SONA COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SONA NAGAR, SALEM-636 005

[An Autonomous Institution]

Name	:	**NITHISH R**
Branch	:	**ADS**
Semester	:	**4**
Register No	:	**1920110032**

Certified that this is the bona-fide record of practical work done by the above student in **U19ADS407 – JAVA PROGRAMMING LAB** during the year **2021 -2022**.

Lab in-Charge Head of the Department

Submitted for University Practical Examination held on …………………………………………….

Internal Examiner External Examiner

INDEX

EXERCISE : 1 **INTRODUCTION**

Ex No: E1a DATE: 04-03-2022

1.Welcome to the world of Java! In this challenge, we practice printing to stdout.

The code stubs in your editor declare a *Solution* class and a *main* method. Complete the *main* method by copying the two lines of code below and pasting them inside the body of your *main* method.

```
System.out.println("Hello, World.");
System.out.println("Hello, Java.");
```

Input Format

There is no input for this challenge.

Output Format

You must print two lines of output:

1. Print Hello, World. on the first line.
2. Print Hello, Java. on the second line.

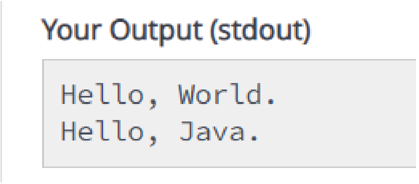
Sample Output

```
Hello, World.
Hello, Java.
```

Program Code:

```
public class Solution {


   public static void main(String[] args) {
      System.out.println("Hello, World.");
      System.out.println("Hello, Java.");
   }
}
```

Output:



Your Output (stdout)

```
Hello, World.
Hello, Java.
```

Result: Thus the above program is compiled and executed successfully.

2. Most HackerRank challenges require you to read input from stdin (standard input) and write output to stdout (standard output).

One popular way to read input from stdin is by using the Scanner class and specifying the *Input Stream* as *System.in*. For example:

```
Scanner scanner = new Scanner(System.in);
String myString = scanner.next();
int myInt = scanner.nextInt();
scanner.close();


System.out.println("myString is: " + myString);
System.out.println("myInt is: " + myInt);
```

The code above creates a *Scanner* object named and uses it to read a *String* and an *int*. It then *closes* the *Scanner* object because there is no more input to read, and prints to stdout using *System.out.println(String)*. So, if our input is:

```
Hi 5
```

Our code will print:

```
myString is: Hi
myInt is: 5
```

Alternatively, you can use the BufferedReader class.

Task

In this challenge, you must read integers from stdin and then print them to stdout. Each integer must be printed on a new line. To make the problem a little easier, a portion of the code is provided for you in the editor below.

Input Format

There are lines of input, and each line contains a single integer.

Sample Input

```
42
100
125
```

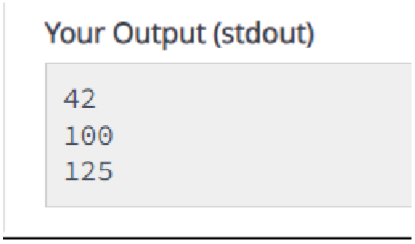
Sample Output

```
42
100
125
```

Program Code:

```
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int a = scan.nextInt();
        int b= scan.nextInt();
        int c= scan.nextInt();

        System.out.println(a);
        System.out.println(b);
        System.out.println(c);
        // Complete this line
        // Complete this line
    }
}
```

Output:



Your Output (stdout)

```
42
100
125
```

Result: Thus the above program is compiled and executed successfully.

3. Rima picks some amount of flowers from her garden and holds with both of her hands, and she needs to know the total number of flowers she picked. Write a program to help Rima.

Input Format

Each line consists of a number. Flowers picked in right hand and flowers picked in left hand.

Constraints

Nil

Output Format

Integer value (Flower picked by her)

Sample Input 0

```
3
```

```
1
```

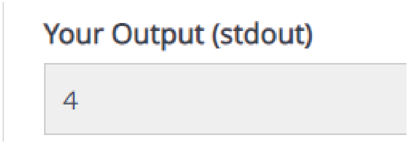
Sample Output 0

```
4
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int  b= s.nextInt();
        int c = a+b;
        System.out.println(c);
    }
}
```

Output:



Your Output (stdout)

```
4
```

Result: Thus the above program is compiled and executed successfully.

4. John faced a difficulty in calculating the purchase value of same product with varied quantity. So he tries to transfer this multiplication job to his robot.

Help John by writing a program to the robot to reduce his difficulty.

Input Format

Two positive integers which are cost and number of the product purchased in each line

Constraints

Nil

Output Format

Single positive integer which is the quantitative total value of that product purchased

Sample Input 0

```
10
8
```

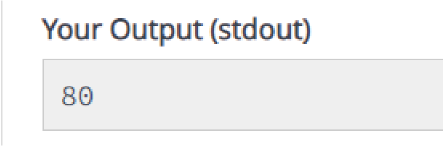
Sample Output 0

```
80
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int b = s.nextInt();
        int c = a*b;
        System.out.println(c);
    }
}
```

Output:



Your Output (stdout)

```
80
```

Result: Thus the above program is compiled and executed successfully.

5. Vivek's sister is studying in 4th standard. She is struggling to score marks in mathematics. So Vivek decided to help her by teaching basic multiplication to her. He explained themultiplication concept and asked her sister to multiply three numbers. Meantime Vivek writesa program to multiply three numbers. Help Vivek to write a suitable program.

Note : Display the result with two decimals

Input Format

Three numbers in first line separated by space

Constraints

Nil

Output Format

Product of three numbers as shown in the sample output

Sample Input 0

```
3 4 5
```

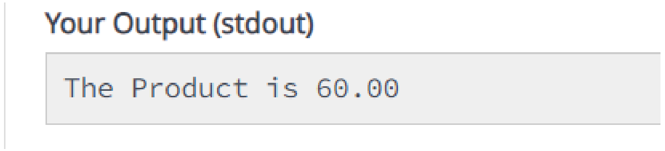
Sample Output 0

```
The Product is 60.00
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int b = s.nextInt();
        int c = s.nextInt();
        Double ans = a*b*c*1.0;
        System.out.printf("The Product is %.2f",ans);
    }
}
```

Output:



Your Output (stdout)

```
The Product is 60.00
```

Result: Thus the above program is compiled and executed successfully.

6. John made a huge profit out of his grocery shop. But, his profit got reduced due to his wrong calculation in providing the balance amount to his customers.

Once he started to realize his mistakes, he is trying to make out this work with his robot. So that his profit is safe.

He finds you to provide a programming solution to his problem. Get purchase value and cash given by the customer and calculate the balance amount.

Input Format

Two positive integers,

First one is the purchase value and other one is the Cash given by the customer in each line

Constraints

Nil

Output Format

Numerical value which is the balance amount to be given back to the Customer

Sample Input 0

```
28
100
```

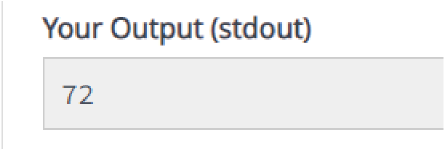
Sample Output 0

```
72
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int b = s.nextInt();
        int c = b-a;
        System.out.println(c);
    }
}
```

Output:



Your Output (stdout)

```
72
```

Result: Thus the above program is compiled and executed successfully.

7. John's new robot in his Grocery Shop has attracted many customers. Now he plans to make the robot to wish a customized welcome message for his regular customers. Write a program tohelp him in the task.

Input Format

Input is a name of a person [character array]

Constraints

Nil

Output Format

Welcome message as shown in sample output

Sample Input 0

```
Karthic
```

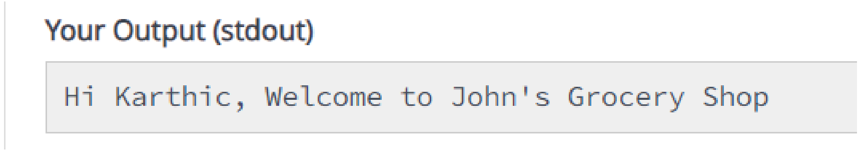
Sample Output 0

```
Hi Karthic, Welcome to John's Grocery Shop
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        String my = scanner.next();
        System.out.println("Hi "+my+", Welcome to John's Grocery Shop");
    }
}
```

Output:



Your Output (stdout)

```
Hi Karthic, Welcome to John's Grocery Shop
```

Result: Thus the above program is compiled and executed successfully.

Ex No: E1b DATE: 11-03-2022

1. Write a Java program that add two numbers.

Input Format

first line contains integer X second line contains integer Y

Constraints

X>0 Y>100

Output Format

integer output

Sample Input 0

```
2
3
```

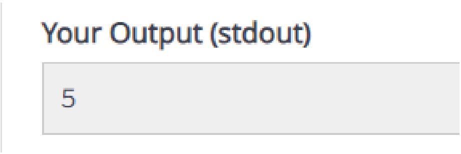
Sample Output 0

```
5
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class Solution {
   public static void main(String args[] ) throws Exception {
      Scanner s = new Scanner(System.in);
      int a = s.nextInt();
      int b = s.nextInt();
      System.out.println(a+b);
   }
}
```

Output:



Your Output (stdout)

```
5
```

Result: Thus the above program is compiled and executed successfully.

2. Write a Java program to get two integer numbers from the user, and return their product value if the product value is smaller than 1000 else return their sum.

Input Format

20

30

Constraints

number1,number2<1000

Output Format

600

Sample Input 0

```
40
30
```

Sample Output 0

```
70
```

Sample Input 1

```
10000
100000
```

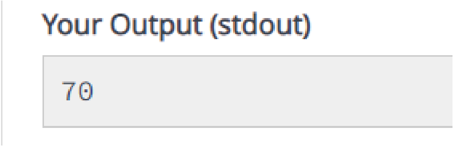
Sample Output 1

```
Invalid Numbers
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int b = s.nextInt();
        if(a>1000 || b>1000){
            System.out.println("Invalid Numbers");
        }
        else{
            int c = a*b;
            if(a*b<1000){
                System.out.println(c);
            }
            else{
                System.out.println(a+b);
            }
        }
    }
}
```

Output:



Your Output (stdout)

```
70
```

Result: Thus the above program is compiled and executed successfully.

3. Write a Java program that asks the user to enter two numbers x and y. If x is greater than y print back to the user "x>y". If x is lesser than y print " x < y ".

Input Format

Get two numbers in a single statement

Constraints

x,y<=1000

Output Format

Output as a string as mentioned in the scenario

Sample Input 0

```
10 20
```

Sample Output 0

```
x<y
```

Sample Input 1

```
1001 2001
```

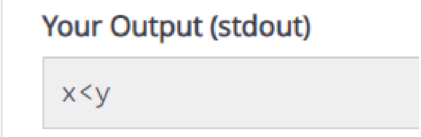
Sample Output 1

```
Invalid
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner s = new Scanner(System.in);
        int a = s.nextInt();
        int b = s.nextInt();
        if(a>1000 || b>1000){
            System.out.println("Invalid");
        }
        else {
            if(a<=b){
                System.out.println("x<y");
            }
            else{
                System.out.println("x>y");
            }
        }
    }
}
```

Output:



Your Output (stdout)

```
x<y
```

Result: Thus the above program is compiled and executed successfully.

4. Write a Java program that asks the user for two numbers (use two different input lines). If the second number the user enters is zero, print back to the user "can't divide by zero", otherwise divide the user's first number by the user's second number and print the result to the user.

Input Format

Get input as integer in two lines

Constraints

1

Output Format

Result should be in integer format

Sample Input 0

```
20
10
```

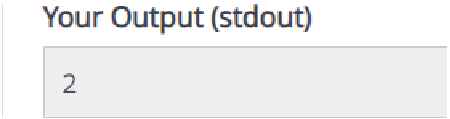
Sample Output 0

```
2
```

Program Code:

```
a = int(input())
b = int(input())
if(b != 0):
    print(int(a/b))
else:
    print("can't divide by zero")
```

Output:



```
2
```

Result: Thus the above program is compiled and executed successfully.

5. After Vivek's coaching, his sister's performance in maths has improved very much. So, Vivek wants to test her knowledge of geometry. So he asked her sister to calculate, Hypotenuse, Area and Perimeter of a right-angled triangle. He will give values of two sides.

Note : Use suitable formula to calculate Hypotenuse, Area and Perimeter

Input Format

Length of Side A

Length of Side B

Constraints

Nil

Output Format

Display area and Perimeter as shown in sample output

Sample Input 0

```
3
4
```

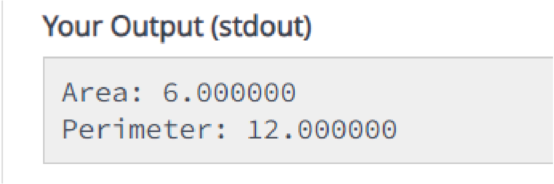
Sample Output 0

```
Area: 6.000000
Perimeter: 12.000000
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.lang.*;

public class Solution {

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        Double a = s.nextDouble();
        Double b = s.nextDouble();
        System.out.printf("Area: %f\n", (a*b)/2);
        System.out.printf("Perimeter: %f", (a+b+Math.sqrt((a*a)+(b*b))));
    }
}
```

Output:



Your Output (stdout)

```
Area: 6.000000
Perimeter: 12.000000
```

Result: Thus the above program is compiled and executed successfully.

Exercise 2: LOOPS

Ex No: E2a DATE: 18-03-2022

1. You will be given a number as input. You have to split the digits of the number and then you have to group the odd and even digits.

