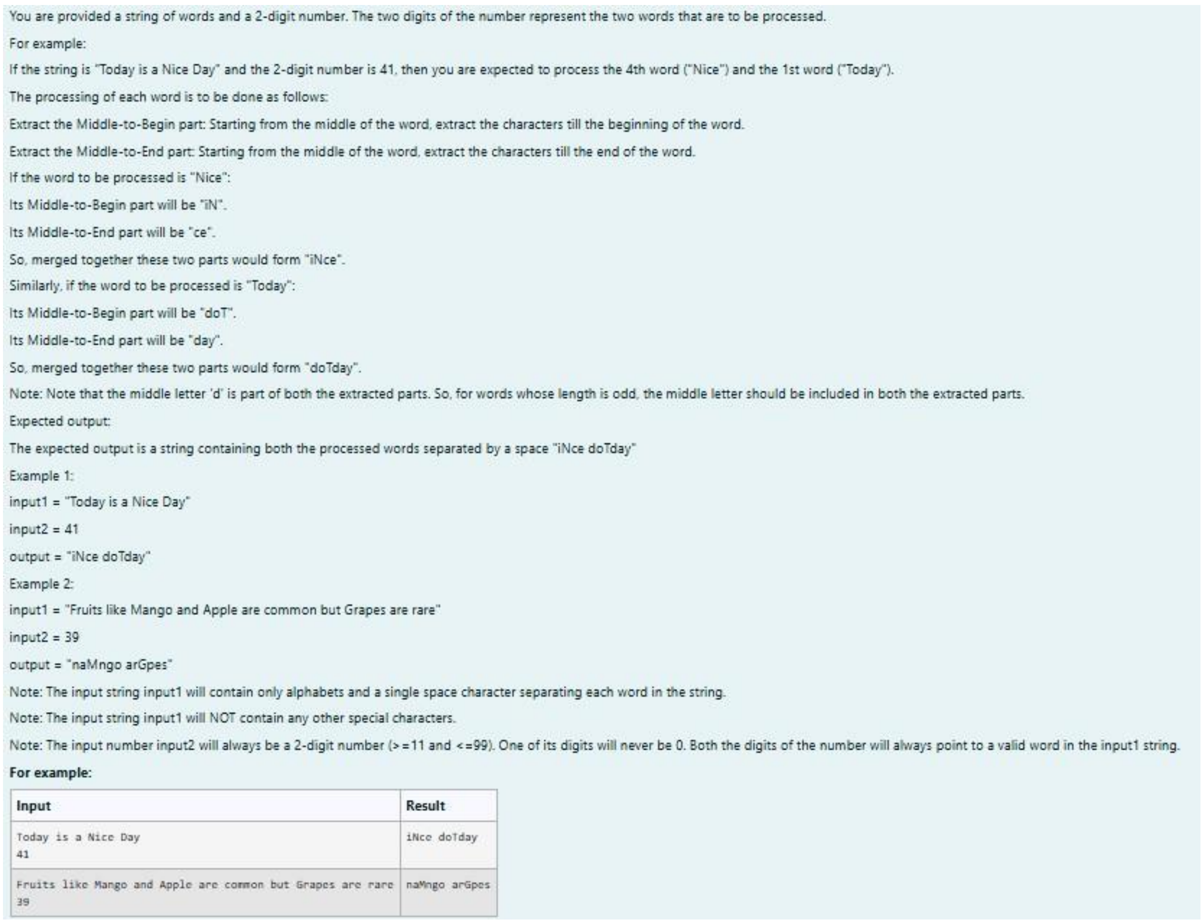


	Expected	Got	
✓	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	✓

Passed all tests! ✓

Lab-06-String, StringBuffer

1.


You are provided a string of words and a 2-digit number. The two digits of the number represent the two words that are to be processed.

For example:

If the string is "Today is a Nice Day" and the 2-digit number is 41, then you are expected to process the 4th word ("Nice") and the 1st word ("Today").

The processing of each word is to be done as follows:

Extract the Middle-to-Begin part: Starting from the middle of the word, extract the characters till the beginning of the word.

Extract the Middle-to-End part: Starting from the middle of the word, extract the characters till the end of the word.

If the word to be processed is "Nice":

Its Middle-to-Begin part will be "iN".

Its Middle-to-End part will be "ce".

So, merged together these two parts would form "iNce".

Similarly, if the word to be processed is "Today":

Its Middle-to-Begin part will be "doT".

Its Middle-to-End part will be "day".

So, merged together these two parts would form "doTday".

Note: Note that the middle letter 'd' is part of both the extracted parts. So, for words whose length is odd, the middle letter should be included in both the extracted parts.

Expected output:

The expected output is a string containing both the processed words separated by a space "iNce doTday"

Example 1:

input1 = "Today is a Nice Day"

input2 = 41

output = "iNce doTday"

Example 2:

input1 = "Fruits like Mango and Apple are common but Grapes are rare"

input2 = 39

output = "naMngo arGpes"

Note: The input string input1 will contain only alphabets and a single space character separating each word in the string.

Note: The input string input1 will NOT contain any other special characters.

Note: The input number input2 will always be a 2-digit number (>=11 and <=99). One of its digits will never be 0. Both the digits of the number will always point to a valid word in the input1 string.

For example:

Input	Result

`Today is a Nice Day`
`41`
	`iNce doTday`

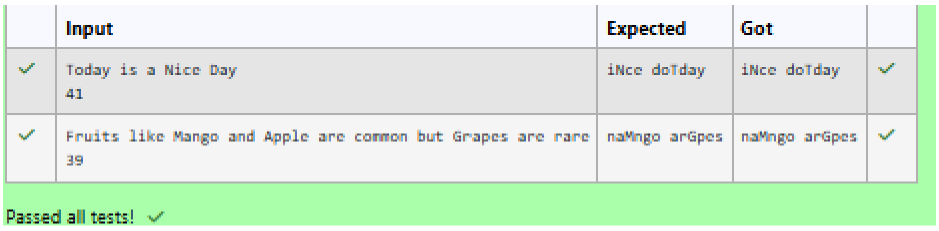
`Fruits like Mango and Apple are common but Grapes are rare`
`39`
	`naMngo arGpes`

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
```

```
}
s
p
a
c
e

=

0

;

f
l
a
g

=
```

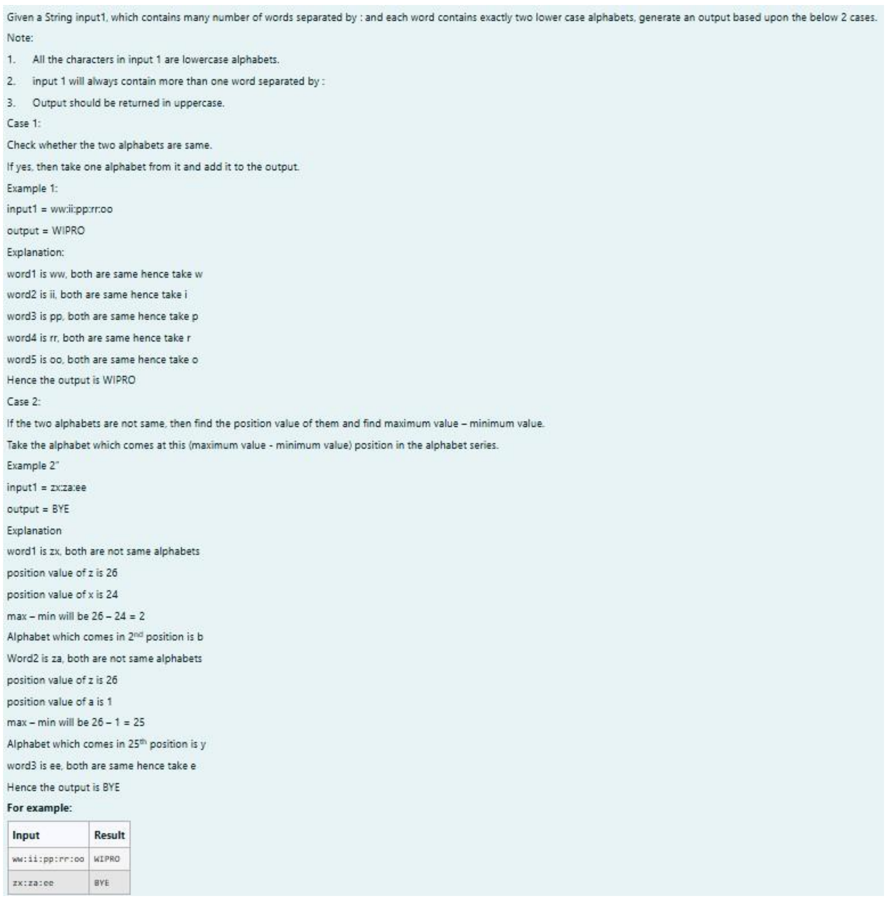
OUTPUT :



	Input	Expected	Got	
✓	Today is a Nice Day 41	iNce doTday	iNce doTday	✓
✓	Fruits like Mango and Apple are common but Grapes are rare 39	naMngo arGpes	naMngo arGpes	✓

Passed all tests! ✓

2.



Given a String input1, which contains many number of words separated by : and each word contains exactly two lower case alphabets, generate an output based upon the below 2 cases.

Note:

1. All the characters in input 1 are lowercase alphabets.
2. input 1 will always contain more than one word separated by :
3. Output should be returned in uppercase.

Case 1:

Check whether the two alphabets are same.

If yes, then take one alphabet from it and add it to the output.

Example 1:

input1 = ww:ii:pp:rr:oo

output = WIPRO

Explanation:

word1 is ww, both are same hence take w

word2 is ii, both are same hence take i

word3 is pp, both are same hence take p

word4 is rr, both are same hence take r

word5 is oo, both are same hence take o

Hence the output is WIPRO

Case 2:

If the two alphabets are not same, then find the position value of them and find maximum value – minimum value.

Take the alphabet which comes at this (maximum value - minimum value) position in the alphabet series.

Example 2"

input1 = zx:za:ee

output = BYE

Explanation

word1 is zx, both are not same alphabets

position value of z is 26

position value of x is 24

max – min will be 26 – 24 = 2

Alphabet which comes in 2nd position is b

Word2 is za, both are not same alphabets

position value of z is 26

position value of a is 1

max – min will be 26 – 1 = 25

Alphabet which comes in 25th position is y

word3 is ee, both are same hence take e

Hence the output is BYE

For example:

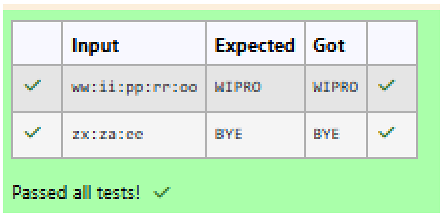
Input	Result
ww:ii:pp:rr:oo	WIPRO
zx:za:ee	BYE

SOLUTION :