Input Format

2983

Constraints

Input number will be of any length

Output Format

28 93

Sample Input 0

```
2983
```

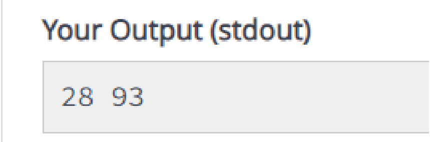
Sample Output 0

```
28 93
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String a = sc.nextLine();
        String odd = "",even = "";
        int n = a.length(), el = 0, ol = 0;

        for(int i=0; i<n; i++){
            // int x = a.substring(i,i+1);
            Integer x = 0;
            try{
                x = Integer.valueOf(a.substring(i,i+1));
                // System.out.println(x); // output = 25
            }
            catch (NumberFormatException ex){
              ex.printStackTrace();
            }
            if((x)%2 == 0){
                even += a.substring(i,i+1);
                el += 1;
            }
            else{
                odd += a.substring(i,i+1);
                ol += 1;
            }
        }
        if(el == 0){
            System.out.println(odd);
        }
        else if(ol == 0){
            System.out.println(even);
        }
        else{
            System.out.println(even+" "+odd);
        }
    }
}
```

Output:



Your Output (stdout)

```
28 93
```

Result: Thus the above program is compiled and executed successfully.

2. A Hailstone series is defined as follows: start with any integer value greater than 0, say x. If x is even, then the next value in the series is x/2; if x is odd, then the next value in the series is 3x + 1. Now apply the same rules to create the next value in the series, and so on.

Input Format

Starting number of the series in the first line

Number of terms to generate in the series in second line

Constraints

Nil

Output Format

Hailstone series as shown in sample output.

Sample Input 0

```
17
5
```

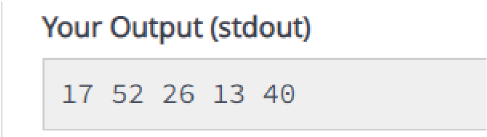
Sample Output 0

```
17 52 26 13 40
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        System.out.print(a+" ");
        for(int i=0; i<b-1; i++){
            if(a%2 == 0){
                a = a/2;
                System.out.print(a+" ");
            }
            else{
                a = (3*a)+1;
                System.out.print(a+" ");
            }
        }
    }
}
```

Output:



```
17 52 26 13 40
```

Result:

Thus the above program is compiled and executed successfully.

3. Janu and Banu are playing a mathematical game. Janu wants to test Banu's calculation speed. So she decided to ask a question which includes a sequence of addition. Janu writes below series on the paper. If she tells a number (n), Banu needs to calculate the answer and tell. Help Banu by writing a suitable code.

1+(1+2)+(1+2+3) + …. (1+2+3…+n)

Input Format

N value as integer in first line

Constraints

Nil

Output Format

Result after calculation

Sample Input 0

```
10
```

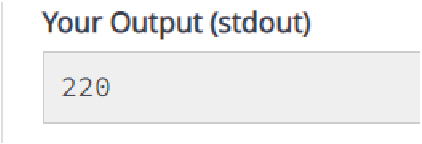
Sample Output 0

```
220
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int n = a, sum = 0;
        for(int i=n; i>0; i--){
            sum += (i*(i+1))/2;
        }
        System.out.println(sum);
    }
}
```

Output:



Your Output (stdout)

```
220
```

Result: Thus the above program is compiled and executed successfully.

4. Print the given pattern when n is 4 a b b c c c d d d d

Input Format

Input contains number of rows 'n'

Constraints

1 <= n <= 26

Output Format

Output contains required pattern with alphabets in sequence. Print Invalid if n>26.

Sample Input 0

```
3
```

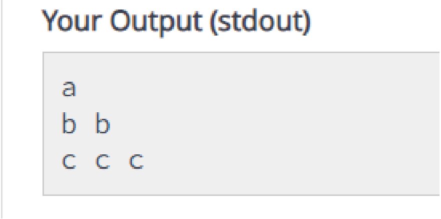
Sample Output 0

```
a
b b
c c c
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class SOlution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        if(n>26){
            System.out.println("Invalid");
        }
        else{
            int x = n;
            String s = "abcdefghijklmnopqrstuvwxyz";
            // char c = 'a';
            for(int i=0; i<x; i++){
                for(int j=0; j<=i; j++){
                    System.out.print(s.charAt(i)+" ");
                }
                System.out.print("\n");
            }
        }
    }
}
```

Output:



Your Output (stdout)

```
a
b b
c c c
```

Result: Thus the above program is compiled and executed successfully.

5. Create a unique ID which is of lowercase 9 digit from the given strings S1 and S2. To make a Unique ID it should contain the sequence of first three letters of the first string, last three letters of second string and ASCII value of last character(lower case) of the first string S1 which should be 3 digit. For example, S1=abcdef, S2=helloAll then Unique ID= abcall102.

Input Format

First line contains String S1 Second line contains String S2

Constraints

3<=S1<=100 3<=S2<=100

Output Format

Unique ID of length 9

Sample Input 0

```
abcd
efgh
```

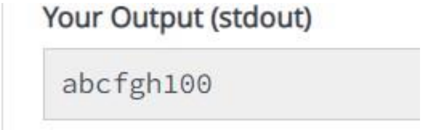
Sample Output 0

```
abcfgh100
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String a = sc.next();
        String b = sc.next();
        int na = a.length();
        int nb = b.length();
        if(na>2 && nb>2){
            System.out.print(a.substring(0,3));
            System.out.print(b.substring(nb-3));
            int x = a.charAt(na-1);
            if(x<100){
                System.out.print(0);
                System.out.print(x);
            }
            else{
                System.out.print(x);
            }
        }
        else{
            System.out.println("Invalid");
        }
    }
}
```

Output:



Your Output (stdout)

```
abcfgh100
```

Result: Thus the above program is compiled and executed successfully.

Ex No: E2b DATE: 23-03-2022

1. Decode the logic and print the pattern that corresponds to the given input.

Input Format

Single Integer N

Constraints

2<=N<=100

Output Format

Output is the form of pattern

Sample Input 0

```
3
```

Sample Output 0

```
10203010011012
**4050809
****607
```

Sample Input 1

```
4
```

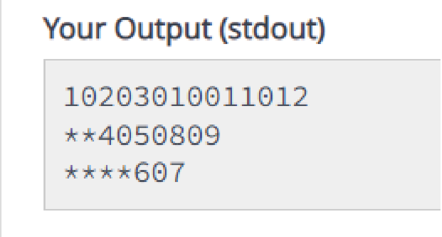
Sample Output 1

```
1020304017018019020
**50607014015016
****809012013
******10011
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
public class Main {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        String s[] = new String[1000];
        int temp = a, ans = 10;
        for(int i=0; i<a; i++){
            s[i] = "";
            for(int j=0; j<temp; j++){
                s[i] += ans;
                ans += 10;
            }
            temp--;
        }
        temp = 1;
        for(int i=a-1; i>=0; i--){
            for(int j=0; j<temp; j++){
                if(j == temp-1){
                    s[i] += ans/10;
                }
                else{
                    s[i] += ans;
                }
                ans += 10;
            }
            temp++;
        }
        for(int i=0; i<a; i++){
            for(int j=0;j<2*i;j++){System.out.print("*");}
            System.out.print(s[i]+"\n");
        }
    }
}
```

Output:


Your Output (stdout)

```
10203010011012
**4050809
****607
```

Result: Thus the above program is compiled and executed successfully.

2. Given a number ‘N’, find all the possible divisors and print them in increasing order.

Input Format

The first line of input contains an integer 'N' denoting the number.

Constraints

1 <= N <= 100000

Output Format

Print a line containing all the divisors in increasing order separated by space.

Sample Input 0

```
10
```

Sample Output 0

```
1 2 5 10
```

Sample Input 1

```
20
```

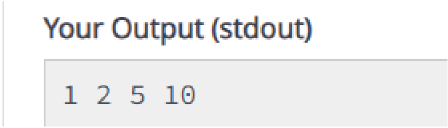
Sample Output 1

```
1 2 4 5 10 20
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        for (int i=1; i<=n; i++){
          if (n % i == 0) {
            System.out.print(i + " ");
          }
        }
    }
}
```

Output:



```
1 2 5 10
```

Result: Thus the above program is compiled and executed successfully.

3. A positive integer d is said to be a factor of another positive integer N if when N is divided by d, the remainder obtained is zero. For example, for number 12, there are 6 factors 1, 2, 3, 4, 6, 12. Every positive integer k has at least two factors, 1 and the number k itself.

Given two positive integers N and K, write a program to print the kth largest factor of N.

Input Format

The first line of input contains comma separated list of positive integer pairs(N, K)

Constraints

1 < N < 10000000000.

1 < K < 600

Output Format

Print the kth highest factor of N. If N does not have k factors, the output should be 1.

Sample Input 0

```
12 3
```

Sample Output 0

```
4
```

Sample Input 1

```
65 2
```

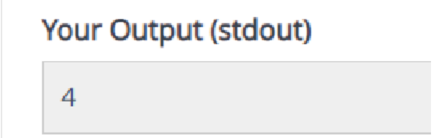
Sample Output 1

```
13
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution {

   public static void main(String args[] ) throws Exception {

       Scanner sc = new Scanner(System.in);

       int n = sc.nextInt();

       int x = sc.nextInt();

       int a = 1;

       for (int i=n; i>0; i--){

          if (n % i == 0) {

              if(a==x){

                  System.out.print(i);

                  return;

                  // System.out.println("nidhsfdsf");

              }

              a++;

          }

       }

       System.out.println(1);

   }

}
```

Output:



Your Output (stdout)

```
4
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 3 ARRAYS

Ex No: E3a DATE: 26-03-2022

1. Kriti has two numbers, her tution teacher give a task to her. The task is that, she wants to find the sum of all digits appearing between those two numbers.

For Ex:

Num1 = 8 and Num2 = 13

Output: 27

(8 + 9 + 1 + 0 + 1 + 1 + 1 + 2 + 1 +3)

Input Format

Input the Number1 Input the Number2

Constraints

0 <= Number1 <=10000

Number1 <= Number2 <=10000

(Negative Numbers and Decimals are not allowed)

Output Format

Display the Sum of all digits appearing between Number1 and Number2

Sample Input 0

```
8
13
```

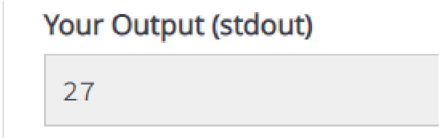
Sample Output 0

```
27
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int sum = 0;
        for(int i=a; i<=b; i++){
            int x = i;
            while(x != 0){
                int rem = x%10;
                sum += rem;
                x /= 10;
            }
        }
        System.out.println(sum);
    }
}
```

Output:



Your Output (stdout)

```
27
```

Result: Thus the above program is compiled and executed successfully.

2. Given sorted list, check the sum of two numbers in the sorted list is a given value.

Input Format

First line of input contains sorted list. Next line contains a value 'v'

Constraints

- 2 < list length < 100
- 2 < v < 100000

Output Format

Output should contain True if sum of list values gives 'v' else False

Sample Input 0

```
1 3 4 8 10
7
```

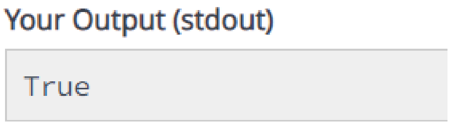
Sample Output 0

```
True
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc=new Scanner(System.in);
        String s = sc.nextLine();
        String[] b = s.split(" ", 120);
        int x = sc.nextInt();
        int n = 0, k = 0;
        int[] a = new int[200];

        for (String d : b){

            a[k] = Integer.parseInt(d);
            k++;
            n++;
        }

        for(int i=0; i<n-1; i++){
            for(int j=i+1; j<n; j++){
                if(a[i] + a[j] == x){
                    System.out.println("True");
                    return;
                }
            }
        }
        System.out.println("False");
    }
}
```

Output:



```
True
```

Result: Thus the above program is compiled and executed successfully.

3. Given an array of integers . Find nearest square root value for each element in the given array and sum up all the nearest square root value.

Input Format

Array of integers

Constraints

- 1 < arraysize< 10000

Output Format

Single integer which is the sum of all nearest square root values.

Sample Input 0

```
66 17 47 4
```

Sample Output 0

```
21
```

Explanation 0

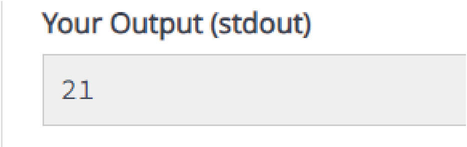
- 64 is nearest square to 66. Hence 8
- 16 is nearest square to 17. Hence 4
- 49 is nearest square to 47. Hence 7
- 4 is a perfect square. Hence 2
So, 8+4+7+2 = 21

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;
```

```
public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc=new Scanner(System.in);
        String s = sc.nextLine();
        String[] t = s.split(" ", 120);
        int n = 0, k = 0, sum = 0;
        int[] a = new int[200];

        for (String d : t){
            a[k] = Integer.parseInt(d);
            k++;
            n++;
        }
        for(int i=0; i<n; i++){
            int num = a[i];
            double cen = Math.sqrt(num);
            int p = (int) Math.ceil(cen), q = (int) Math.floor(cen);
            int difofx = Math.abs(p*p-num), difofy = Math.abs(q*q-num);
            int ans;
            if(difofx > difofy){
                ans = q;
            }
            else{
                ans = p;
            }
            sum+=ans;
        }
        System.out.println(sum);
    }
}
```

Output:



Your Output (stdout)

```
21
```

Result: Thus the above program is compiled and executed successfully.

4. Print the given pattern * * * * * *

Input Format

Input contains number of rows 'n'

Constraints

1 <= n <= 100

Output Format

Output contains required pattern with 'n' rows

Sample Input 0

```
4
```

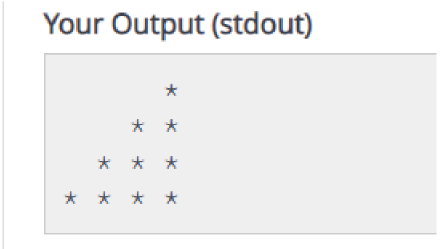
Sample Output 0

```
   *
  * *
 * * *
* * * *
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int x = a-1;
        int t = 1;
        for(int i=0; i<a; i++){
            for(int j=0; j<x; j++){
                System.out.print("  ");
            }
            x--;
            for(int j=0; j<t; j++){
               System.out.print("* ");
            }
            t++;
            System.out.print("\n");

        }
    }
}
```

Output:



Your Output (stdout)

```
      *
    * *
  * * *
* * * *
```

Result: Thus the above program is compiled and executed successfully.

5. Rakesh has an integer I and a Positive integer P with GCD(I, P)=1, find the multiplicative order of a module P is the smallest positive integer S with I^S(mod P)=1 (0 < S < P).

Input : I = 4 , P = 7

Output : 3

explanation :

GCD(4, 7) = 1

I^S(mod P) = 1 (smallest positive integer S)

4^1 = 4(mod 7) = 4

4^2 = 16(mod 7) = 2

4^3 = 64(mod 7) = 1

4^4 = 256(mod 7) = 4

```
4^5 = 1024(mod 7) = 2

4^6 = 4096(mod 7) = 1
```

Input Format

Input the Integer I Input the positive integer P

Constraints

Display the multiplicative order of a module P is the smallest positive integer S with I^S(mod P)=1 (0 < S < P).

Output Format

I and P should be Positive Integers. Negative numbers and decimals are not allowed.

Sample Input 0

```
7
11
```

Sample Output 0

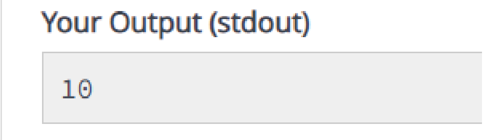
```
10
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    static int fdPower(int base, int expo){
        int result = 1;
        while (expo != 0) {
            result *= base;
            --expo;
        }
        return result;
    }
    static int fdLCM(int a, int b){
```

```
    int gr = a<b ? b : a;
    int ls = a>b ? b : a;
    boolean  ok=false;
    int mul=gr;
    int     mulFac=1;
    while(ok!=true){
      if(mul%ls>0){
        mul=gr*mulFac;
        mulFac++;
      }else{
        ok=true;
      }
    }
    return mul;
  }
  public static void main(String args[] ) throws Exception {
    Scanner ss = new Scanner(System.in);
    int a = ss.nextInt();
    int b = ss.nextInt();
    if(a>0 && b>0 && a%1==0 && b%1==0){
      int prod = a*b;
      int lcm = fdLCM(a,b);
      int gcd = prod/lcm;
      int i = 0;
      int ans;
      int uniq=0;
      int[] ansArr = {0};
      int count=0;
      while(i<b-1){
        ans=(fdPower(a,i))%b;
        ansArr = Arrays.copyOf(ansArr, ansArr.length + 1);
        ansArr[ansArr.length-1] = ans;
        if(i==0){
          uniq=1;
          count++;
        }else if(i>0){
          if(uniq==ans){
            break;
          }else{
            count++;
          }
        }
        i++;
      }
      if(count<0 && count%1==0){
        System.out.println("-1");
      }else{
        System.out.println(count);
      }
    }else{
      System.out.println("-1");
    }
  }
}
```

Output:



Your Output (stdout)

```
10
```

Result: Thus the above program is compiled and executed successfully.

6. Tanmoy has seriously preparing for competitive coding due to some campus placements in his college. For a day he will solve one problem, the today's problem is he will take a random number(R), he wants to find the sum of all GCD that can be formed by picking all the pairs from 1 to that Random Number(R)

Ex:

Input : 4

Output : 7

Explanation:

Numbers from 1 to 4 are: 1, 2, 3, 4

Result = gcd(1,2) + gcd(1,3) + gcd(1,4) + gcd(2,3) + gcd(2,4) + gcd(3,4)

```
= 1 + 1 + 1 + 1 + 2 + 1


= 7
```

Input Format

Input the random number(R)

Constraints

0 < random number(R) <= 10000

(negative numbers and decimals are not allowed)

Output Format

Display the sum of all GCD that can be formed by picking all the pairs from 1 to that Random Number(R).

Sample Input 0

```
8
```

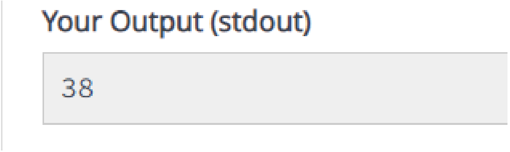
Sample Output 0

```
38
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution  { static

    int gcd(int a, int b){

        if (a == 0)

        return   b;

        if (b == 0)

        return   a;

        if (a == b)

            return a;
        if (a > b)
            return gcd(a-b, b);
        return gcd(a, b-a);
    }
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int sum = 0;
        for(int i=1; i<n; i++){
            for(int j=i+1; j<=n; j++){
              sum += gcd(i,j);
            }
        }
        System.out.println(sum);
    }
}
```

Output:



Your Output (stdout)

```
38
```

Result:

Thus the above program is compiled and executed successfully.

Ex No: E3b DATE: 02-04-2022

1. Professor Rakesh is very much interested in Mathematical algorithms. Professor Rakesh Given you two positive integers 'a' and 'b'. You are required to calculate the sum of the numbers divisible by 3 and 5, between 'a' and 'b' both inclusive and return same.

Input Format

The first and only line of input consists of two integers a and b

Constraints

1 < a,b < 1000

Output Format

Output the sum of numbers that are divisible by 3 and 5

Sample Input 0

```
12 50
```

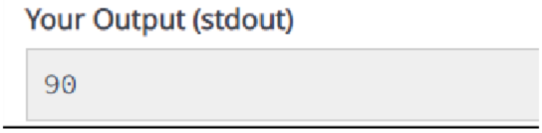
Sample Output 0

```
90
```

Program Code:

```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        int sum = 0;
        for(int i=a; i<=b; i++){
            if(i%3 == 0 && i%5 == 0){
                sum += i;
            }
        }
        System.out.println(sum);
    }
}
```

Output:

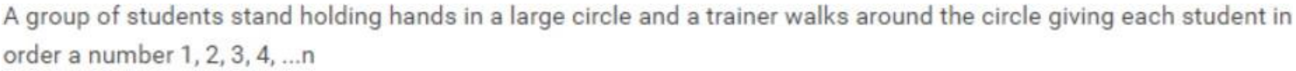


Your Output (stdout)

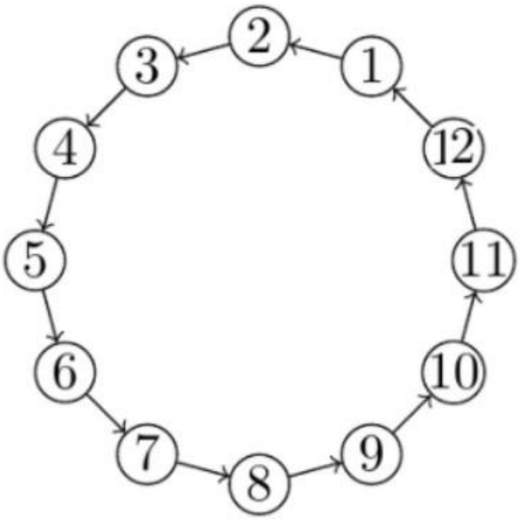
```
90
```

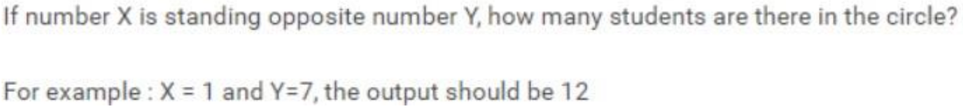
Result: Thus the above program is compiled and executed successfully.

2.



A group of students stand holding hands in a large circle and a trainer walks around the circle giving each student in order a number 1, 2, 3, 4, ...n





If number X is standing opposite number Y, how many students are there in the circle?

For example : X = 1 and Y=7, the output should be 12

Input Format

X and Y value

Constraints

Nil

Output Format

Total number of students

Sample Input 0

```
1 8
```

Sample Output 0

```
14
```

Sample Input 1

```
2 4
```

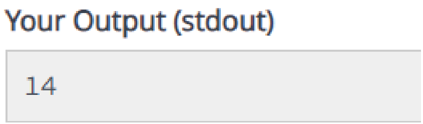
Sample Output 1

```
4
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.math.*;

public class Solution {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a = sc.nextInt();
        int b = sc.nextInt();
        System.out.println(2*(Math.max(a,b)-Math.min(a,b)));
    }
}
```

Output:

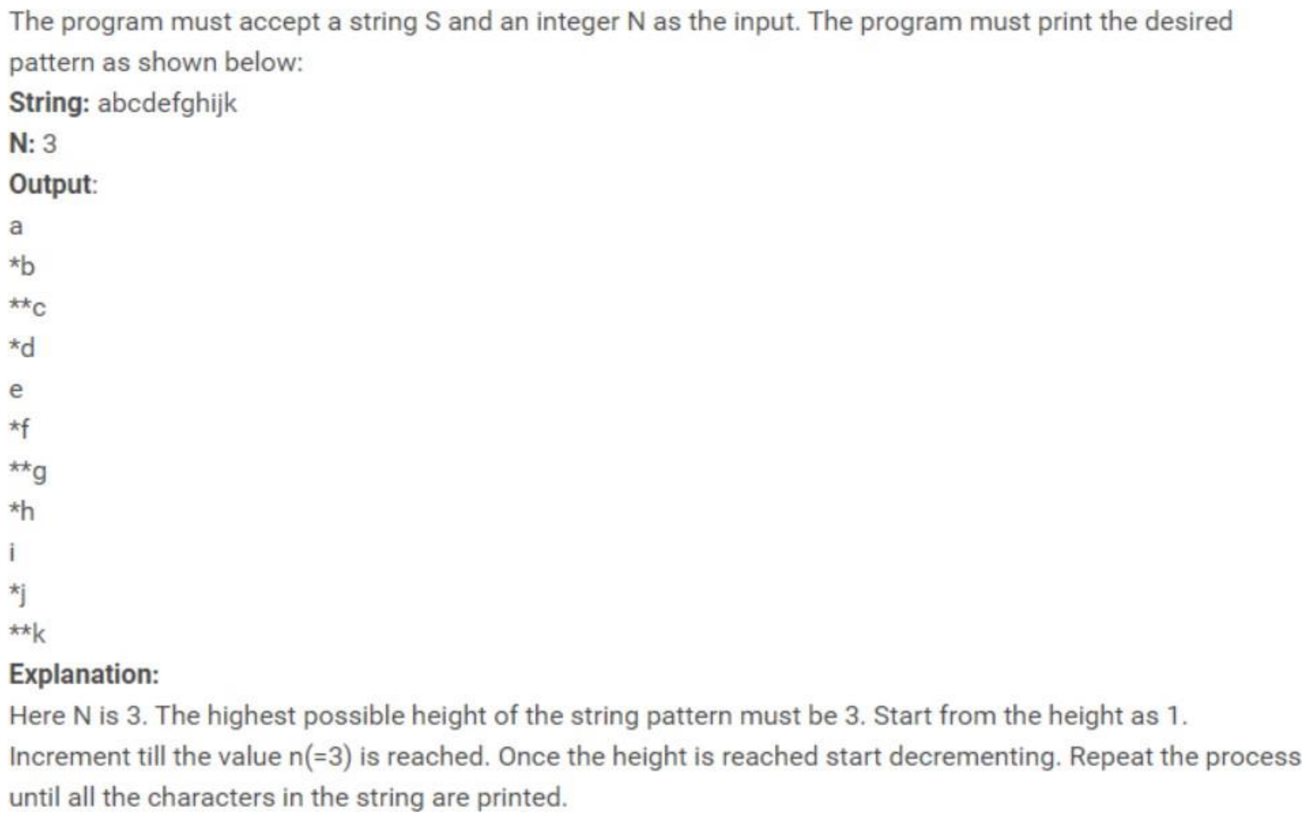


Your Output (stdout)

```
14
```

Result: Thus the above program is compiled and executed successfully.

3.



The program must accept a string S and an integer N as the input. The program must print the desired pattern as shown below:

String: abcdefghijk

N: 3

Output:

```
a
*b
**c
*d
e
*f
**g
*h
i
*j
**k
```

Explanation:

Here N is 3. The highest possible height of the string pattern must be 3. Start from the height as 1. Increment till the value n(=3) is reached. Once the height is reached start decrementing. Repeat the process until all the characters in the string are printed.

Input Format

Two lines of input where first line has the string and second line has the value of N

Constraints

Nil

Output Format

Output is displayed as shown is sample test case.

Sample Input 0

```
happiness
5
```

Sample Output 0

```
h
*a
**p
***p
****i
***n
**e
*s
s
```

Sample Input 1

```
pRogramminG
2
```

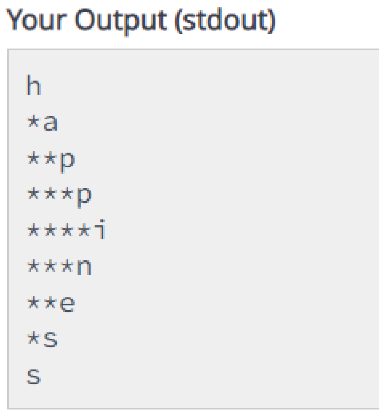
Sample Output 1

```
p
*R
o
*g
r
*a
m
*m
i
*n
G
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s = sc.next();
        int x = sc.nextInt();
        int n = s.length();
         int num = 0, temp = 0;
          for(int i=0; i<n; i++){
            char c = s.charAt(i);
            if(num == x-1){
                temp = 1;
            }
            else if(num == 0){
                temp = 0;
            }
            if(temp == 0){
                for(int j=0; j<num; j++){
                    System.out.print("*");
                }
                num++;
            }
            else{
                for(int j=0; j<num; j++){
                    System.out.print("*");
                }
                num--;
            }
            System.out.print(c+"\n");
        }
    }
}
```

Output:



Your Output (stdout)

```
h
*a
**p
***p
****i
***n
**e
*s
s
```

Result: Thus the above program is compiled and executed successfully.