```
i
m
p
o
r
t
 
j
a
v
a
.
u
t
```

```
}
String h = temp.toString();
f
  o
  r
  (
```

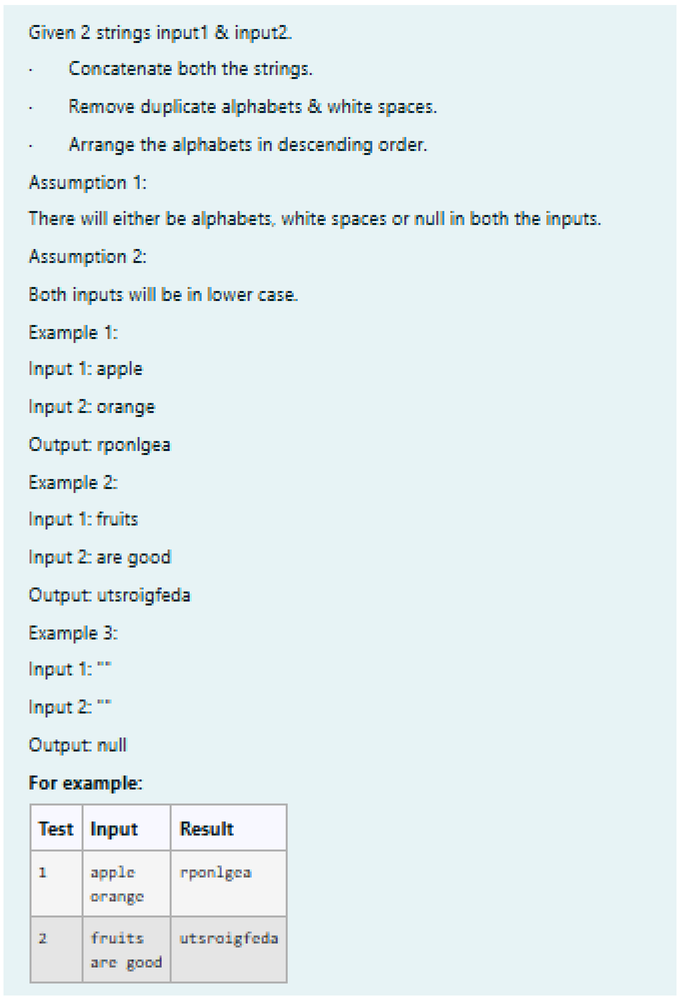
OUTPUT :



	Input	Expected	Got	
✓	ww:ii:pp:rr:oo	WIPRO	WIPRO	✓
✓	zx:za:ee	BYE	BYE	✓

Passed all tests! ✓

3.



Given 2 strings input1 & input2.

- Concatenate both the strings.
- Remove duplicate alphabets & white spaces.
- Arrange the alphabets in descending order.

Assumption 1:

There will either be alphabets, white spaces or null in both the inputs.

Assumption 2:

Both inputs will be in lower case.

Example 1:

Input 1: apple

Input 2: orange

Output: rponlgea

Example 2:

Input 1: fruits

Input 2: are good

Output: utsroigfeda

Example 3:

Input 1: ""

Input 2: ""

Output: null

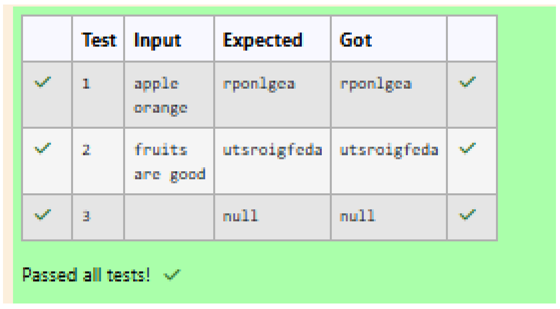
For example:

Test	Input	Result
1	apple orange	rponlgea
2	fruits are good	utsroigfeda

SOLUTION :

```
import java.util.*;


public class HelloWorld {
    public
        static
        void
        main(S
        tring[]
        args) {
        Scanne
        r scan
        = new
        Scanne
        r(Syste
        m.in);
        String
        a =
        scan.n
        extLin
        e();
        Stri
```

OUTPUT :

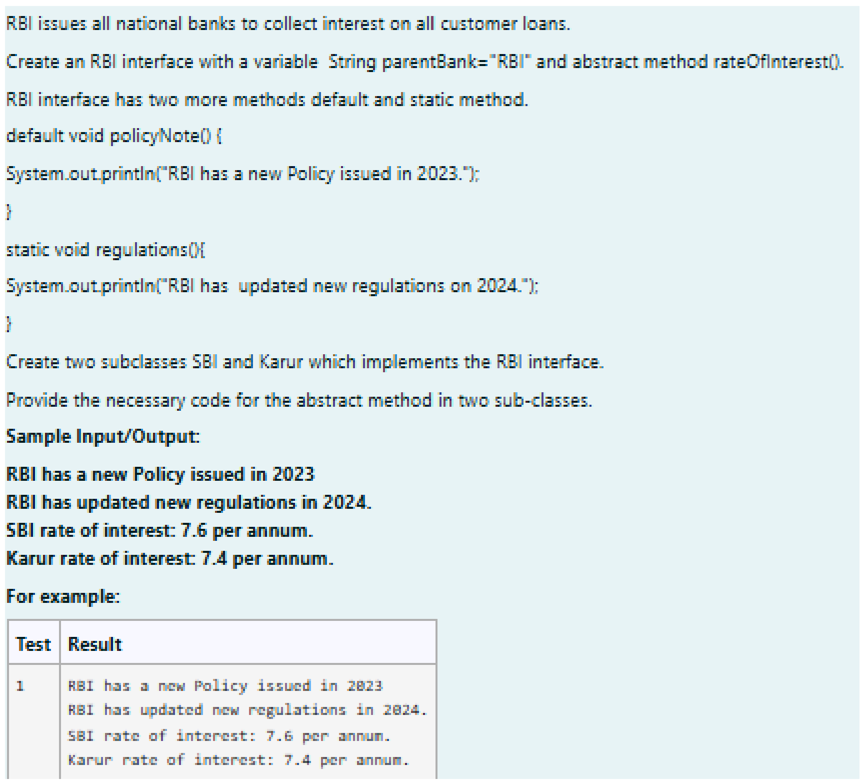


	Test	Input	Expected	Got	
✓	1	apple orange	rponlgea	rponlgea	✓
✓	2	fruits are good	utsroigfeda	utsroigfeda	✓
✓	3		null	null	✓

Passed all tests! ✓

Lab-07-Interfaces

1.



RBI issues all national banks to collect interest on all customer loans.

Create an RBI interface with a variable String parentBank="RBI" and abstract method rateOfInterest().

RBI interface has two more methods default and static method.

```
default void policyNote() {
System.out.println("RBI has a new Policy issued in 2023.");
}
static void regulations(){
System.out.println("RBI has  updated new regulations on 2024.");
}
```

Create two subclasses SBI and Karur which implements the RBI interface.

Provide the necessary code for the abstract method in two sub-classes.

Sample Input/Output:

RBI has a new Policy issued in 2023
RBI has updated new regulations in 2024.
SBI rate of interest: 7.6 per annum.
Karur rate of interest: 7.4 per annum.

For example:

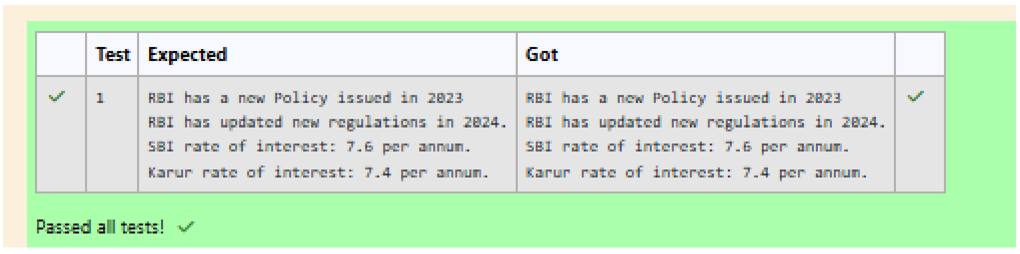
Test	Result
1	RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum.

SOLUTION :

```
/
/

D
e
f
i
n
e

t
h
e

R
```

```
        return 7.6;
    }
}

// Karur
class
implem
enting
RBI
interfac
eclass
Karur
implem
ents
RBI {
    //
    I
    m
    pl
```

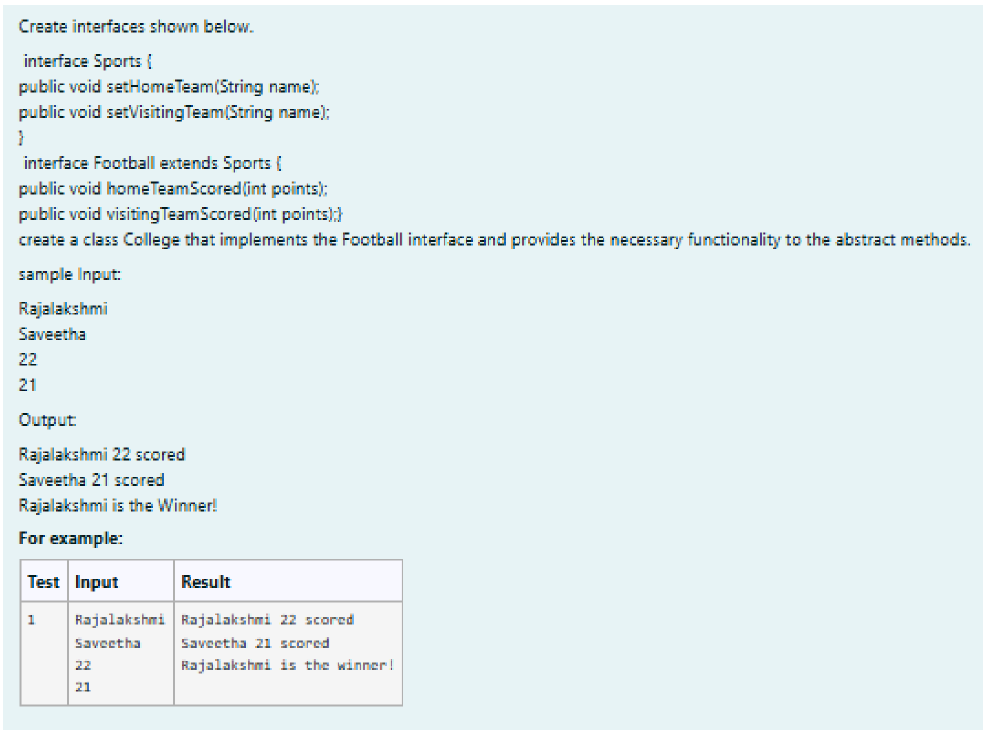
OUTPUT :



	Test	Expected	Got	
✓	1	RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum.	RBI has a new Policy issued in 2023 RBI has updated new regulations in 2024. SBI rate of interest: 7.6 per annum. Karur rate of interest: 7.4 per annum.	✓

Passed all tests! ✓

2.


Create interfaces shown below.

```
 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);}
```

create a class College that implements the Football interface and provides the necessary functionality to the abstract methods.

sample Input:

```
Rajalakshmi
Saveetha
22
21
```

Output:

```
Rajalakshmi 22 scored
Saveetha 21 scored
Rajalakshmi is the Winner!
```

For example:

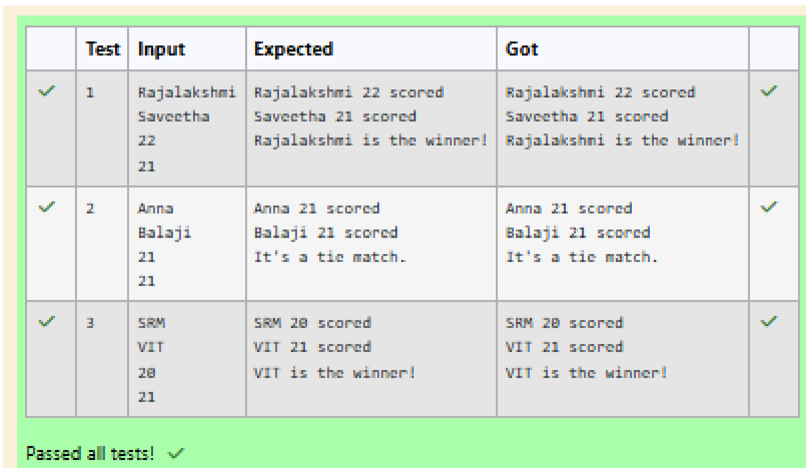
Test	Input	Result
1	Rajalakshmi Saveetha 22 21	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner!

SOLUTION :

```
import java.util.Scanner;

interface Sports {
    voi
    d
    set
    Ho
    me
    Te
    am
    (St
    rin
    g
    na
    me
    );
    voi
```

```
        homeTeamPoints += points;
        System.out.println(homeTeam + " " + points + " scored");
    }

    public void visitingTeamScored(int points) {
        visitingTeamPoints += points;
        System.out.println(visitingTeam + " " + points + " scored");
    }

    public void winningTeam() {
        if (homeTeamPoints > visitingTeamPoints) {
            System.out.println(homeTeam + " is the winner!");
        } else if (homeTeamPoints < visitingTeamPoints) {
            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) {
        Scanner sc = new Scanner(System.in);