4. Kate is given a task to arrange the students marksheets based upon their total score. However the teacher has asked to do it in such a way that odd numbered roll numbers have to be arranged in descending order and even numbered roll numbers in ascending order.

Input Format

First line has the number of students.

Second line displays the marks roll number wise separated by space.

Constraints

Nil

Output Format

Output displays the sorted list.

Sample Input 0

```
5
45 12 46 95 78
```

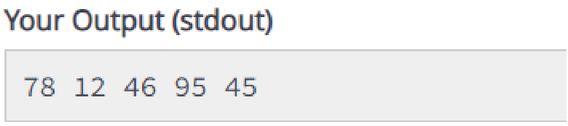
Sample Output 0

```
78 12 46 95 45
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        Integer[] o;
        Integer[] e;
        if(n%2 == 0){
            o = new Integer[(n/2)];
            e = new Integer[(n/2)];
        }
        else{
            o = new Integer[(n/2)+1];
            e = new Integer[(n/2)];
        }
        int temp = 0, oi = 0, ei = 0;
        for(int i=0; i<n; i++){
            if(temp == 0){
                o[oi] = sc.nextInt();
                oi++;
                temp = 1;
            }
            else{
                e[ei] = sc.nextInt();
                ei++;
                temp = 0;
            }
        }
        oi = 0;
        ei = 0;
        temp = 0;
        Arrays.sort(o, Collections.reverseOrder());
        Arrays.sort(e);
        for(int i=0; i<n ; i++){
            // System.out.println(o[i]);
            if(temp == 0){
                System.out.print(o[oi] + " ");
                oi++;
                temp = 1;
            }
            else{
                System.out.print(e[ei] + " ");
                ei++;
                temp = 0;
            }
        }
    }
}
```

Output:



Your Output (stdout)

```
78 12 46 95 45
```

Result: Thus the above program is compiled and executed successfully.

5. A team is participating in a competition in which one member has to say a string other one has to say a substring from it and the third one has to say the string that has to replace the substring and the last person has to say the modified string. Write a program to obtain a string, substring, and the string that has to replace the substring and prints the modified string.

Input Format

The first line of the input consists of a string.

The second line of the input consists of a substring.

The third line of the input consists of a string that has to replace the substring.

Constraints

Nil

Output Format

Output prints the modified string.

Sample Input 0

```
Miss the boat
the
a
```

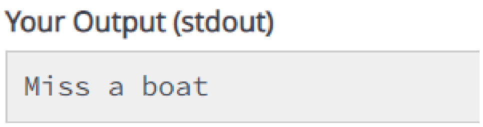
Sample Output 0

```
Miss a boat
```

Program Code:

```
import java.io.*;
import java.util.*;

public class Solution {
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        String str=sc.nextLine();
        String con=sc.next();
        String rep=sc.next();
        String newstr = str.replace(con,rep);
        System.out.println(newstr);
    }
}
```

Output:



Your Output (stdout)

```
Miss a boat
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 4 DATATYPES

Ex No: E4a DATE: 25-04-2022

1. Java has 8 primitive data types; *char, boolean, byte, short, int, long, float, and double*. For this exercise, we'll work with the primitives used to hold integer values (*byte, short, int,* and *long*):

- A *byte* is an 8-bit signed integer.
- A *short* is a 16-bit signed integer.
- An *int* is a 32-bit signed integer.
- A *long* is a 64-bit signed integer.

Given an input integer, you must determine which primitive data types are capable of properly storing that input.

To get you started, a portion of the solution is provided for you in the editor.

Reference: https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

Input Format

The first line contains an integer, , denoting the number of test cases. Each test case, , is comprised of a single line with an integer, , which can be arbitrarily large or small.

Output Format

For each input variable and appropriate primitive , you must determine if the given primitives are capable of storing it. If yes, then print:

```
n can be fitted in:
* dataType
```

If there is more than one appropriate data type, print each one on its own line and order them by size (i.e.:).

If the number cannot be stored in one of the four aforementioned primitives, print the line:

```
n can't be fitted anywhere.
```

Sample Input

```
5
-150
150000
1500000000
213333333333333333333333333333333333
-100000000000000
```

Sample Output

```
-150 can be fitted in:
* short
* int
* long
150000 can be fitted in:
* int
* long
1500000000 can be fitted in:
* int
* long
213333333333333333333333333333333333 can't be fitted anywhere.
-100000000000000 can be fitted in:
* long
```

Explanation

-150 can be stored in a *short*, an *int*, or a *long*.

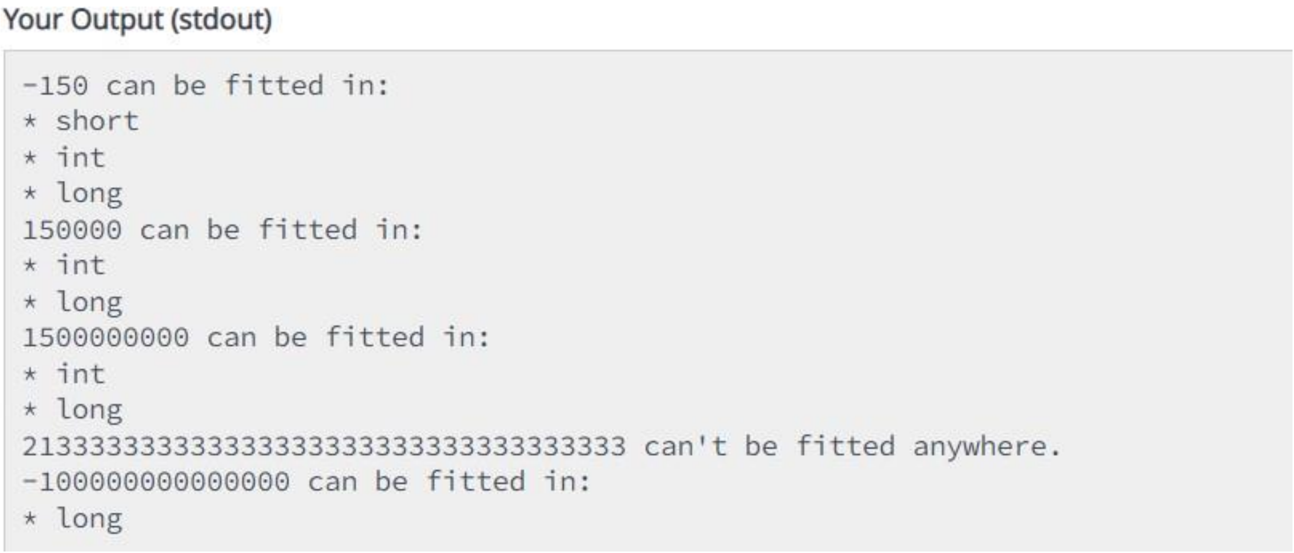
$213333333333333333333333333333333333$ is very large and is outside of the allowable range of values for the primitive data types discussed in this problem.

Program Code:

```
import java.util.*;
import java.io.*;
class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int t = sc.nextInt();
        for (int i = 0; i < t; i++) {
            try {
                long x = sc.nextLong();
                System.out.println(x + " can be fitted in:");
                if (x >= Byte.MIN_VALUE && x <= Byte.MAX_VALUE) {
                    System.out.println("* byte");
                }
                if (x >= Short.MIN_VALUE && x <= Short.MAX_VALUE) {
```

```
                System.out.println("* short");
            }
            if (x >= Integer.MIN_VALUE && x <= Integer.MAX_VALUE) {
                System.out.println("* int");
            }
            if (x >= Long.MIN_VALUE && x <= Long.MAX_VALUE) {
                System.out.println("* long");
            }
        } catch (Exception e) {
            System.out.println(sc.next() + " can't be fitted anywhere.");
        }
    }
    sc.close();
  }
}
```

Output:



Your Output (stdout)

```
-150 can be fitted in:
* short
* int
* long
150000 can be fitted in:
* int
* long
1500000000 can be fitted in:
* int
* long
213333333333333333333333333333333333 can't be fitted anywhere.
-100000000000000 can be fitted in:
* long
```

Result: Thus the above program is compiled and executed successfully.

2. The challenge here is to read lines of input until you reach *EOF*, then number and print all lines of content.

Hint: Java's *Scanner.hasNext()* method is helpful for this problem.

Input Format

Read some unknown lines of input from *stdin(System.in)* until you reach *EOF*; each line of input contains a non-empty *String*.

Output Format

For each line, print the line number, followed by a single space, and then the line content received as input.

Sample Input

```
Hello world
I am a file
Read me until end-of-file.
```

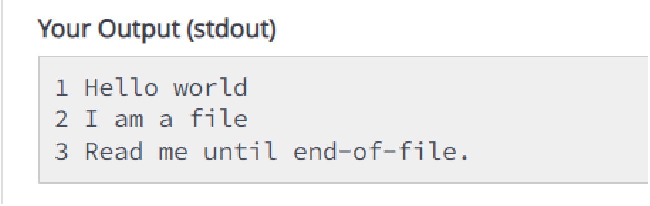
Sample Output

```
1 Hello world
2 I am a file
3 Read me until end-of-file.
```

Program Code:

```
import java.io.*;
import java.util.*;


public class EndOfFile {
   public static void main(String []args) {
      Scanner sc = new Scanner(System.in);
      int i=1;
      while (sc.hasNext()) {
         String s=sc.nextLine();
         System.out.println(i + " " + s);
         i++;
      }
   }
}
```

Output:



```
1 Hello world
2 I am a file
3 Read me until end-of-file.
```

Result: Thus the above program is compiled and executed successfully.

3. Static initialization blocks are executed when the class is loaded, and you can initialize static variables in those blocks.

It's time to test your knowledge of *Static initialization blocks*. You can read about it here.

You are given a class *Solution* with a *main* method. Complete the given code so that it outputs the area of a parallelogram with breadth and height . You should read the variables from the standard input.

If or , the output should be *"java.lang.Exception: Breadth and height must be positive"* without quotes.

Input Format

There are two lines of input. The first line contains B: the breadth of the parallelogram. The next line contains H : the height of the parallelogram.

Constraints

- $-100 \leq B \leq 100$
- $-100 \leq H \leq 100$

Output Format

If both values are greater than zero, then the *main* method must output the area of the *parallelogram*. Otherwise, print *"java.lang.Exception: Breadth and height must be positive"* without quotes.

Sample input 1

```
1
3
```

Sample output 1

```
3
```

Sample input 2

```
-1
2
```

Sample output 2

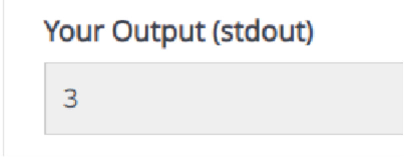
```
java.lang.Exception: Breadth and height must be positive
```

Program Code:

```
static boolean flag;
static int B,H;
```

```
static{
   Scanner io = new Scanner(System.in);
   B = io.nextInt();
   H = io.nextInt();
   if(B>0 && H>0)
   {
      flag = true;
   }
   else
   {
      System.out.println("java.lang.Exception: Breadth and height must be positive");
   }
}
```

Output:



Your Output (stdout)

```
3
```

Result: Thus the above program is compiled and executed successfully.

4. You are given an integer , you have to convert it into a string.

Please complete the partially completed code in the editor. If your code successfully converts into a string the code will print "*Good job*". Otherwise it will print "*Wrong answer*".

can range between to inclusive.

Sample Input 0

```
100
```

Sample Output 0

```
Good job
```

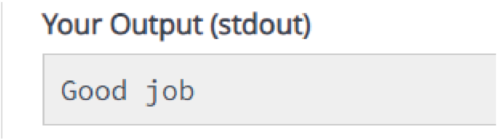
Program Code:

```
import java.util.*;
import java.security.*;
public class Solution {
 public static void main(String[] args) {
```

```
DoNotTerminate.forbidExit();


try {
 Scanner in = new Scanner(System.in);
 int n = in .nextInt();
 in.close();
 //String s=???; Complete this line below
 String s=String.valueOf(n);
 if (n == Integer.parseInt(s)) {
  System.out.println("Good job");
 } else {
  System.out.println("Wrong answer.");
 }
} catch (DoNotTerminate.ExitTrappedException e) {
 System.out.println("Unsuccessful Termination!!");
}
}
}
class DoNotTerminate {
public static class ExitTrappedException extends SecurityException {
 private static final long serialVersionUID = 1;
}
public static void forbidExit() {
 final SecurityManager securityManager = new SecurityManager() {
  @Override
  public void checkPermission(Permission permission) {
   if (permission.getName().contains("exitVM")) {
   throw new ExitTrappedException();
   }
  }
 };
 System.setSecurityManager(securityManager);
}
```

}

Output:



Your Output (stdout)

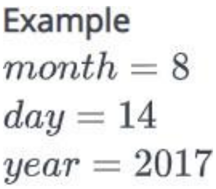
```
Good job
```

Result: Thus the above program is compiled and executed successfully.

5. The Calendar class is an abstract class that provides methods for converting between a specific instant in time and a set of calendar fields such as YEAR, MONTH, DAY_OF_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week.

You are given a date. You just need to write the method, , which returns the *day* on that date. To simplify your task, we have provided a portion of the code in the editor.

Example



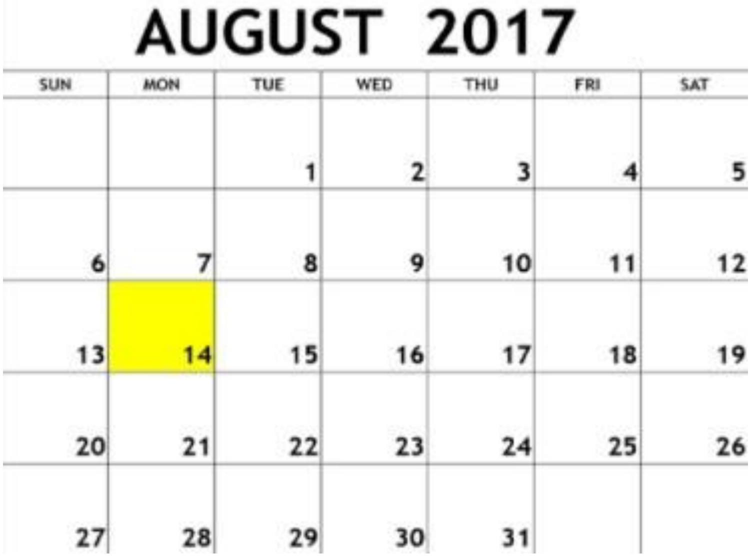
Example

$month = 8$

$day = 14$

$year = 2017$

The method should return $MONDAY$ as the day on that date.



AUGUST 2017

SUN	MON	TUE	WED	THU	FRI	SAT
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Function Description

Complete the *findDay* function in the editor below.

findDay has the following parameters:

- *int:* month
- *int:* day
- *int:* year

Returns

- *string:* the day of the week in capital letters

Input Format

A single line of input containing the space separated month, day and year, respectively, in $MM\ DD\ YYYY$ format.

Constraints

- $2000 < year < 3000$

Sample Input

```
08 05 2015
```

Sample Output

```
WEDNESDAY
```

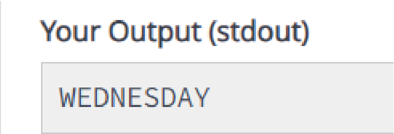
Explanation

The day on August 5^{th} 2015 was WEDNESDAY.

Program Code:

```
class Result {
    public static String findDay(int month, int day, int year) {
  Calendar cal = Calendar.getInstance();


        cal.set(Calendar.MONTH, month-1);


        cal.set(Calendar.DAY_OF_MONTH, day);


        cal.set(Calendar.YEAR, year);


        String[] day_of_week = {"SUNDAY", "MONDAY", "TUESDAY", "WEDNESDAY", "THURSDAY", "FRIDAY","SATURDAY"};


        return day_of_week[cal.get(Calendar.DAY_OF_WEEK)-1];
    }


}
```

Output:



Your Output (stdout)

```
WEDNESDAY
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 5 **STREAM CLASS**

Ex No: E5a DATE: 02-05-2022

Write a class that reads characters from the keyboard using the method read(char[]

b); the maximum number of characters that can be read at a time is fixed and determined by the size of the buffer. Use the attached program snippet attached in the BB/initial version. Here is a list of modifications that you have to do:

1. You no longer need the variable "i" to stock the values that were read. You will use a buffer of type char[] instead, as shown previously.

2. You will need to change the condition statement for the while loop. Notably, the variable "i" is not part of the solution anymore, you will need to use the method read(char[] b) instead.

3. Once the data is read by the method read, it is stored in the buffer. You will need to convert this buffer to a String.

4. Note that some symbols are not displayable. You will need to use the method trim to remove these characters. For example, yourString.trim() returns a new object of type String without the characters that are not displayable.

5. Write the characters that you got to the console.

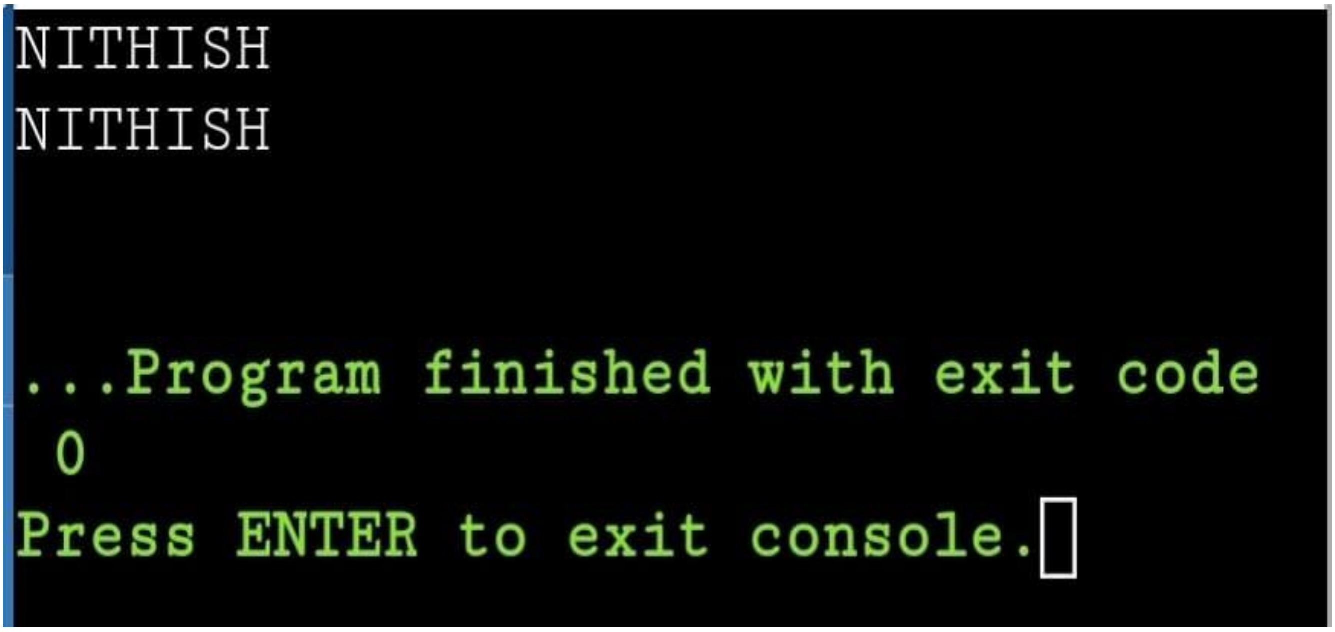
6. Finally, empty your buffer using the following command: Arrays.fill(buffer, '\u0000'); No matter how many characters you’ve entered the resulting String is always 256 characters long.

Program Code:

```
import java.io.InputStreamReader;

import java.io.IOException;

import java.util.Arrays;


public class Main {

  public static void main( String[] args ) throws IOException {

    InputStreamReader in = new InputStreamReader( System.in );

    char[] buffer = new char[300];

    int num = in.read(buffer);

    String str = new String (buffer);

    str = str.trim();

    System.out.println(str);

    Arrays.fill( buffer, '\u0000' );

  }
```

}

Output:



```
NITHISH
NITHISH


...Program finished with exit code
 0
Press ENTER to exit console.
```

Result: Thus the above program is compiled and executed successfully.

2. Write a class that prints all the lines that contain a certain word. For each line containing the word, print the line number followed by the line itself.

- You will need to use the method readLine() from the class BufferedReader. This method returns a String containing the content of the line or null if you have reached the end of the file.

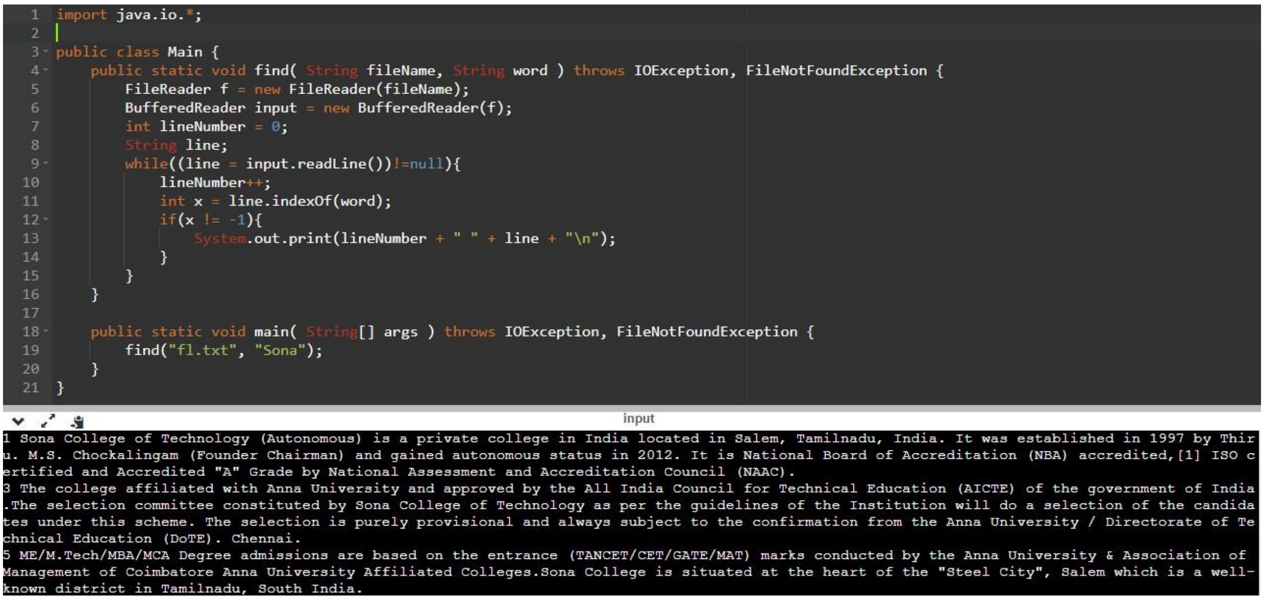
- You will then need to check if the word is part of the String. To do so, use the method indexOf(yourWord) that returns the position of your word in the String or -1 if it is not part of the String. The code that you will need to modify for this exercise is attached in BB JavaIOFind.java

Program Code:

```
import java.io.*;
public class Main {
  public static void find( String fileName, String word ) throws IOException,
        FileNotFoundException {
     FileReader f = new FileReader(fileName);
BufferedReader input = new BufferedReader(f);
     int lineNumber = 0;
```

```
            int x = line.indexOf(word);
            if(x != -1){
                System.out.print(lineNumber + " " + line + "\n");
            }
        lineNumber++;
        }


    public static void main( String[] args ) throws IOException, FileNotFoundException {
        find("fl.txt", "Sona");
    }
}
```

Output:



```
import java.io.*;

public class Main {
    public static void find( String fileName, String word ) throws IOException, FileNotFoundException {
        FileReader f = new FileReader(fileName);
        BufferedReader input = new BufferedReader(f);
        int lineNumber = 0;
        String line;
        while((line = input.readLine())!=null){
            lineNumber++;
            int x = line.indexOf(word);
            if(x != -1){
                System.out.print(lineNumber + " " + line + "\n");
            }
        }
    }

    public static void main( String[] args ) throws IOException, FileNotFoundException {
        find("fl.txt", "Sona");
    }
}
```

```
1 Sona College of Technology (Autonomous) is a private college in India located in Salem, Tamilnadu, India. It was established in 1997 by Thir
u. M.S. Chockalingam (Founder Chairman) and gained autonomous status in 2012. It is National Board of Accreditation (NBA) accredited,[1] ISO c
ertified and Accredited "A" Grade by National Assessment and Accreditation Council (NAAC).
3 The college affiliated with Anna University and approved by the All India Council for Technical Education (AICTE) of the government of India
.The selection committee constituted by Sona College of Technology as per the guidelines of the Institution will do a selection of the candida
tes under this scheme. The selection is purely provisional and always subject to the confirmation from the Anna University / Directorate of Te
chnical Education (DoTE). Chennai.
5 ME/M.Tech/MBA/MCA Degree admissions are based on the entrance (TANCET/CET/GATE/MAT) marks conducted by the Anna University & Association of
Management of Coimbatore Anna University Affiliated Colleges.Sona College is situated at the heart of the "Steel City", Salem which is a well-
known district in Tamilnadu, South India.
```

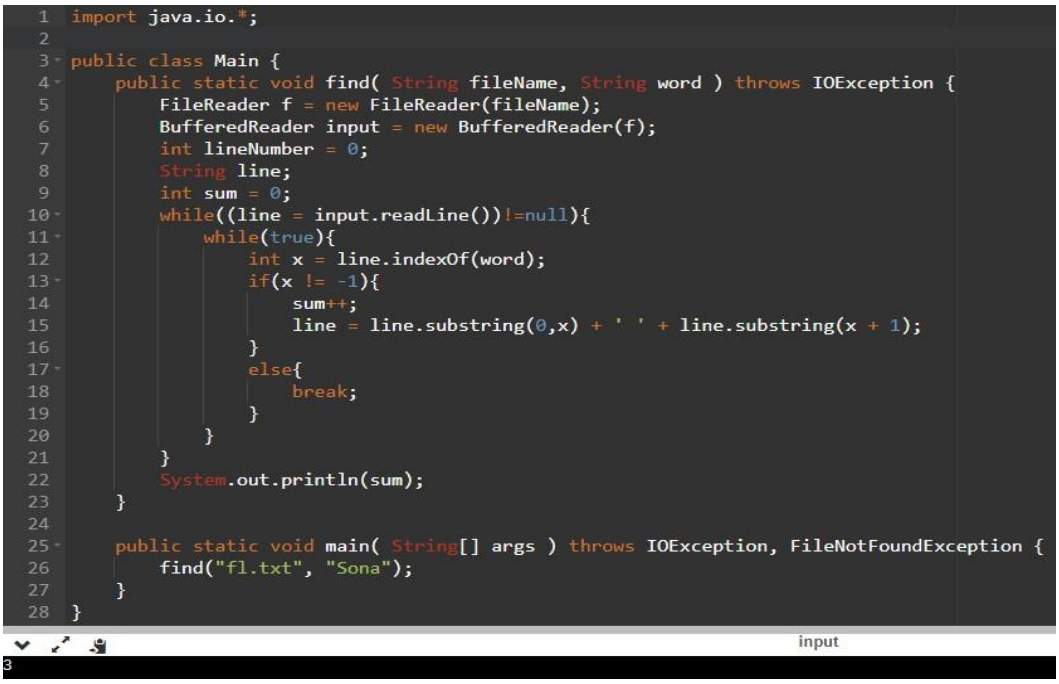
Result: Thus the above program is compiled and executed successfully.