        // Get home team name
        String hname = sc.nextLine();

        // Get visiting team name
        String vteam = sc.nextLine();

        // Create College object
        College match = new College();
        match.setHomeTeam(hname);
        match.setVisitingTeam(vteam);

        // Get points scored by home team
        int htpoints = sc.nextInt();
        match.homeTeamScored(htpoints);

        // Get points scored by visiting team
        int vtpoints = sc.nextInt();
        match.visitingTeamScored(vtpoints);

        // Determine and print the winning team
        match.winningTeam();

        sc.close();
    }
}
```

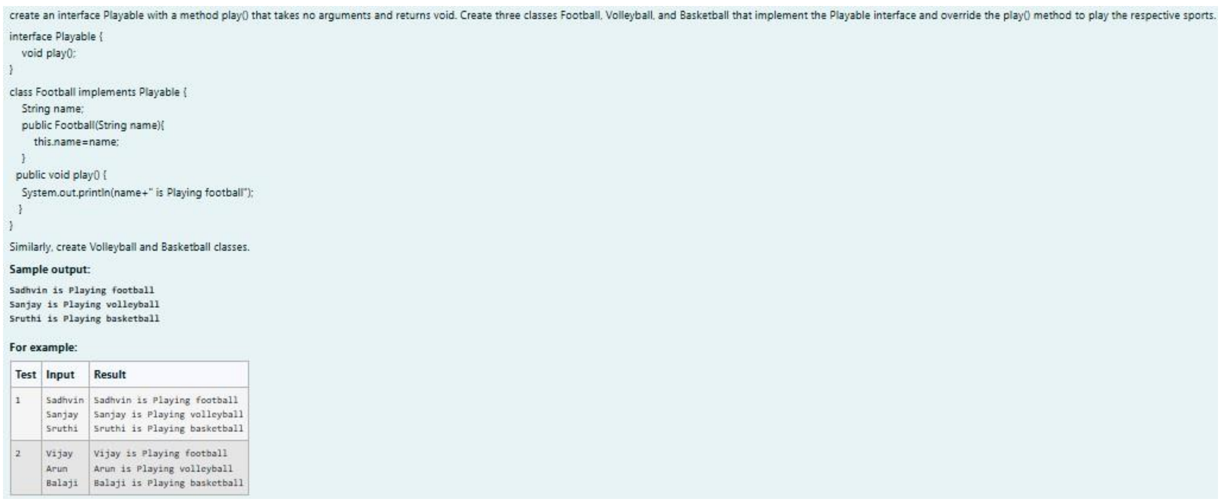
OUTPUT :



	Test	Input	Expected	Got	
✓	1	Rajalakshmi Saveetha 22 21	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner!	Rajalakshmi 22 scored Saveetha 21 scored Rajalakshmi is the winner!	✓
✓	2	Anna Balaji 21 21	Anna 21 scored Balaji 21 scored It's a tie match.	Anna 21 scored Balaji 21 scored It's a tie match.	✓
✓	3	SRM VIT 20 21	SRM 20 scored VIT 21 scored VIT is the winner!	SRM 20 scored VIT 21 scored VIT is the winner!	✓

Passed all tests! ✓

3.



create an interface Playable with a method play() that takes no arguments and returns void. Create three classes Football, Volleyball, and Basketball that implement the Playable interface and override the play() method to play the respective sports.

```
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");
  }
}
```

Similarly, create Volleyball and Basketball classes.

Sample output:

```
Sadhvin is Playing football
Sanjay is Playing volleyball
Sruthi is Playing basketball
```

For example:

Test	Input	Result
1	Sadhvin Sanjay Sruthi	Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball
2	Vijay Arun Balaji	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball

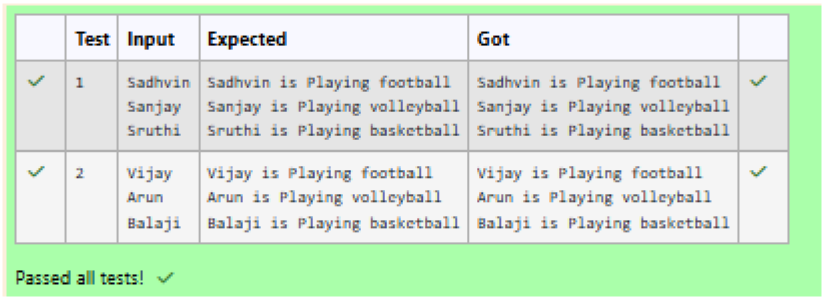
SOLUTION :

```
import java.util.Scanner;

/
/

D
e
f
i
n
e
```

```
    public void play() {
        System.out.println(name + " is Playing football");
    }
}

// Volleyball class implementing Playable interface
class Volleyball implements Playable {
    String name;

    // Constructor
    public Volleyball(String name) {
        this.name = name;
    }

    // Override the play method
    public void play() {
        System.out.println(name + " is Playing volleyball");
    }
}

// Basketball class implementing Playable interface
class Basketball implements Playable {
    String name;

    // Constructor
    public Basketball(String name) {
        this.name = name;
    }

    // Override the play method
    public void play() {
        System.out.println(name + " is Playing basketball");
    }
}

// Main class to test the functionality
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input for Football player

        String footballPlayerName = scanner.nextLine();
        Football footballPlayer = new Football(footballPlayerName);

        // Input for Volleyball player

        String volleyballPlayerName = scanner.nextLine();
        Volleyball volleyballPlayer = new Volleyball(volleyballPlayerName);
```

```
// Input for Basketball player

String basketballPlayerName = scanner.nextLine();
Basketball basketballPlayer = new
Basketball(basketballPlayerName);

//
Call
```

OUTPUT :

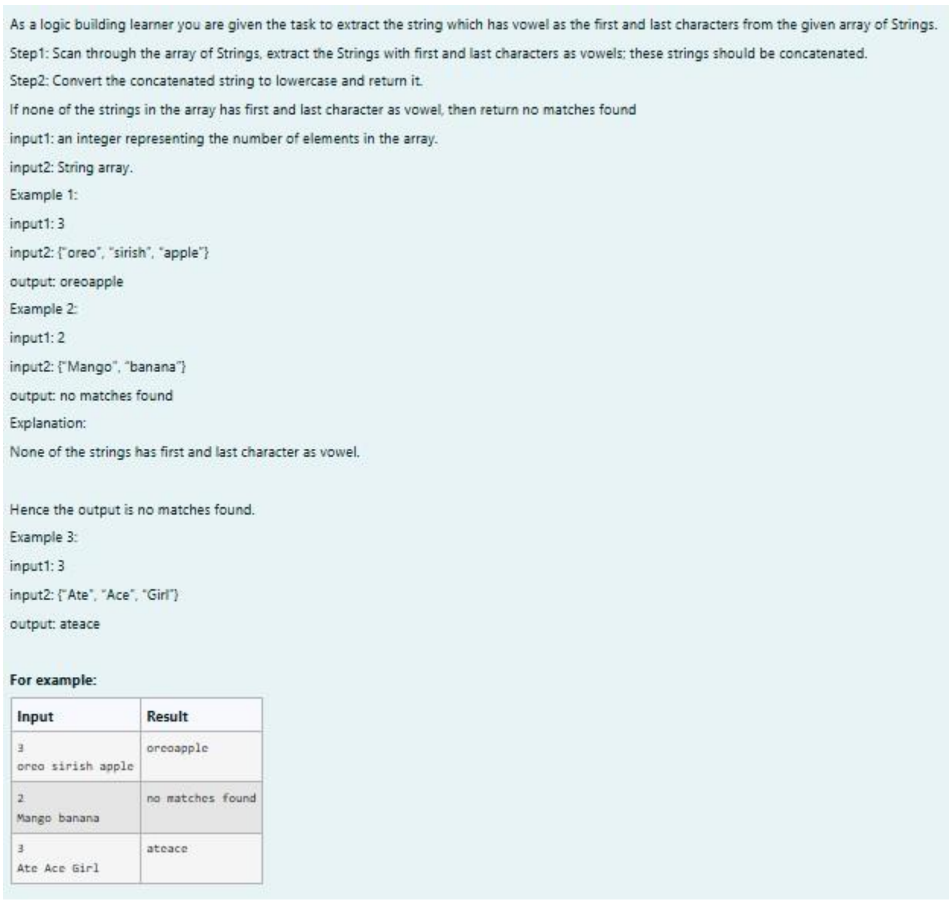


	Test	Input	Expected	Got	
✓	1	Sadhvin Sanjay Sruthi	Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball	Sadhvin is Playing football Sanjay is Playing volleyball Sruthi is Playing basketball	✓
✓	2	Vijay Arun Balaji	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball	Vijay is Playing football Arun is Playing volleyball Balaji is Playing basketball	✓

Passed all tests! ✓

Lab-08 - Polymorphism, Abstract Classes, final Keyword

1.



As a logic building learner you are given the task to extract the string which has vowel as the first and last characters from the given array of Strings.

Step1: Scan through the array of Strings, extract the Strings with first and last characters as vowels; these strings should be concatenated.

Step2: Convert the concatenated string to lowercase and return it.

If none of the strings in the array has first and last character as vowel, then return no matches found

input1: an integer representing the number of elements in the array.

input2: String array.

Example 1:

input1: 3

input2: {"oreo", "sirish", "apple"}

output: oreoapple

Example 2:

input1: 2

input2: {"Mango", "banana"}

output: no matches found

Explanation:

None of the strings has first and last character as vowel.

Hence the output is no matches found.

Example 3:

input1: 3

input2: {"Ate", "Ace", "Girl"}

output: ateace

For example:

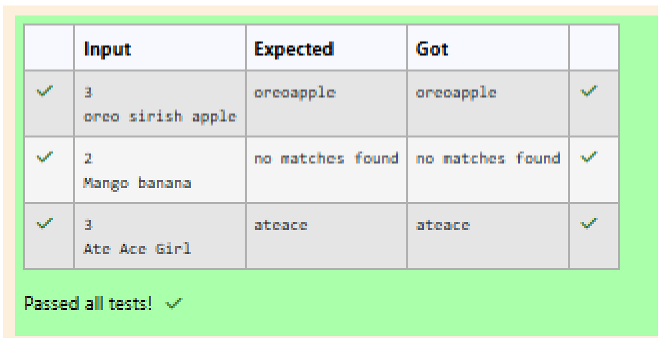
Input	Result
3 oreo sirish apple	oreoapple
2 Mango banana	no matches found
3 Ate Ace Girl	ateace

SOLUTION :

```
import java.util.Scanner;

public class VowelStringExtractor {

    // Method to extract strings with
    vowels as first and last
    characterspublic static String
    extractVowelStrings(String[]
    stringArray) {
        StringBuilder result = new StringBuilder();
        String vowels = "aeiouAEIOU"; // String containing
        all vowels

        //
```

```
public static
   void
   main(Stri
   ng[] args)
   { Scanner
   scanner =
   new
   Scanner(S
   ystem.in);

   /

   /
```

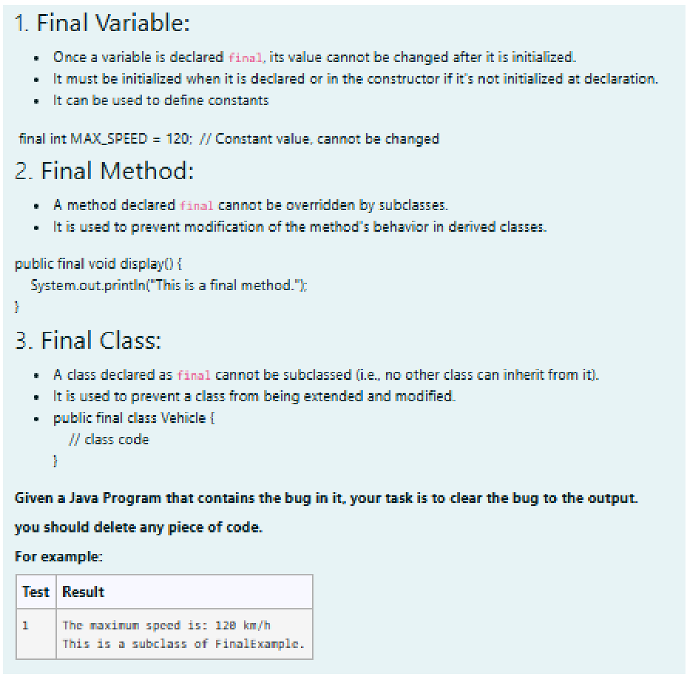
OUTPUT :



	Input	Expected	Got	
✓	3 oreo sirish apple	oreoapple	oreoapple	✓
✓	2 Mango banana	no matches found	no matches found	✓
✓	3 Ate Ace Girl	ateace	ateace	✓

Passed all tests! ✓

2.



1. Final Variable:

- Once a variable is declared `final`, its value cannot be changed after it is initialized.
- It must be initialized when it is declared or in the constructor if it's not initialized at declaration.
- It can be used to define constants

```
final int MAX_SPEED = 120;  // Constant value, cannot be changed
```

2. Final Method:

- A method declared `final` cannot be overridden by subclasses.
- It is used to prevent modification of the method's behavior in derived classes.