3. Write a class that counts the number of occurrences of the word Sona in the following file: Sona.txt

Program Code:

```
import java.io.*;


public class Main {
    public static void find( String fileName, String word ) throws IOException {
        FileReader f = new FileReader(fileName);
BufferedReader input = new BufferedReader(f);
        int lineNumber = 0;
        String line;
        int sum = 0;
        while((line = input.readLine())!=null){
            while(true){
                int x = line.indexOf(word);
                if(x != -1){
                    sum++;
                    line = line.substring(0,x) + ' ' + line.substring(x + 1);
                }
                else{
                    break;
                }
            }
        }
        System.out.println(sum);
    }

    public static void main( String[] args ) throws IOException, FileNotFoundException {
        find("fl.txt", "Sona");
    }
}
```

Output:



```
import java.io.*;

public class Main {
    public static void find( String fileName, String word ) throws IOException {
        FileReader f = new FileReader(fileName);
        BufferedReader input = new BufferedReader(f);
        int lineNumber = 0;
        String line;
        int sum = 0;
        while((line = input.readLine())!=null){
            while(true){
                int x = line.indexOf(word);
                if(x != -1){
                    sum++;
                    line = line.substring(0,x) + ' ' + line.substring(x + 1);
                }
                else{
                    break;
                }
            }
        }
        System.out.println(sum);
    }

    public static void main( String[] args ) throws IOException, FileNotFoundException {
        find("fl.txt", "Sona");
    }
}
```

```
3
```

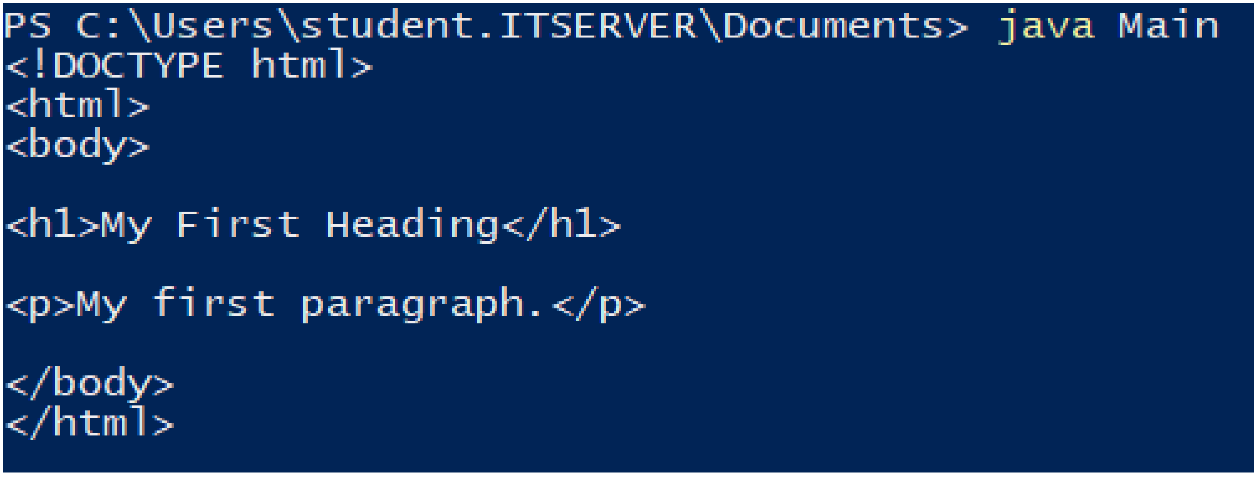
Result: Thus the above program is compiled and executed successfully.

4. Write a class that fetches the content of a Web page and prints it on the console.

Program Code:

```
import java.io.*;
public class Main {
   public static void find( String fileName) throws IOException {
      FileReader f = new FileReader(fileName);
      String val;
      BufferedReader br = new BufferedReader(f);
      while ((val = br.readLine()) != null) {
         System.out.println(val);
      }
      br.close();
   }
   public static void main( String[] args ) throws IOException, FileNotFoundException {
      find("D:\\as.html");
   }
}
```

Output:



```
PS C:\Users\student.ITSERVER\Documents> java Main
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 6 **STRING**

Ex No: E6a DATE: 09-05-2022

1. Generate a password from the given N set of strings and digits separated by colon(:) and each set is separated by, (comma) with following conditions: Find the largest digit X smaller than or equal length of the string and choose the character in the string at position X assuming that index starting from 1.

If there is no such digit exists place X (capital x)instead.

INPUT: Given N set of strings and digits separated by colon (:) and each set is separated by, (comma).

OUTPUT: Password string of length N.

Input Format

RAVIKUMAR:879567,ASHIF:678,YUKESH:54890

Constraints

NIL

Output Format

RXS

Explanation: 9 is the largest digit lesser than or equal to the length of the digit R is the character present at the ninth position

In second string there no digit lesser than the length of the string Hence X is placed

And in third string 5 is the digit hence S is placed

Sample Input 0

```
RAVIKUMAR:879567,ASHIF:678,YUKESH:54890
```

Sample Output 0

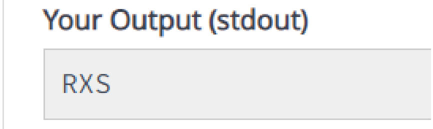
```
RXS
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;
```

```
public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc=new Scanner(System.in);
        String s=sc.nextLine();
        String[] s1=s.split(",");
        String s3="";
        for(int  i=0;i<s1.length;i++){
            String s2[]=s1[i].split(":");
            int min=1000,k=s2[0].length(),m=0;
            for(int j=0;j<s2[1].length();j++){
                String                                    s4=s2[1];
                if(Character.getNumericValue(s4.charAt(j))<=k){
                    if(min>=k-Character.getNumericValue(s4.charAt(j))){
                        min=k-Character.getNumericValue(s4.charAt(j));
                        m=Character.getNumericValue(s4.charAt(j));
                    }
                }

            }
            String s5=s2[0];
            if(m>s2[0].length()||min==1000){
                s3+="X";
            }
            else{
            s3+=s5.charAt(m-1);
            }
        }
        System.out.println(s3);
    }
}
```

Output:



Your Output (stdout)

```
RXS
```

Result: Thus the above program is compiled and executed successfully.

2. Given a string S and a number K,print the string K times.

Input Format

string followed by(space) number of times to be printed

Constraints

1<=K<=1000

Output Format

strings

Sample Input 0

```
hello 3
```

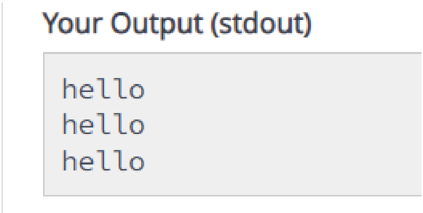
Sample Output 0

```
hello
hello
hello
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution {

    public static void main(String args[] ) throws Exception {

        Scanner sc = new Scanner(System.in);

        String  s  =  sc.next();

        int  n  =  sc.nextInt();

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

            System.out.println(s);

        }

    }
}
```

Output:



Your Output (stdout)

```
hello
hello
hello
```

Result: Thus the above program is compiled and executed successfully.

3. Given a sentence, count the number of characters in the given sentence(without counting the space).

Input Format

Sentence

Constraints

1<=character<=1000

Output Format

Character count

Sample Input 0

```
hello how are you
```

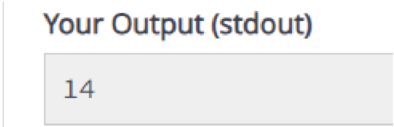
Sample Output 0

```
14
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        int sum = 0;
        for(int i=0; i<s.length(); i++){
            if(s.charAt(i) != ' ')
                sum++;
        }
        System.out.println(sum);
    }
}
```

Output:



Your Output (stdout)

```
14
```

Result: Thus the above program is compiled and executed successfully.

4. Given a number N and array of N integers, print the minimum element.

Input Format

Number N Array of elements seperated by space

Constraints

1<=N<=1000

Output Format

Minimum element

Sample Input 0

```
3
1 2 3
```

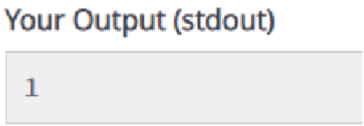
Sample Output 0

```
1
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int min = 800000000;
        for(int f=0; f<n; f++){
            int x = sc.nextInt();
            if(x<min)
                min = x;
        }
        System.out.println(min);
    }
}
```

Output:



Your Output (stdout)

```
1
```

Result: Thus the above program is compiled and executed successfully.

Ex No: E6b DATE: 20-05-2022

1. Susan is a good programmer. He always thinks logic for some silly questions. Now she wants to write a program to get the string as an input, remove the consonants and print the string which doesn't contain consonants

Input Format

A single line string as an input

Constraints

no constraints

Output Format

string contains only vowels if no vowels in the string then print "No".

Sample Input 0

```
hai hello
```

Sample Output 0

```
ai eo
```

Explanation 0

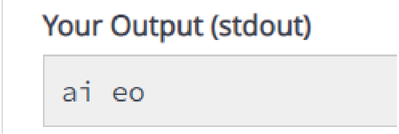
Since h is consonants in "hai" remove j and print remaining h and l are consonants in "hello" remove h and l and print remaining

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        String a = "";
        int temp = 1;
        for(int i=0; i<s.length(); i++){
            if(s.charAt(i) == 'a' || s.charAt(i) == 'e' || s.charAt(i) == 'i' || s.charAt(i) == 'o' || s.charAt(i) ==
'u' || s.charAt(i) == 'A' || s.charAt(i) == 'E' || s.charAt(i) == 'I' || s.charAt(i) == 'O' || s.charAt(i) == 'U'){
```

```
                a = a + s.charAt(i);
                temp = 0;
            }
            else if(s.charAt(i) == ' ' && a.charAt(a.length()-1) != ' '){
                a = a + ' ';
            }
        }
        if(temp == 0)
            System.out.println(a);
        else
            System.out.println("No");
    }
}
```

<u>Output:</u>



```
ai eo
```

<u>Result:</u> Thus the above program is compiled and executed successfully.

2. Write a function which takes as input a string that consists only of the digits from 0 to 9 and outputs the number showing how many times each digit occured in the string.

Input Format

numbers from 0 to 9 only

Constraints

no constraints

Output Format

number occurrences

Sample Input 0

```
01223334444566
```

Sample Output 0

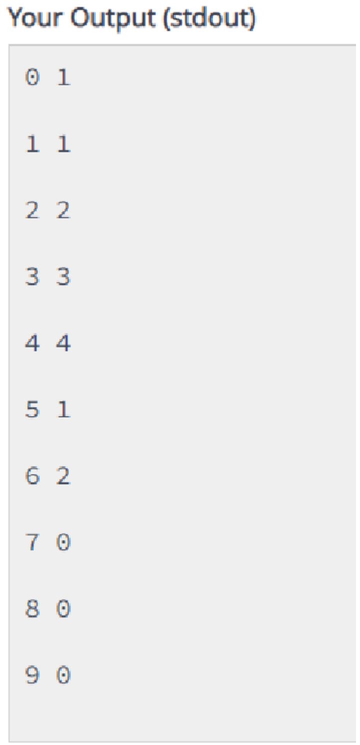
```
0 1


1 1
```

```
2 2

3 3

4 4

5 1

6 2

7 0

8 0

9 0
```

Program Code:

```
import java.io.*;
import java.util.*;

public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        int[] a = new int[11];
        for(int i=0; i<s.length(); i++){
            int x = (int)s.charAt(i) - 48;
            a[x]++;
        }
        for(int i=0; i<10; i++){
            System.out.print(i + " " + a[i] + "\n\n");
        }
    }
}
```

Output:



Your Output (stdout)

```
0 1

1 1

2 2

3 3

4 4

5 1

6 2

7 0

8 0

9 0
```

Result: Thus the above program is compiled and executed successfully.

3. Arrange and Print consecutive characters together in a line.

Input Format

ABCXYZACCD

Constraints

Allows Only Uppercase Letters starting from A to Z

Output Format

ABC XYZ A C CD

Sample Input 0

```
ABCXYZACCD
```

Sample Output 0

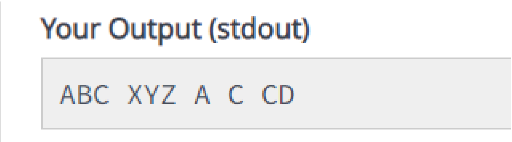
```
ABC XYZ A C CD
```

Program Code:

```
import java.io.*;
import java.util.*;
```

```
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        for(int i=0; i<s.length(); i++){
            int x = (int)s.charAt(i) - 65;
            System.out.print(s.charAt(i));
            i++;
            while(true){
                if(i<s.length() && x+1 == (int)s.charAt(i)-65){
                    x++;
                    System.out.print(s.charAt(i));
                    i++;
                }
                else{
                    System.out.print(' ');
                    i--;
                    break;
                }
            }
            // System.out.print(i);
        }
    }
}
```

Output:



Your Output (stdout)

```
ABC XYZ A C CD
```

Result: Thus the above program is compiled and executed successfully.

4. A string is said to be palindrome, if it reads the same from both the ends. Given a string S, you are allowed to perform cyclic shifts. More formally, you can pick any one character from any end (head or tail) and you can append that character at the other end.

For example, if the string is "abc", then if we do a shift using the character at head position then the string becomes "bca". Similarly, if we do the shift using the character at the tail then the input string becomes "cab". Your task is to find out the minimum number of shifts needed to make the given string, a palindrome. In case, we can't convert the string to palindrome then print -1.

Input Format

Containing a string "S"

Constraints

1<=|S|<=300, S will contains only lower case alphabets ('a'-'z').

Output Format

Print the minimum number of cyclic shifts for the string if it can be made a palindrome, else -1.

Sample Input 0

```
aabb
```

Sample Output 0

```
1
```

Sample Input 1

```
aabbccbbaa
```

Sample Output 1

```
0
```

Sample Input 2

```
oneweraarewenot
```

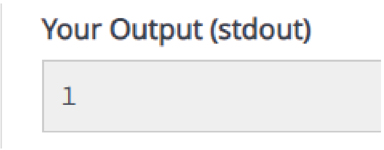
Sample Output 2

```
7
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    static boolean isPalindrome(String str) {
        int i = 0, j = str.length() - 1;
        while (i < j) {
            if (str.charAt(i) != str.charAt(j))
                return false;
            i++;
            j--;
        }
        return true;
    }
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        String s = sc.nextLine();
        int ans = 0;
        int a[] = new int[27];
        for(int i=0; i<s.length(); i++){
            a[s.charAt(i)-97]++;
        }
        int od = 0, temp = 1;
        for(int i=0; i<26; i++){
            if(a[i]%2 != 0 && od == 0){
                od = 1;
            }
            else if(a[i]%2 != 0 && od == 1){
                temp = 0;
            }
        }
        // System.out.println(od);
        if(temp == 1){
            for(int i=0; i<s.length(); i++){
                if(isPalindrome(s)){
                    System.out.print(ans);
                    break;
                }
                else{
                    s = s.substring(1, s.length()) + s.substring(0,1);
                    // System.out.println(s);
                    ans++;
                }
            }
        }
        else{
            System.out.print(-1);
        }
    }
}
```

Output:



Your Output (stdout)

```
1
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 7 **COLLECTION INTERFACE- I**

Ex No: E7a DATE: 27-05-2022

1. Arrays are of fixed size and are difficult to use compared to collections. So we are gonna move into collections. The basic collection is a list. Now let us try out basic ArrayList.

Create a class Solution and in the main method get the usernames and store them in an ArrayList. After getting all the names, just display them in the same order.

Refer sample input/output for other further details and format of the output.

Input Format

The first line of the input consists of the number of users.

Next input is the user names.

Constraints

Nil

Output Format

The output displays the usernames.

Sample Input 0

```
2
Aldo
Stalin
```

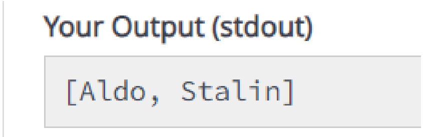
Sample Output 0

```
[Aldo, Stalin]
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {


    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class should be named Solution. */
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        ArrayList al = new ArrayList();
        for(int i=0;i<n;i++)
          al.add(sc.next());
        System.out.println(al);
    }
}
```

Output:



```
[Aldo, Stalin]
```

Result: Thus the above program is compiled and executed successfully.

2. Equals method - ArrayList

Let's dive deep into array list and explore its inbuilt functions. We usually perform equal operation to compare objects. Now try the same feature here. Experiment with equalsl() method in this problem.

Obtain two list of names and check whether they are same or not.

Input Format

The first line of the input consists of the value of n that represents the number of elements in both lists.

Next input is the elements of the first set.

Last input is the elements of the second set.

Constraints

Nil

Output Format

The output prints whether both the lists are same or not. Refer sample output.

Sample Input 0

```
3
aldo
stalin
J
aldo
stalin
```

```
J
```

Sample Output 0

```
Both lists are same
```

Sample Input 1

```
3
aldo
stalin
J
aldo
L
stalin
```

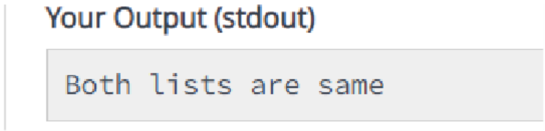
Sample Output 1

```
Both lists are different
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
ArrayList l1 = new ArrayList();
ArrayList l2 = new ArrayList();
for(int i=0;i<n;i++){
l1.add(sc.next());
}
for(int i=0;i<n;i++){
l2.add(sc.next());
}
for(int i=0;i<n;i++){
if(!(l1.get(i).equals(l2.get(i)))){
System.out.println("Both lists are different");
return;
}
}
System.out.println("Both lists are same");
    }
}
```

Output:



Your Output (stdout)

```
Both lists are same
```

Result: Thus the above program is compiled and executed successfully.

3. contains() & indexOf() methods in ArrayList

Write a program to get the hall name details and store in the ArrayList and search the hall and display it's position details.

Get hall names in the Solution class and store it an ArrayList. Hall number is nothing but the position at which the hall is present in the list starting from 0.

Refer sample input/output for other further details and format of the output.

Input Format

The first line of the input is the number of halls.

Next input is the hall names.

The last input is the hall name to be searched.

Constraints

Nil

Output Format

The output prints the index of hall name if present otherwise hall not found. Refer sample output.

Sample Input 0

```
3
APJ
VAL
MEC
VAL
```

Sample Output 0

```
1
```

Sample Input 1

```
3
APJ
VAL
IOT
SPK
```

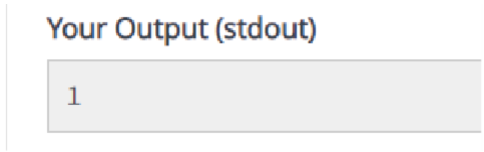
Sample Output 1

```
SPK hall is not found
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        ArrayList al = new ArrayList(n);
        for(int i=0; i<n; i++){
            al.add(sc.next());
        }
        String s = sc.next();
        if(al.contains(s)){
            System.out.println(al.indexOf(s));
        }
        else{
            System.out.println(s+" hall is not found");
        }
    }
}
```

Output:



```
1
```

Result: Thus the above program is compiled and executed successfully.

Ex No: E7b DATE: 02-06-2022

1. Get the inputs from the user and create a linked list. Print the elements.

Input Format

No of element in the list

List elements separated by space

Constraints

Nil

Output Format

Linked List

Refer output format

Sample Input 0

```
3
152 125 698
```

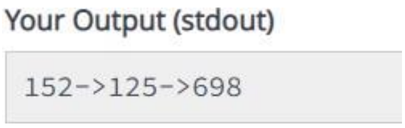
Sample Output 0

```
152->125->698
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        LinkedList ll = new LinkedList();
        for(int i=0; i<n; i++){
            ll.add(sc.nextInt());
        }
        int i;
        for(i=0; i<n-1; i++){
            System.out.print(ll.get(i)+"->");
        }
        System.out.print(ll.get(i));
    }
}
```

Output:



Your Output (stdout)

```
152->125->698
```

Result: Thus the above program is compiled and executed successfully.

2. Create a linked list with floating point numbers and print the list. Also print the first element of the linked list.

Input Format

No of element in the list

List elements separated by space

Constraints

Nil

Output Format

Linked list

Print the first element which is floating point number

Sample Input 0

```
3
10.5 23.2 11.0
```

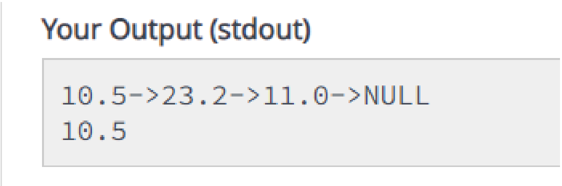
Sample Output 0

```
10.5->23.2->11.0->NULL
10.5
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution {

    public static void main(String args[] ) throws Exception {

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();

        LinkedList ll = new LinkedList();

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

            ll.add(sc.nextDouble());

        }

        int i;

        for(i=0; i<n-1; i++){

            System.out.print(ll.get(i)+"->");

        }

        System.out.print(ll.get(i)+"->NULL\n");

        System.out.println(ll.get(0));

    }

}
```

Output:



Your Output (stdout)

```
10.5->23.2->11.0->NULL
10.5
```

Result: Thus the above program is compiled and executed successfully.

3. A train from Delhi to Bangalore is standing on platform number 2 for half an hour. Passengers to the train receive a message from the train authorities that the coach numbers have been changed and are now allocated some random numbers. Also, from that day onward all trains have such kind of arrangement irrespective of the number of coaches. It is also said that the order to the the coaches will be printed on the board near the engine coach.

Given a specific coach number, every passenger has to enter into the first first coach after the engine and move forward until his/her respective coach is reached. If the train does not have the respective coach number, then the passenger must get out of the last coach.

Input Format

First line has the number of coaches in the train. Second line has the coach numbers in order separated by a space. Third line has the coach number to be found.

Constraints

Nil

Output Format

Output displays the coach number along with a Yes or No answer as shown in sample test cases

Sample Input 0

```
10
1 23 5 12 8 7984 1 654 46 51
21
```

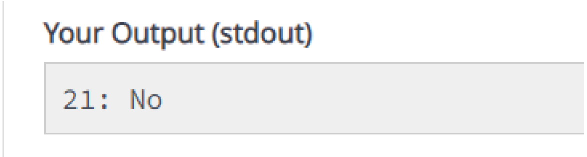
Sample Output 0

```
21: No
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        ArrayList al = new ArrayList();
        for(int i=0; i<n; i++){
            al.add(sc.nextInt());
        }
        int x = sc.nextInt();
        int flag = -1;
        for(int i=0; i<n; i++){
            if((int)al.get(i) == x){
                flag = 1;
                break;
            }
            else{
                flag = 0;
            }
        }
        if(flag == 1){
            System.out.println(x+": Yes");
        }
        else{
            System.out.println(x+": No");
        }
    }
}
```

Output:



Your Output (stdout)

```
21: No
```

Result: Thus the above program is compiled and executed successfully.

4. A linked list is given write a code to insert element at the end.

Input Format

Elemnet to be inserted at the end

No of element in the list

List elements separated by space

Constraints

Nil

Output Format

List with added element

Refer Sample Output

Sample Input 0

```
525
3
152 125 698
```

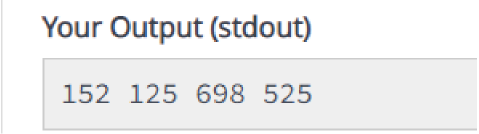
Sample Output 0

```
152 125 698 525
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        LinkedList ll = new LinkedList();
        int x = sc.nextInt();
        int n = sc.nextInt();
        for(int i=0; i<n; i++){
            ll.add(sc.nextInt());
        }
        ll.add(x);
        for(int i=0; i<n+1; i++){
            System.out.print(ll.get(i)+" ");
        }
    }
}
```

Output:



Your Output (stdout)

```
152 125 698 525
```

Result: Thus the above program is compiled and executed successfully.

5. Given a linked list L, rearrange it in a zig-zag form(Mountain). List L should be converted in away a < b > c < d > e < f .. where a, b, c.. are consecutive elements of linked list.

   Note:

The last element from the user should be the first element in the linked list.

Input Format

The first line denotes the data d inside the linked list L

Constraints

1 < d < 100

Output Format

Prints the zig-zag form of the list L.

Sample Input 0

```
7
3 2 4 5 6 7 8
```

Sample Output 0

```
8->7->6->5->4->2->3
7->8->5->6->2->4->3
```

Sample Input 1

```
8
1 2 3 4 5 5 7 3
```

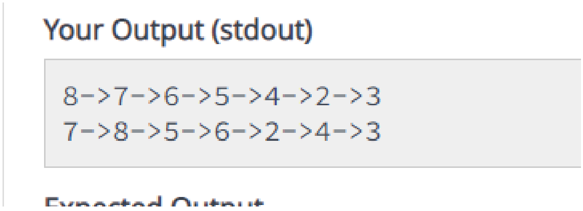
Sample Output 1

```
3->7->5->5->4->3->2->1
3->7->5->5->3->4->1->2
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        LinkedList ll = new LinkedList();
        for(int i=0; i<n; i++){
            int t = sc.nextInt();
            ll.add(t);
        }
        for(int i=n-1; i>0; i--){
            System.out.print(ll.get(i)+"->");
        }
        System.out.println(ll.get(0));
        int x = 1;
        for(int i=n-1; i>0; i--){
            if(x == 1){
                if((int)ll.get(i)>(int)ll.get(i-1)){
                    int temp = (int)ll.get(i);
                    ll.set(i, (int)ll.get(i-1));
                    ll.set(i-1, temp);
                }
                x = 0;
            }
            else{
                if((int)ll.get(i)<(int)ll.get(i-1)){
                    int temp = (int)ll.get(i);
                    ll.set(i, (int)ll.get(i-1));
                    ll.set(i-1, temp);
                }
                x = 1;
            }
        }
        for(int i=n-1; i>0; i--){
            System.out.print(ll.get(i)+"->");
        }
        System.out.println(ll.get(0));
    }
}
```

Output:



```
8->7->6->5->4->2->3
7->8->5->6->2->4->3
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 8 **COLLECTION INTERFACE- II**

Ex No: E8a DATE: 05-06-2022

1. Write a program to insert elements in the stack and display the topmost element.

Input Format

Number of inputs Stack elements

Constraints

Nil

Output Format

Topmost element in the stack

Sample Input 0

```
5
3 9 4 7 6
```

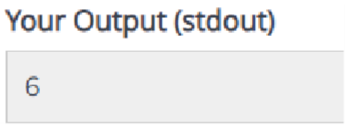
Sample Output 0

```
6
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
   public static void main(String args[] ) throws Exception {
      Scanner sc = new Scanner(System.in);
      int n = sc.nextInt();
      Stack s = new Stack();
      for(int i=0; i<n; i++){
         s.push(sc.nextInt());
      }
      System.out.println(s.peek());
   }
}
```

Output:



Your Output (stdout)

```
6
```

Result: Thus the above program is compiled and executed successfully.