```
public final void display() {
    System.out.println("This is a final method.");
}
```

3. Final Class:

- A class declared as `final` cannot be subclassed (i.e., no other class can inherit from it).
- It is used to prevent a class from being extended and modified.
- public final class Vehicle {
      // class code
  }

Given a Java Program that contains the bug in it, your task is to clear the bug to the output.

you should delete any piece of code.

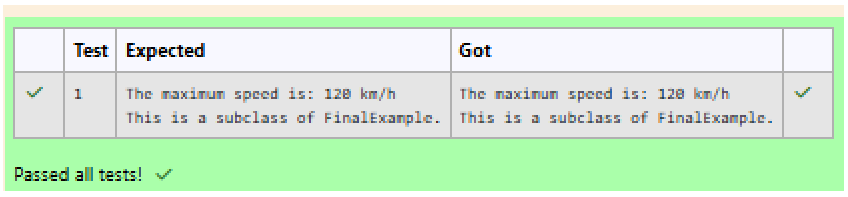
For example:

Test	Result
1	The maximum speed is: 120 km/h This is a subclass of FinalExample.

SOLUTION :

```
/
/

F
i
n
a
l

c
l
a
s
s

d
```

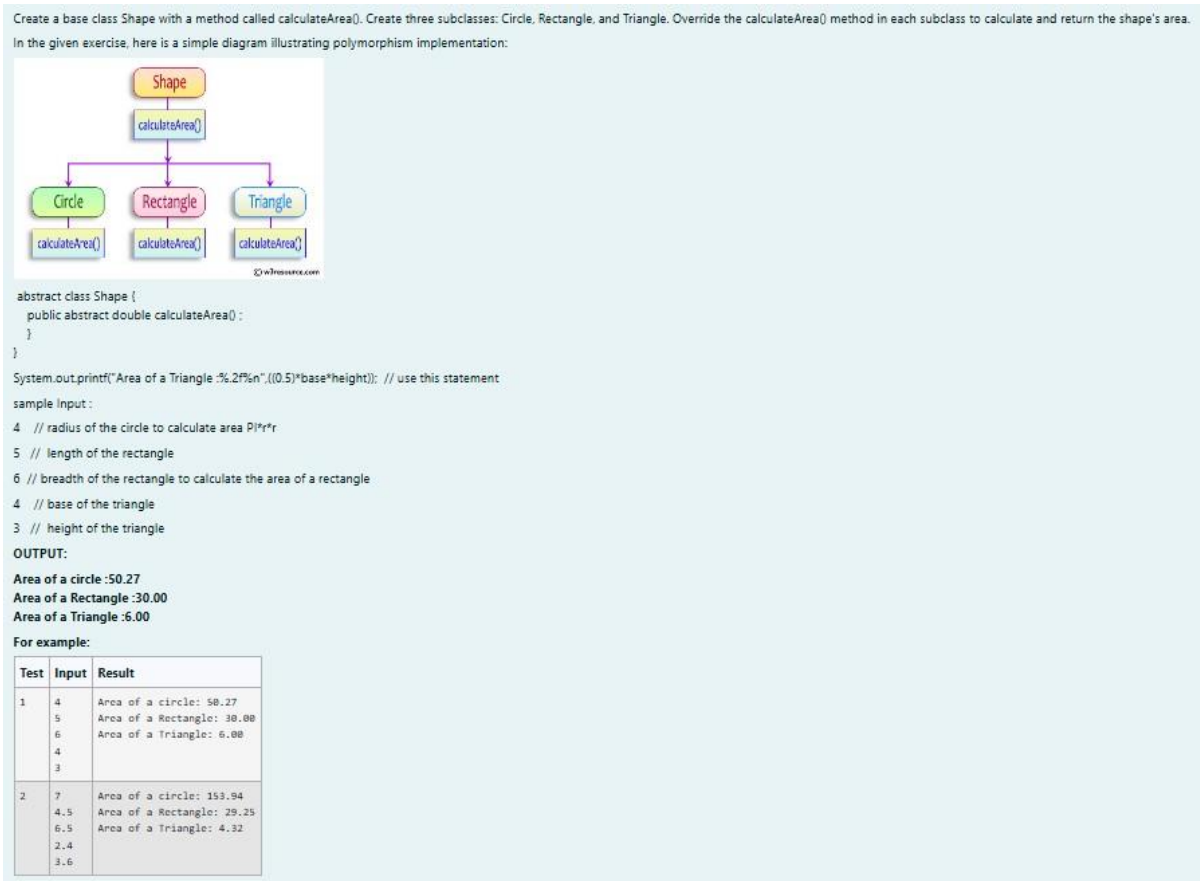
OUTPUT :



	Test	Expected	Got	
✓	1	The maximum speed is: 120 km/h This is a subclass of FinalExample.	The maximum speed is: 120 km/h This is a subclass of FinalExample.	✓

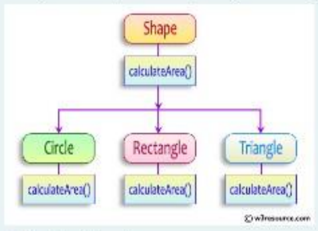
Passed all tests! ✓

3.



Create a base class Shape with a method called calculateArea(). Create three subclasses: Circle, Rectangle, and Triangle. Override the calculateArea() method in each subclass to calculate and return the shape's area.

In the given exercise, here is a simple diagram illustrating polymorphism implementation:



```
 abstract class Shape {
   public abstract double calculateArea() ;
   }
}
```

System.out.printf("Area of a Triangle :%.2f%n",((0.5)*base*height)); // use this statement

sample Input :

4 // radius of the circle to calculate area PI*r*r

5 // length of the rectangle

6 // breadth of the rectangle to calculate the area of a rectangle

4 // base of the triangle

3 // height of the triangle

OUTPUT:

Area of a circle :50.27
Area of a Rectangle :30.00
Area of a Triangle :6.00

For example:

Test	Input	Result
1	4 5 6 4 3	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00
2	7 4.5 6.5 2.4 3.6	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32

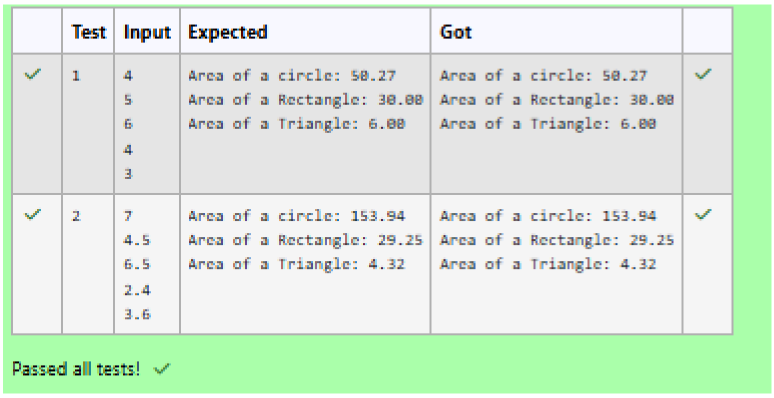
SOLUTION :

```
import java.util.Scanner;

/
/

A
b
s
t
r
```

```
    public double calculateArea() {
        return Math.PI * radius * radius; // Area of circle: πr²
    }
}

// Rectangle class
class Rectangle extends Shape {
    private double length;
    private double breadth;

    public Rectangle(double length, double breadth) {
        this.length = length;
        this.breadth = breadth;
    }

    @Override
    public double calculateArea() {
        return length * breadth; // Area of rectangle: length * breadth
    }
}

// Triangle class
class Triangle extends Shape {
    private double base;
    private double height;

    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }

    @Override
    public double calculateArea() {
        return 0.5 * base * height; // Area of triangle: 0.5 * base * height
    }
}

// Main class to test the shapes
public class ShapeTest {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input for Circle

        double radius = scanner.nextDouble();
        Circle circle = new Circle(radius);
        System.out.printf("Area of a circle: %.2f%n", circle.calculateArea());

        // Input for Rectangle
```

```
doubl

e

lengt

h =

scann

er.ne
```

OUTPUT :

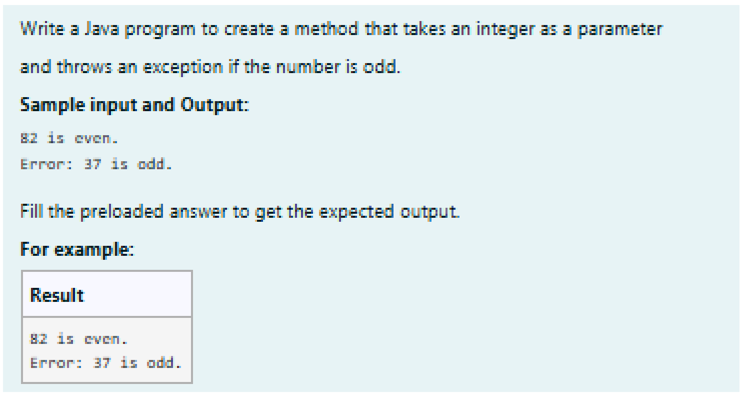


	Test	Input	Expected	Got	
✓	1	4 5 6 4 3	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00	Area of a circle: 50.27 Area of a Rectangle: 30.00 Area of a Triangle: 6.00	✓
✓	2	7 4.5 6.5 2.4 3.6	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32	Area of a circle: 153.94 Area of a Rectangle: 29.25 Area of a Triangle: 4.32	✓

Passed all tests! ✓

Lab-09-Exception Handling

1.



Write a Java program to create a method that takes an integer as a parameter and throws an exception if the number is odd.

Sample input and Output:

```
82 is even.
Error: 37 is odd.
```

Fill the preloaded answer to get the expected output.

For example:

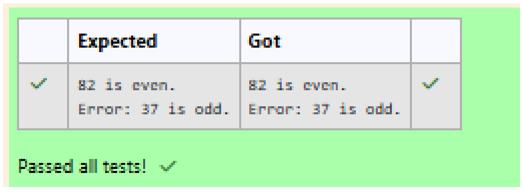
Result
82 is even. Error: 37 is odd.

SOLUTION :

```
class prog {
```

```
i
n
t

n

=

8
2
;

t
r
```

OUTPUT :

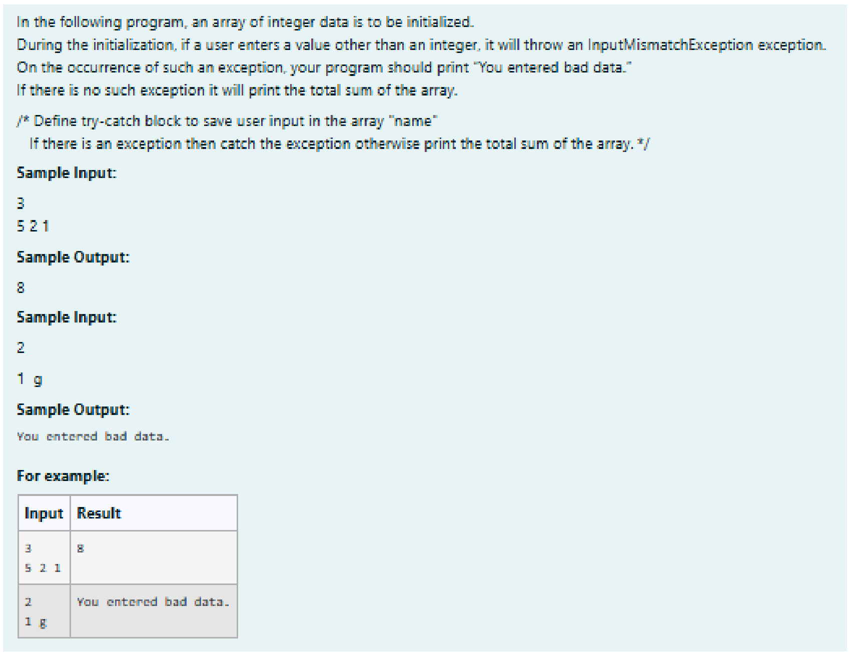


	Expected	Got	

✓	`82 is even.`
	`Error: 37 is odd.`
		`82 is even.`
		`Error: 37 is odd.`
			✓

Passed all tests! ✓

2.



In the following program, an array of integer data is to be initialized.
During the initialization, if a user enters a value other than an integer, it will throw an InputMismatchException exception.
On the occurrence of such an exception, your program should print "You entered bad data."
If there is no such exception it will print 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. */

Sample Input:

3
5 2 1

Sample Output:

8

Sample Input:

2

1 g

Sample Output:

```
You entered bad data.
```

For example:

Input	Result

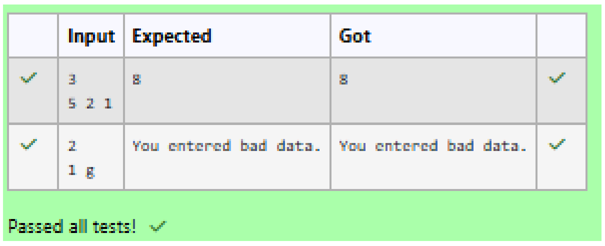
`3`
`5 2 1`
	`8`

`2`
`1 g`
	`You entered bad data.`

SOLUTION :

```
import java.util.Scanner;
import java.util.InputMismatchException;

class prog {
    public
        stati
        c
        void
        main
        (Stri
        ng[]
        args)
        {
        Scan
        ner
        sc =
        new
        Scan
        ner(
        Syst
        em.i
```

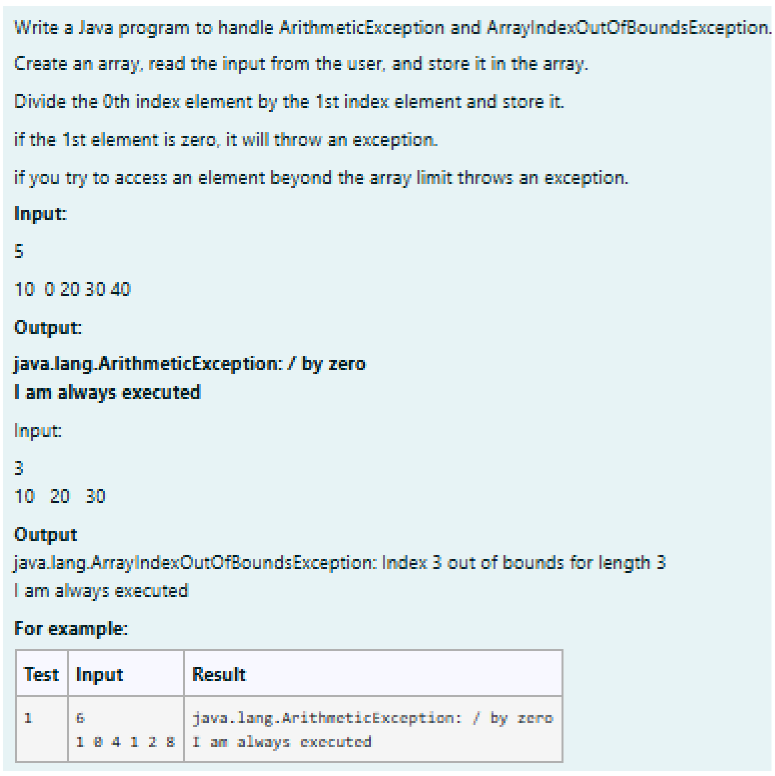
OUTPUT :



	Input	Expected	Got	
✓	3 5 2 1	8	8	✓
✓	2 1 g	You entered bad data.	You entered bad data.	✓

Passed all tests! ✓

3.



Write a Java program to handle ArithmeticException and ArrayIndexOutOfBoundsException.

Create an array, read the input from the user, and store it in the array.

Divide the 0th index element by the 1st index element and store it.

if the 1st element is zero, it will throw an exception.

if you try to access an element beyond the array limit throws an exception.

Input:

5

10 0 20 30 40

Output:

java.lang.ArithmeticException: / by zero
I am always executed

Input:

3
10 20 30

Output
java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3
I am always executed

For example:

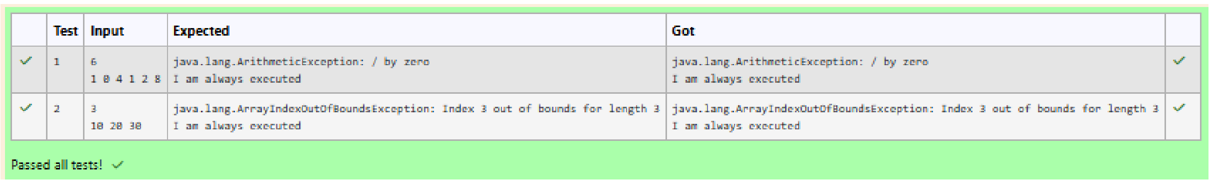
Test	Input	Result
1	6 1 0 4 1 2 8	java.lang.ArithmeticException: / by zero I am always executed

SOLUTION :

```
import java.util.Scanner;

public
    class
    Excep
    tionH
    andlin
    gExa
    mple
    {
    public
    static
    void
    main(
    String
    []
    args)
```

```
        } finally {
            // This block is always executed
        }

        try {
            // Attempt to access an out-of-bounds index
            int
outOfBoundsValue =
numbers[3]; // This will
trigger
ArrayIndexOutOfBoundsExc
option if size < 4
```

OUTPUT :

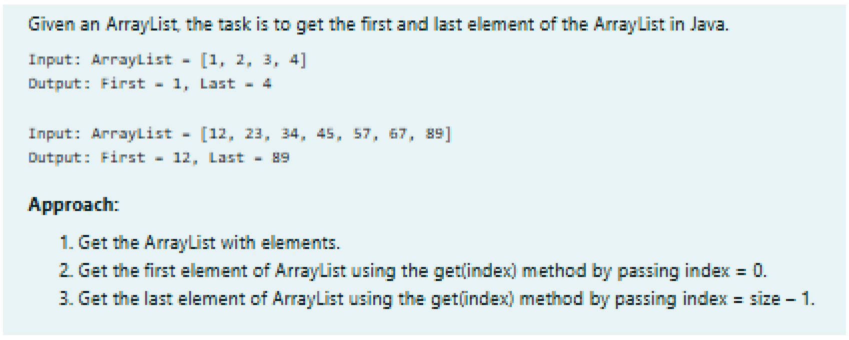


	Test	Input	Expected	Got	
✓	1	6 1 0 4 1 2 8	java.lang.ArithmeticException: / by zero I am always executed	java.lang.ArithmeticException: / by zero I am always executed	✓
✓	2	3 10 20 30	java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3 I am always executed	java.lang.ArrayIndexOutOfBoundsException: Index 3 out of bounds for length 3 I am always executed	✓

Passed all tests! ✓

Lab-10- Collection- List

1.



Given an ArrayList, the task is to get the first and last element of the ArrayList in Java.

```
Input: ArrayList = [1, 2, 3, 4]
Output: First = 1, Last = 4

Input: ArrayList = [12, 23, 34, 45, 57, 67, 89]
Output: First = 12, Last = 89
```

Approach:

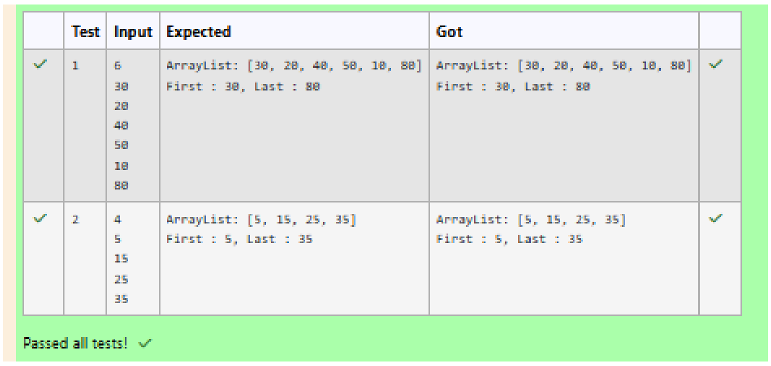
1. Get the ArrayList with elements.
2. Get the first element of ArrayList using the get(index) method by passing index = 0.
3. Get the last element of ArrayList using the get(index) method by passing index = size – 1.

SOLUTION :

```
i
m
p
o
r
t
```

```
      int numElements = scanner.nextInt();

      for
         (i
         nt
         i
         =
         0;
         i
         <
         n
         u
```

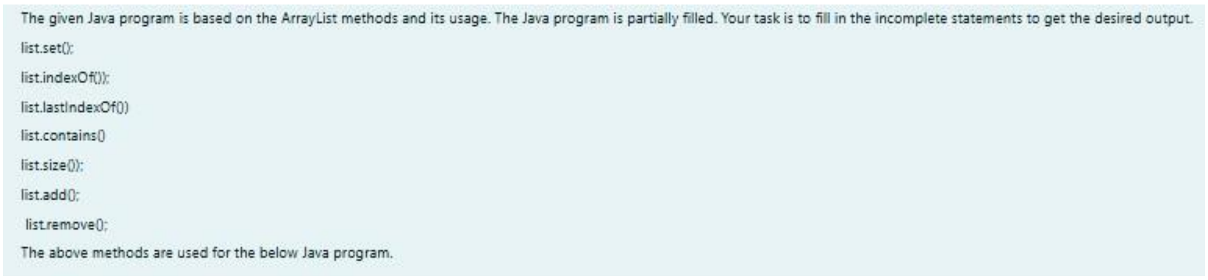
OUTPUT :



	Test	Input	Expected	Got	
✓	1	6 30 20 40 50 10 80	ArrayList: [30, 20, 40, 50, 10, 80] First : 30, Last : 80	ArrayList: [30, 20, 40, 50, 10, 80] First : 30, Last : 80	✓
✓	2	4 5 15 25 35	ArrayList: [5, 15, 25, 35] First : 5, Last : 35	ArrayList: [5, 15, 25, 35] First : 5, Last : 35	✓

Passed all tests! ✓

2.



The given Java program is based on the ArrayList methods and its usage. The Java program is partially filled. Your task is to fill in the incomplete statements to get the desired output.

list.set();

list.indexOf());

list.lastIndexOf())

list.contains()

list.size());

list.add();

list.remove();

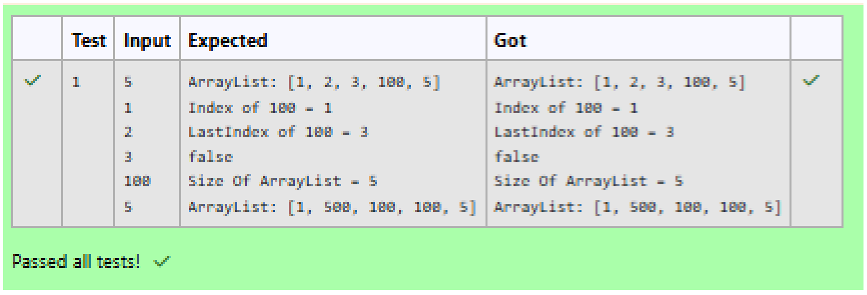
The above methods are used for the below Java program.

SOLUTION :

```
i
m
p
o
r
```

```
 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(3);                             // code here
  System.out.print("ArrayList: " + list);
 }
}
```

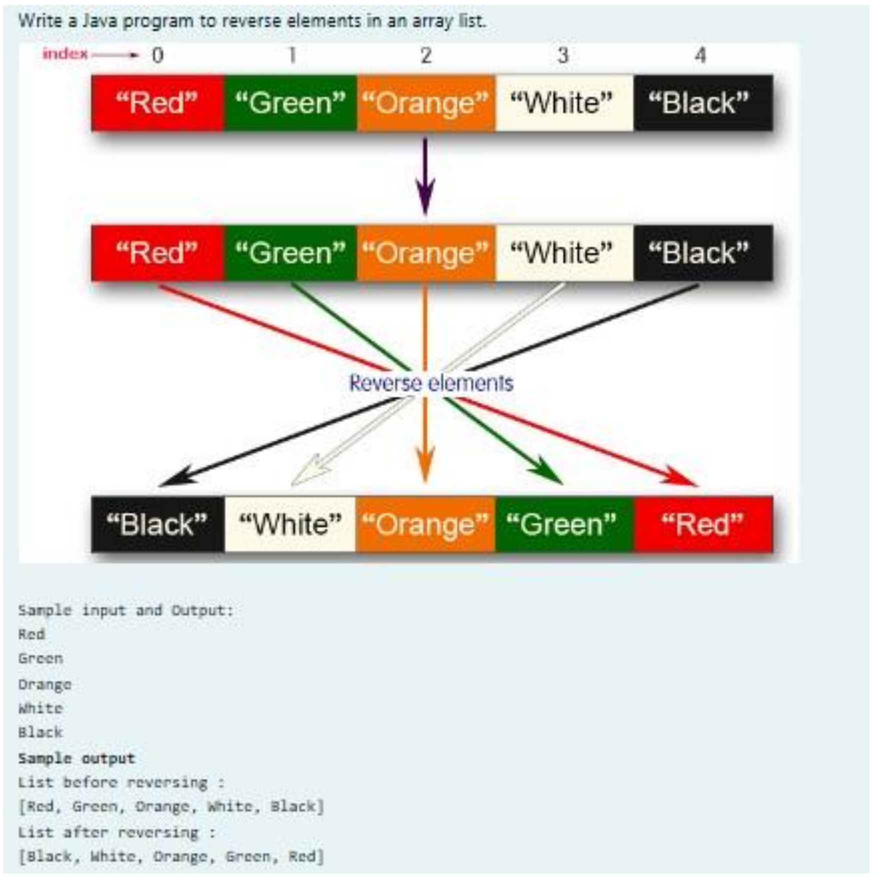
OUTPUT :



	Test	Input	Expected	Got	
✓	1	5 1 2 3 100 5	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	ArrayList: [1, 2, 3, 100, 5] Index of 100 = 1 LastIndex of 100 = 3 false Size Of ArrayList = 5 ArrayList: [1, 500, 100, 100, 5]	✓

Passed all tests! ✓

3.



Write a Java program to reverse elements in an array list.

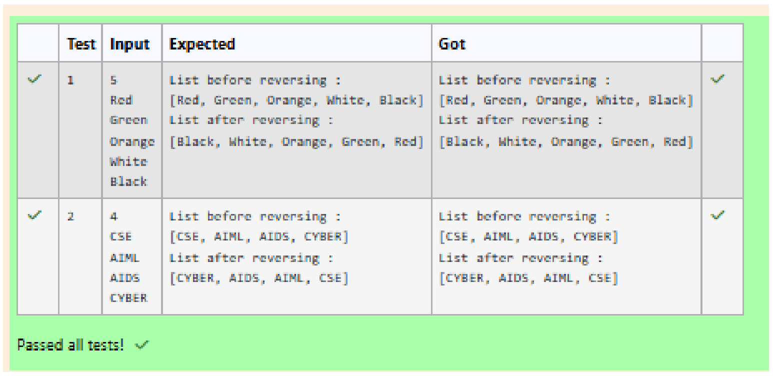
```
Sample input and Output:
Red
Green
Orange
White
Black
Sample output
List before reversing :
[Red, Green, Orange, White, Black]
List after reversing :
[Black, White, Orange, Green, Red]
```

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
t
i
```

```
    }
```

OUTPUT :

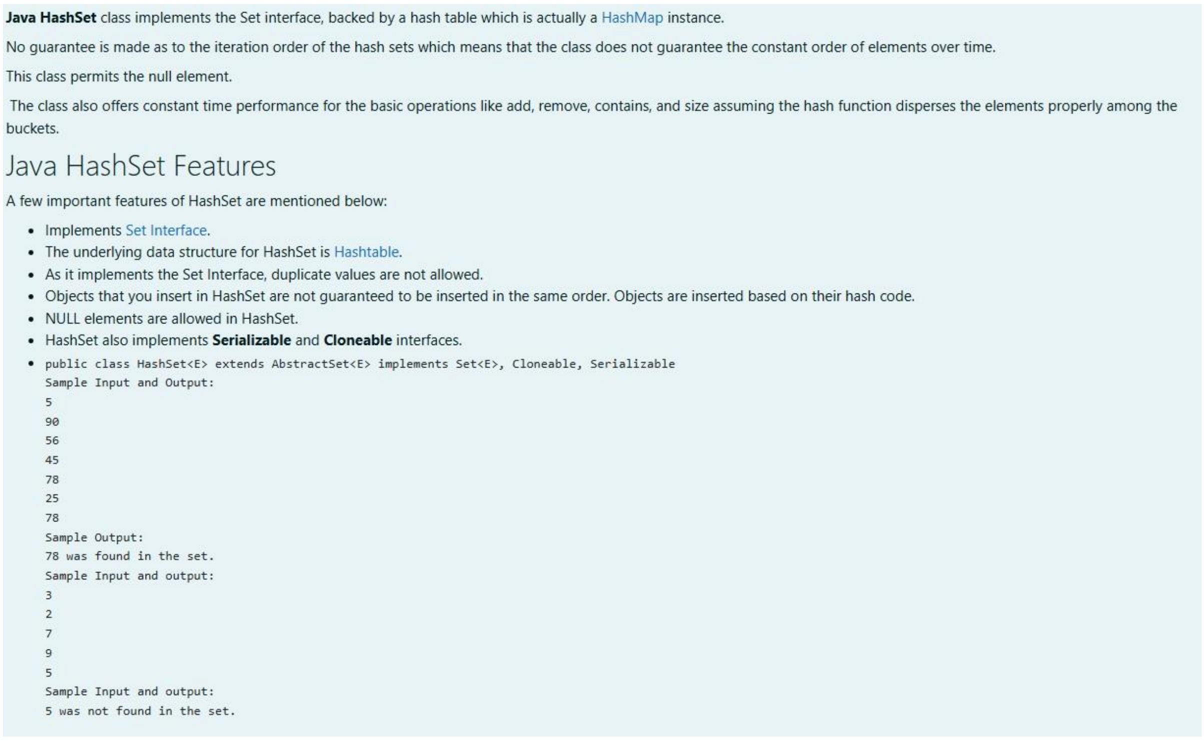


	Test	Input	Expected	Got	
✓	1	5 Red Green Orange White Black	List before reversing : [Red, Green, Orange, White, Black] List after reversing : [Black, White, Orange, Green, Red]	List before reversing : [Red, Green, Orange, White, Black] List after reversing : [Black, White, Orange, Green, Red]	✓
✓	2	4 CSE AIML AIDS CYBER	List before reversing : [CSE, AIML, AIDS, CYBER] List after reversing : [CYBER, AIDS, AIML, CSE]	List before reversing : [CSE, AIML, AIDS, CYBER] List after reversing : [CYBER, AIDS, AIML, CSE]	✓

Passed all tests! ✓

Lab-11-Set, Map

1.



Java HashSet class implements the Set interface, backed by a hash table which is actually a HashMap instance.

No guarantee is made as to the iteration order of the hash sets which means that the class does not guarantee the constant order of elements over time.

This class permits the null element.

The class also offers constant time performance for the basic operations like add, remove, contains, and size assuming the hash function disperses the elements properly among the buckets.

Java HashSet Features

A few important features of HashSet are mentioned below:

- Implements Set Interface.
- The underlying data structure for HashSet is Hashtable.
- As it implements the Set Interface, duplicate values are not allowed.
- Objects that you insert in HashSet are not guaranteed to be inserted in the same order. Objects are inserted based on their hash code.
- NULL elements are allowed in HashSet.
- HashSet also implements **Serializable** and **Cloneable** interfaces.

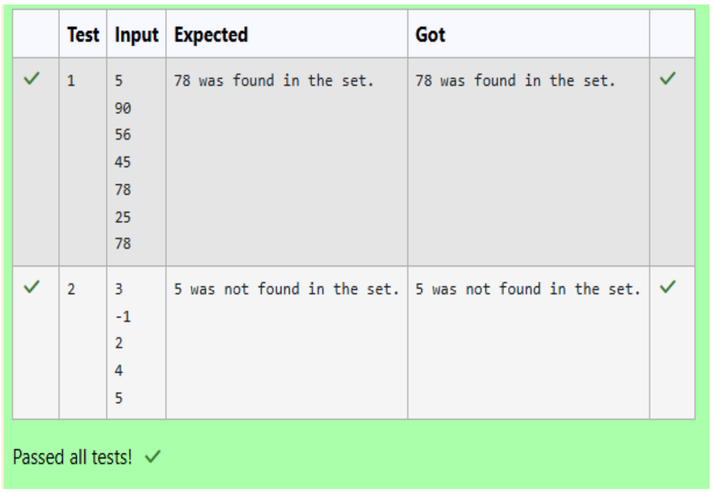
-
  ```
  public class HashSet<E> extends AbstractSet<E> implements Set<E>, Cloneable, Serializable
  Sample Input and Output:
  5
  90
  56
  45
  78
  25
  78
  Sample Output:
  78 was found in the set.
  Sample Input and output:
  3
  2
  7
  9
  5
  Sample Input and output:
  5 was not found in the set.
  ```

SOLUTION :

```
i
m
p
o
r
t
```

```
// Create a
HashSet object
to store
numbers
HashSet<Intege
r> numbers =
new
HashSet<>();

/
/

A
d
```

OUTPUT :



	Test	Input	Expected	Got	
✓	1	5 90 56 45 78 25 78	78 was found in the set.	78 was found in the set.	✓
✓	2	3 -1 2 4 5	5 was not found in the set.	5 was not found in the set.	✓

Passed all tests! ✓

2.

Write a Java program to compare two sets and retain elements that are the same.

Sample Input and Output:

```
5
Football
Hockey
Cricket
Volleyball
Basketball
7     // HashSet 2:
Golf
Cricket
Badminton
Football
Hockey
Volleyball
Handball
```

SAMPLE OUTPUT:

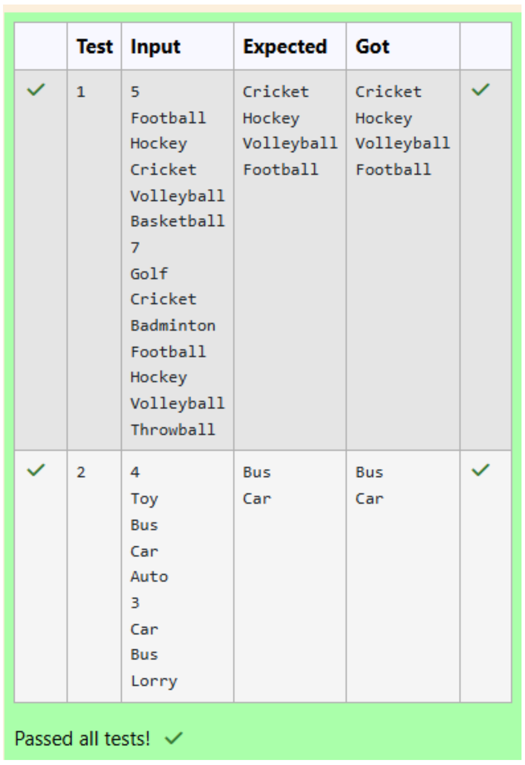
```
Football
Hockey
Cricket
Volleyball
Basketball
```

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
```

```
int size2 = Integer.parseInt(scanner.nextLine());

// Create a HashSet to
store the second set of
elementsSet<String> set2
= new HashSet<>();

//
Re
ad
ele
m
en
```

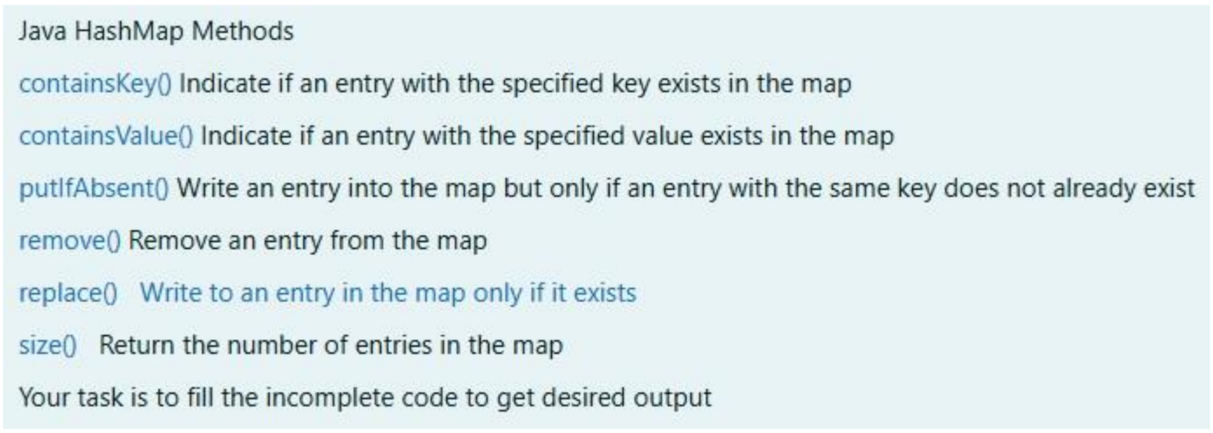
OUTPUT :



	Test	Input	Expected	Got	
✓	1	5 Football Hockey Cricket Volleyball Basketball 7 Golf Cricket Badminton Football Hockey Volleyball Throwball	Cricket Hockey Volleyball Football	Cricket Hockey Volleyball Football	✓
✓	2	4 Toy Bus Car Auto 3 Car Bus Lorry	Bus Car	Bus Car	✓

Passed all tests! ✓

3.


Java HashMap Methods

containsKey() Indicate if an entry with the specified key exists in the map

containsValue() Indicate if an entry with the specified value exists in the map

putIfAbsent() Write an entry into the map but only if an entry with the same key does not already exist

remove() Remove an entry from the map

replace() Write to an entry in the map only if it exists

size() Return the number of entries in the map

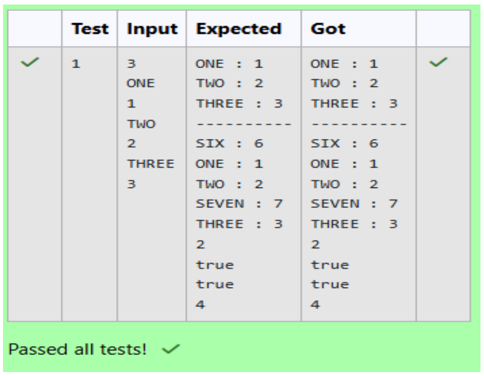
Your task is to fill the incomplete code to get desired output

SOLUTION :

```
i
m
p
o
r
t

j
a
v
a
.
u
t
i
l
.
H
a
s
h
M
```

```
        anotherMap.put("SEVEN", 7);

        // Inserting key-value pairs of map to anotherMap using putAll() method
        anotherMap.putAll(map); // This line fills in the missing code

        // 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); // Prints the value associated with key "TWO" (if it
exists)

        // Checking whether key 'ONE' exists in map
        System.out.println(map.containsKey("ONE")); // Prints true if "ONE" is a key,
false otherwise

        // Checking whether value '3' exists in map
        boolean valueExists = map.containsValue(3); // You can use a variable to store
the result
        System.out.println(valueExists); // Prints true if value 3 exists in the map, false
otherwise

        // Retrieving the number of key-value pairs present in map
        System.out.println(map.size()); // Prints the number of entries in the map
    }
}
```

OUTPUT :

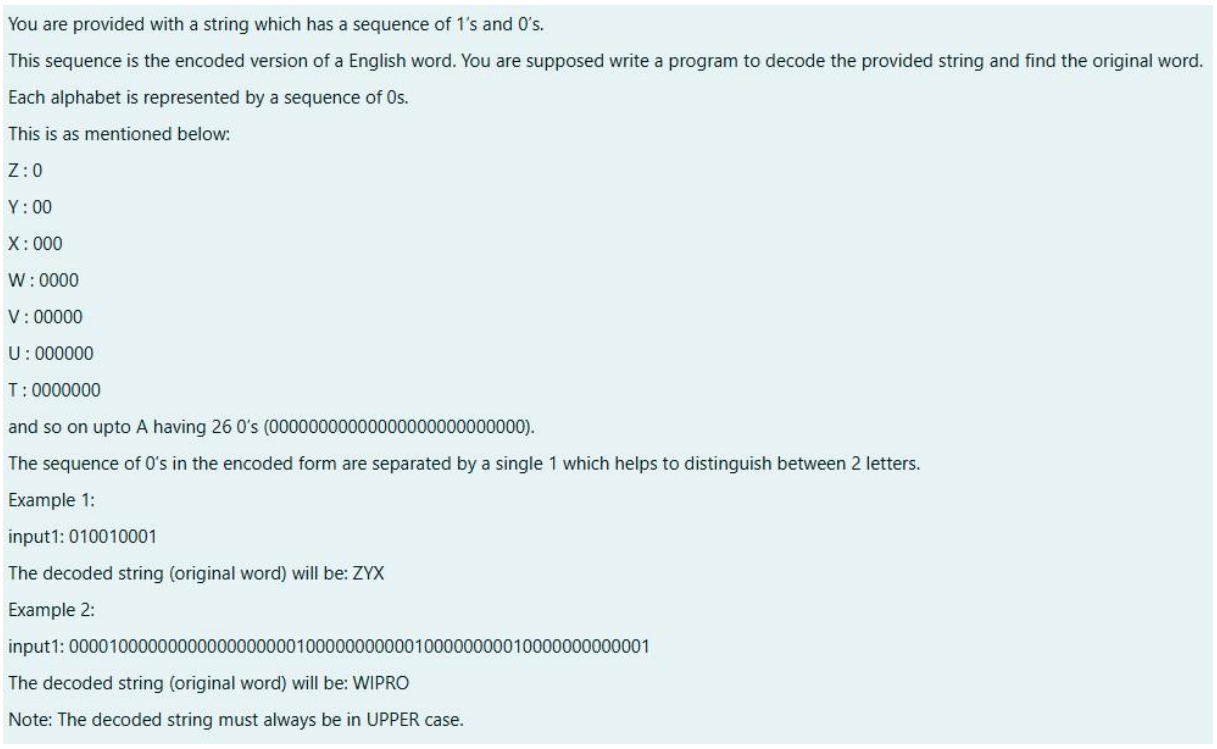


	Test	Input	Expected	Got	
✓	1	3 ONE 1 TWO 2 THREE 3	ONE : 1 TWO : 2 THREE : 3 ---------- SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	ONE : 1 TWO : 2 THREE : 3 ---------- SIX : 6 ONE : 1 TWO : 2 SEVEN : 7 THREE : 3 2 true true 4	✓

Passed all tests! ✓

Lab-12-Introduction to I/O, I/O Operations, Object Serialization

1.



You are provided with a string which has a sequence of 1's and 0's.

This sequence is the encoded version of a English word. You are supposed write a program to decode the provided string and find the original word.

Each alphabet is represented by a sequence of 0s.

This is as mentioned below:

Z : 0

Y : 00

X : 000

W : 0000

V : 00000

U : 000000

T : 0000000

and so on upto A having 26 0's (00000000000000000000000000).

The sequence of 0's in the encoded form are separated by a single 1 which helps to distinguish between 2 letters.

Example 1:

input1: 010010001

The decoded string (original word) will be: ZYX

Example 2:

input1: 0000100000000000000000010000000000010000000001000000000000001

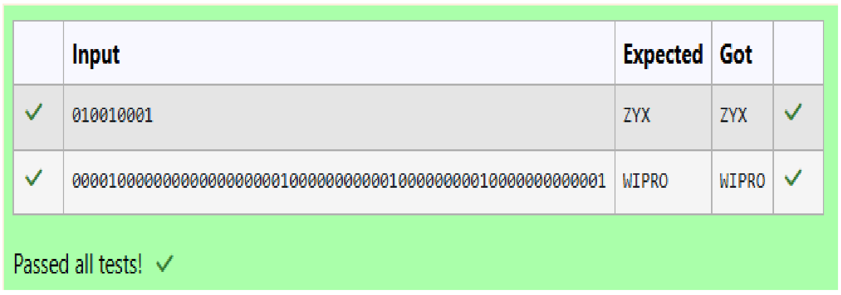
The decoded string (original word) will be: WIPRO

Note: The decoded string must always be in UPPER case.

SOLUTION :

```
import java.util.Scanner;

public class DecodeString {
    public static
        void
        main(String
        [] args) {
        Scanner
        scanner =
        new
        Scanner(Sys
        tem.in);
        String
        encodedStri
}
```

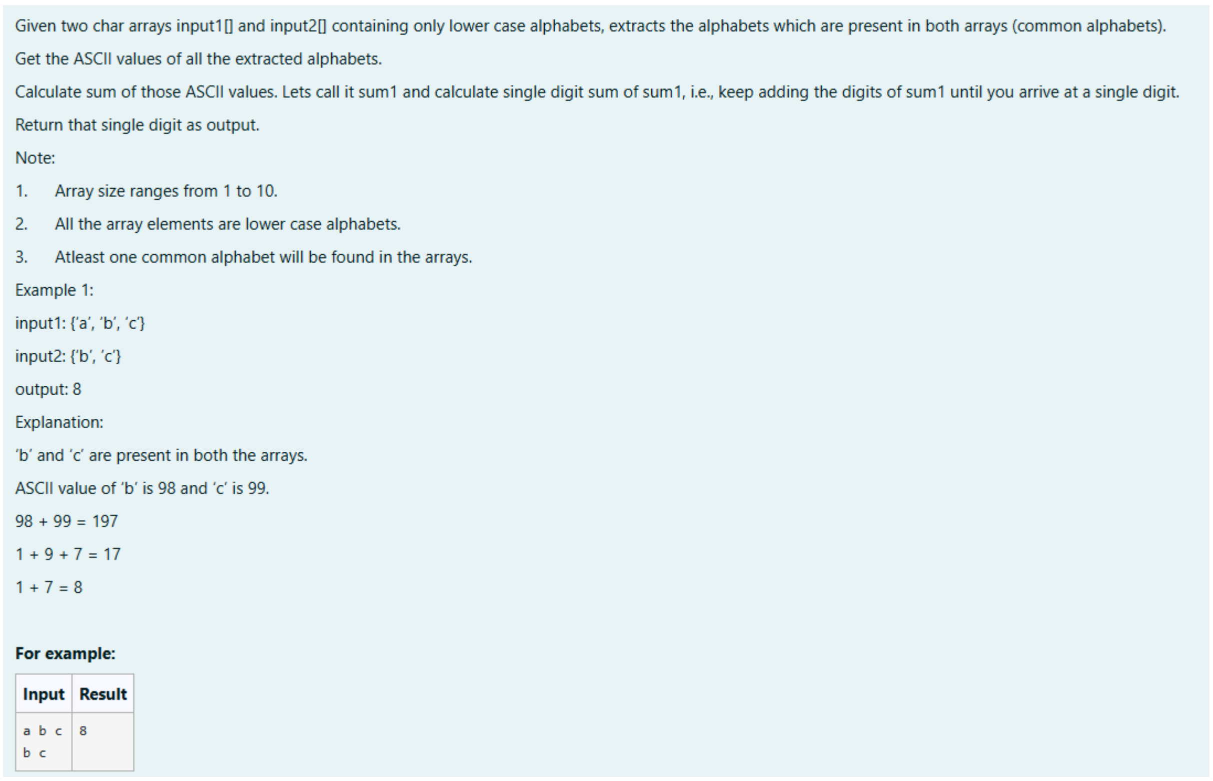
OUTPUT :



	Input	Expected	Got	
✓	010010001	ZYX	ZYX	✓
✓	0000100000000000000000000100000000000100000000001000000000001	WIPRO	WIPRO	✓

Passed all tests! ✓

2.



Given two char arrays input1[] and input2[] containing only lower case alphabets, extracts the alphabets which are present in both arrays (common alphabets).

Get the ASCII values of all the extracted alphabets.

Calculate sum of those ASCII values. Lets call it sum1 and calculate single digit sum of sum1, i.e., keep adding the digits of sum1 until you arrive at a single digit.

Return that single digit as output.

Note:

1. Array size ranges from 1 to 10.
2. All the array elements are lower case alphabets.
3. Atleast one common alphabet will be found in the arrays.

Example 1:

input1: {'a', 'b', 'c'}

input2: {'b', 'c'}

output: 8

Explanation:

'b' and 'c' are present in both the arrays.

ASCII value of 'b' is 98 and 'c' is 99.

98 + 99 = 197

1 + 9 + 7 = 17

1 + 7 = 8

For example:

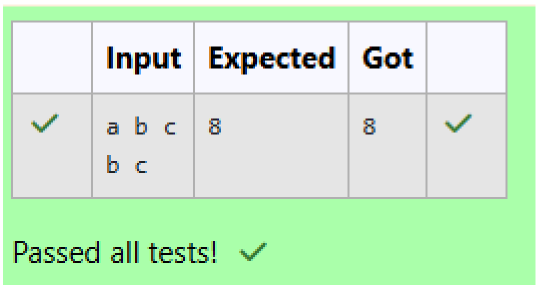
Input	Result
a b c b c	8

SOLUTION :

```
i
m
p
o
r
t

j
a
```

```
    return sum;
}

public static int
    calculateCommonAlphabetSum(char[]
    input1, char[] input2) {Set<Character> set1
    = new HashSet<>();
    f
        o
        r

        (
        c
        h
        a
        r
```

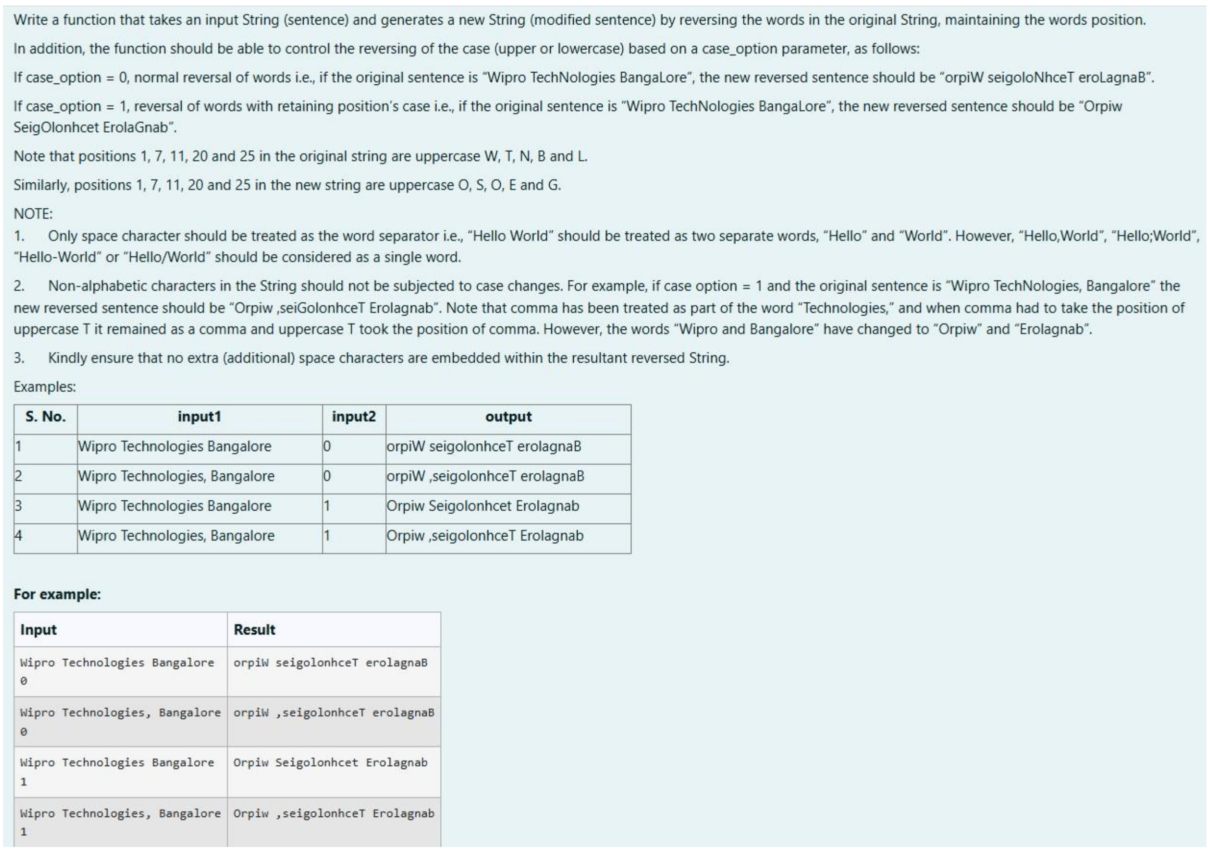
OUTPUT :



	Input	Expected	Got	
✓	a b c b c	8	8	✓

Passed all tests! ✓

3.



Write a function that takes an input String (sentence) and generates a new String (modified sentence) by reversing the words in the original String, maintaining the words position.

In addition, the function should be able to control the reversing of the case (upper or lowercase) based on a case_option parameter, as follows:

If case_option = 0, normal reversal of words i.e., if the original sentence is "Wipro TechNologies BangaLore", the new reversed sentence should be "orpiW seigoloNhceT eroLagnaB".

If case_option = 1, reversal of words with retaining position's case i.e., if the original sentence is "Wipro TechNologies BangaLore", the new reversed sentence should be "Orpiw SeigOlonhcet ErolaGnab".

Note that positions 1, 7, 11, 20 and 25 in the original string are uppercase W, T, N, B and L.

Similarly, positions 1, 7, 11, 20 and 25 in the new string are uppercase O, S, O, E and G.

NOTE:
1. Only space character should be treated as the word separator i.e., "Hello World" should be treated as two separate words, "Hello" and "World". However, "Hello,World", "Hello;World", "Hello-World" or "Hello/World" should be considered as a single word.

2. Non-alphabetic characters in the String should not be subjected to case changes. For example, if case option = 1 and the original sentence is "Wipro TechNologies, Bangalore" the new reversed sentence should be "Orpiw ,seiGolonhceT Erolagnab". Note that comma has been treated as part of the word "Technologies," and when comma had to take the position of uppercase T it remained as a comma and uppercase T took the position of comma. However, the words "Wipro and Bangalore" have changed to "Orpiw" and "Erolagnab".

3. Kindly ensure that no extra (additional) space characters are embedded within the resultant reversed String.

Examples:

S. No.	input1	input2	output
1	Wipro Technologies Bangalore	0	orpiW seigolonhceT erolagnaB
2	Wipro Technologies, Bangalore	0	orpiW ,seigolonhceT erolagnaB
3	Wipro Technologies Bangalore	1	Orpiw Seigolonhcet Erolagnab
4	Wipro Technologies, Bangalore	1	Orpiw ,seigolonhceT Erolagnab

For example:

Input	Result

`Wipro Technologies Bangalore`
`0`
	`orpiW seigolonhceT erolagnaB`

`Wipro Technologies, Bangalore`
`0`
	`orpiW ,seigolonhceT erolagnaB`

`Wipro Technologies Bangalore`
`1`
	`Orpiw Seigolonhcet Erolagnab`

`Wipro Technologies, Bangalore`
`1`
	`Orpiw ,seigolonhceT Erolagnab`

SOLUTION :

i

m

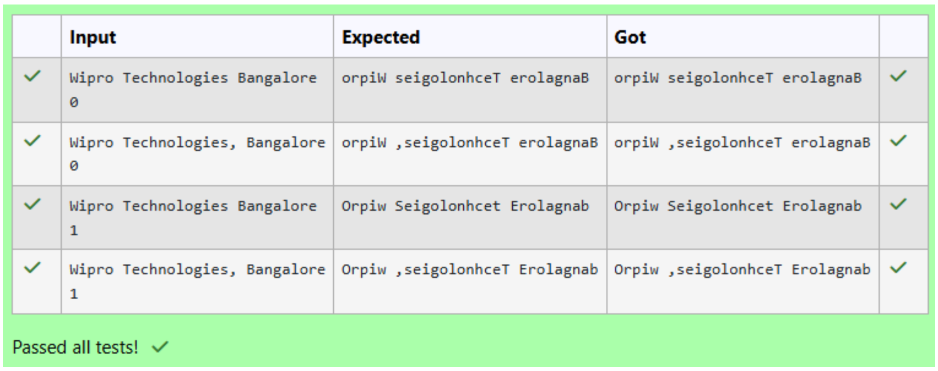
p

o

r

t

```
            result.append(applyCaseConversion(reversedWord, word)).append(" ");
         }
      }

      // Remove the trailing space and return the result
      return result.toString().trim();
   }

   private static String applyCaseConversion(String reversedWord, String
originalWord) {
      // StringBuilder to store the adjusted word
      StringBuilder adjustedWord = new StringBuilder();

      // Iterate over each character in the reversed word
      for (int i = 0; i < reversedWord.length(); i++) {
         char reversedChar = reversedWord.charAt(i);
         char originalChar = originalWord.charAt(i);

         if (Character.isLowerCase(originalChar)) {
            // If the original character was lowercase, the reversed character should be
uppercase
            adjustedWord.append(Character.toLowerCase(reversedChar));
         } else if (Character.isUpperCase(originalChar)) {
            // If the original character was uppercase, the reversed character should be
lowercase
            adjustedWord.append(Character.toUpperCase(reversedChar));
         } else {
            // Non-alphabetic characters remain unchanged
            adjustedWord.append(reversedChar);
         }
      }

      return adjustedWord.toString();
   }

   public static void main(String[] args) {
      // Create a Scanner object to get input from the user
      Scanner scanner = new Scanner(System.in);

      // Get sentence input from the user

      String sentence = scanner.nextLine();

      // Get case option input from the user

      int caseOption = scanner.nextInt();

      // Validate the case option
      if (caseOption != 0 && caseOption != 1) {
        System.out.println("Invalid case option. Please
        enter 0 or 1.");
    } else {
        // Call the function and print the result
        String result =
        reverseWordsWithCase(sente
        nce, caseOption);
```
OUTPUT :



	Input	Expected	Got	
✓	`Wipro Technologies Bangalore` `0`	`orpiW seigolonhceT erolagnaB`	`orpiW seigolonhceT erolagnaB`	✓
✓	`Wipro Technologies, Bangalore` `0`	`orpiW ,seigolonhceT erolagnaB`	`orpiW ,seigolonhceT erolagnaB`	✓
✓	`Wipro Technologies Bangalore` `1`	`Orpiw Seigolonhcet Erolagnab`	`Orpiw Seigolonhcet Erolagnab`	✓
✓	`Wipro Technologies, Bangalore` `1`	`Orpiw ,seigolonhceT Erolagnab`	`Orpiw ,seigolonhceT Erolagnab`	✓

Passed all tests! ✓