2. Write a program to delete the top element in a stack and display the stack elements.

Input Format

Number of inputs Stack elements

Constraints

Nil

Output Format

Deleted element in the stack Stack elements Refer sample output

Sample Input 0

```
5
3 6 4 9 8
```

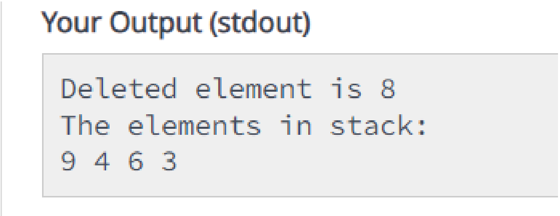
Sample Output 0

```
Deleted element is 8
The elements in stack:
9 4 6 3
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        Stack s = new Stack();
        for(int i=0 ;i <n; i++){
           s.push(sc.nextInt());
        }
        System.out.println("Deleted element is "+ s.pop()+"\nThe elements in stack:");
        for(int i=0; i<n-1; i++){
           System.out.print(s.pop()+" ");
        }
    }
}
```

Output:



Your Output (stdout)

```
Deleted element is 8
The elements in stack:
9 4 6 3
```

Result: Thus the above program is compiled and executed successfully.

3. Write a Java program to implement a stack and search for an element in the stack.

Input Format

Number of inputs Stack elements Searching element in the stack

Constraints

Nil

Output Format

Output prints whether the element is found or not

Sample Input 0

```
5
1 2 3 4 5
3
```

Sample Output 0

```
Element found
```

Sample Input 1

```
5
1 2 3 4 5
```

```
37
```

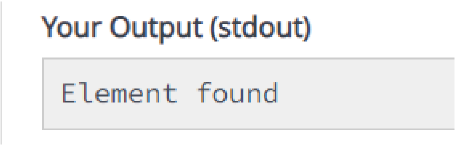
Sample Output 1

```
Element not found
```

Program Code:

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {
  public static void main(String args[] ) throws Exception {
    Scanner sc = new Scanner(System.in);
    int n = sc.nextInt();
    Stack s = new Stack();
    for(int i=0; i<n; i++){
      s.push(sc.nextInt());
    }
    int x = sc.nextInt();
    for(int i=0; i<n; i++){
      if((int)s.pop() == x){
        System.out.println("Element found");
        return;
      }
    }
    System.out.println("Element not found");
  }
}
```

Output:



```
Element found
```

Result: Thus the above program is compiled and executed successfully.

4. Write a Java program using collection concept to reverse a string s using stack.

Step 1:

Call createStack() inside main method with needed parameters.

Step 2:

Call push() function to push each character from the input string.

Step 3:

Call pop() function inside a reverse() function and print the characters in the reverse order.

Input Format

Your editor reads a single line containing a string called s

Constraints

1 < s < 100

Output Format

Displays the reversed string.

Sample Input 0

```
kumar
```

Sample Output 0

```
ramuk
```

Sample Input 1

```
Sample Input
```

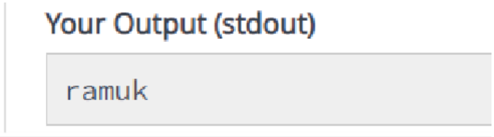
Sample Output 1

```
tupnI elpmaS
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution {

    public static void main(String args[] ) throws Exception {
        Scanner sc = new Scanner(System.in);
        Stack s = new Stack();
        String str = sc.nextLine();
        for(int i=0; i<str.length(); i++){
            s.push(str.charAt(i));
        }
        for(int i=0; i<str.length(); i++){
            System.out.print(s.pop());
        }
    }
}
```

Output:



Your Output (stdout)

```
ramuk
```

Result: Thus the above program is compiled and executed successfully.

5 Write a Java program to check whether the string is palindrome or not using Stack.

Implement this program using collection concept.

Input Format

Input consist of a string

Constraints

Nil

Output Format

Output prints whether the string is palindrome or not.

Sample Input 0

```
malayalam
```

Sample Output 0

```
malayalam is palindrome
```

Sample Input 1

```
raser
```

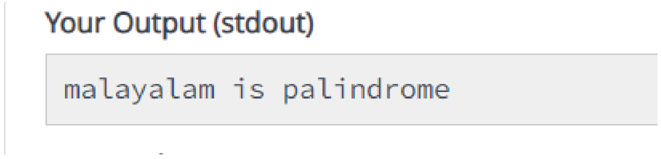
Sample Output 1

```
raser is not a palindrome
```

Program Code:

```
import java.io.*;

import java.util.*;

import java.text.*;

import java.math.*;

import java.util.regex.*;


public class Solution {

    public static void main(String args[] ) throws Exception {

        Scanner sc = new Scanner(System.in);

        Stack s = new Stack();

        String str = sc.next();

        String f = str,l="";

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

            s.push(str.charAt(i));

        }

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

            l+=s.pop();

        }

        if(f.equals(l)){

            System.out.println(f+" is palindrome");

        }

        else{

            System.out.println(f+" is not a palindrome");

        }

    }

}
```

Output:



Your Output (stdout)

```
malayalam is palindrome
```

Result: Thus the above program is compiled and executed successfully.

Ex No: E8b DATE: 10-06-2022

1. Try using a Set in your application. The property of Set is, it doesn't allow duplicate elements and does not maintain order like a list. Understand it by going through and completing the problem.

In the Main method, obtain username input from the user.

Add the usernames to a set.

Display the number of unique usernames at the end of the program.

Input Format

The first line of the input consists of the number of users.

Next input is the user names.

Constraints

Nil

Output Format

The output displays the number of unique elements in the set.

Sample Input 0

```
4
Ram
Christopher
Ahamed
Ahamed
```

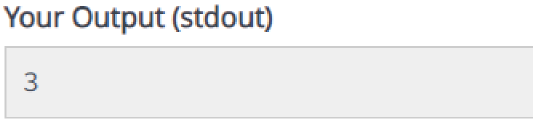
Sample Output 0

```
3
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        HashSet h = new HashSet();
        for(int i=0; i<n; i++){
          h.add(sc.next());
        }
        System.out.println(h.size());
    }
}
```

Output:



Your Output (stdout)

```
3
```

Result: Thus the above program is compiled and executed successfully

2. Experiment with containsAll() method in this problem.

Create Solution class. Obtain email addresses from the user and add them to a Set. At last get a String that has multiple email addresses in CSV format. Print "Email addresses are present" if all email addresses are present in the Set, else print "Email addresses are not present".

Input Format

The first line of the input consists of the number of email addresses.

Next input is the email addresses.

The last input is the email addresses to be searched in CSV format.

Constraints

Nil

Output Format

The output prints whether the email addresses are present or not.

Sample Input 0

```
4
Merry@gmail.com
Peter@yahoo.com
Christian@hotmail.com
Merry@gmail.com
Merry@gmail.com,Peter@yahoo.com
```

Sample Output 0

```
Email addresses are present
```

Sample Input 1

```
2
Manikandan@yahoo.com
Bala@google.co.in
Bala@google.co.in,jerry@gmail.com
```

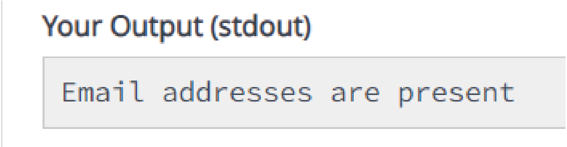
Sample Output 1

```
Email addresses are not present
```

Program Code:

```
import java.io.*;
import java.util.*;
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        HashSet h = new HashSet();
        for(int i=0; i<n; i++){
            h.add(sc.next());
        }
        String str = sc.next();
        String[] s = str.split(",", 1000);
        ArrayList<String> al = new ArrayList<String>(Arrays.asList(s));
        if(h.containsAll(al))
            System.out.println("Email addresses are present");
        else
            System.out.println("Email addresses are not present");
    }
}
```

Output:



Your Output (stdout)

```
Email addresses are present
```

Result: Thus the above program is compiled and executed successfully

3. In the Main method, obtain username input from the user. Add the usernames to a set. What if the user of our application enter a null value accidentally, will it be stored in the set? Go on try it yourself and understand the effect. Create the Solution class and get Username input from the user and display the number of unique usernames at the end.

Input Format

The first line of the input is the number of users.

Next input is the user names.

Constraints

Nil

Output Format

The output prints the number of unique elements in the list.

Sample Input 0

```
6
Mary
Roshan

Mary

Roshan
```

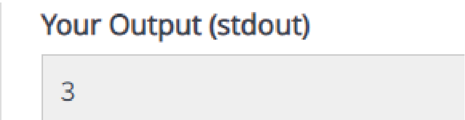
Sample Output 0

```
3
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashSet h = new HashSet();
        int n = sc.nextInt();
        for(int i=0; i<n; i++){
            String s = "";
            s += sc.nextLine();
            h.add(s);
        }
        System.out.println(h.size());
    }
}
```

Output:



```
Your Output (stdout)

3
```

Result: Thus the above program is compiled and executed successfully

4. Playing cards during travel is a fun-filled experience. For this game, they wanted to collect all four unique symbols. Can you help these guys to collect unique symbols from a set of cards?

Create Solution class with attributes symbol and number. From our main method collect each card details (symbol and number) from the user. Collect all these cards in a set, since the set is used to store unique values or objects. Cards need to be compared with each other to identify whether both the cards are the same symbol. For this, we need to implement equals method to tell whether both the cards are same or not.

Once we collect all four different symbols display the first occurrence of card details in alphabetical order.

Input Format

Input a card symbol

Input a card number

Repeat

Constraints

Nil

Output Format

Four symbols gathered in N cards

Cards in set

Sample Input 0

```
a
2
b
```

```
1
a
1
c
3
d
5
```

Sample Output 0

```
Four symbols gathered in 5 cards
Cards in Set are:
a 2
b 1
c 3
d 5
```

Sample Input 1

```
x
5
b
1
x
9
c
3
d
5
```

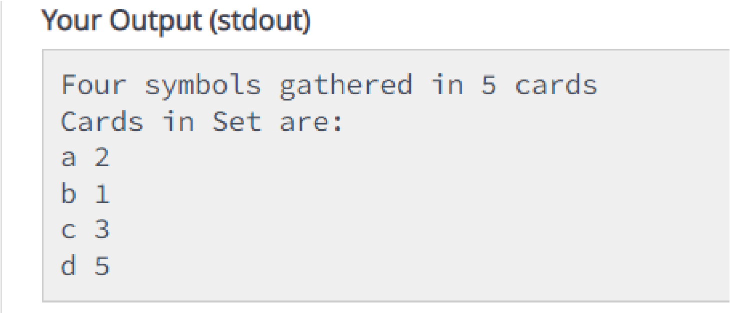
Sample Output 1

```
Four symbols gathered in 5 cards
Cards in Set are:
b 1
c 3
d 5
x 5
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Map<String, Integer> m = new TreeMap<>();
        int temp = 0,total = 0;
        while(temp!=4){
            String x = sc.next();
            int num = sc.nextInt();
            if(!m.containsKey(x)){
                m.put(x,num);
                temp++;
            }
            total+=1;
        }
        System.out.println("Four symbols gathered in "+total+" cards");
        System.out.println("Cards in Set are: ");
        for(Map.Entry<String, Integer> e : m.entrySet()){
            System.out.println(e.getKey() + " "+ e.getValue());
        }


    }
}
```

Output:



Your Output (stdout)

```
Four symbols gathered in 5 cards
Cards in Set are:
a 2
b 1
c 3
d 5
```

Result: Thus the above program is compiled and executed successfully

EXERCISE : 9 EXCEPTION HANDLING

DATE: 12-06-2022

1. An exception is an unwanted or unexpected event, which occurs during the execution of a program i.e at runtime, it disrupts the normal flow of the program. For example, there are 10 statements in your program and there occurs an exception at statement 5, rest of the code will not be executed i.e. statement 6 to 10 will not run. If we perform exception handling, rest of the statement will be executed. That is why we use exception handling.

For practice in exception handling, obtain the cost for 'n' days of an item and n as input and calculate the cost per day for the item. In case, zero is given as input for n, an arithmetic exception is thrown, handle the exception and prompt the user accordingly (Refer sample I/O).

Create a driver class called Solution. In the Main method, obtain input from the user and store the values in int type. Handle exception if one occurs.

Input Format

First line of the input consist of the cost of item for n days

Second line of the input consist of value of n

Constraints

Nil

Output Format

Output prints the cost per day of the item

Sample Input 0

```
100
0
```

Sample Output 0

```
Cannot divide by zero
```

Sample Input 1

```
100
20
```

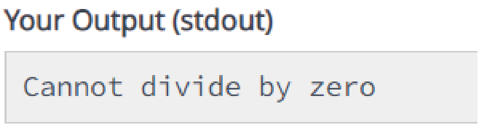
Sample Output 1

```
5
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int x = sc.nextInt();
        try{
            System.out.println(n/x);
        }
        catch(ArithmeticException e){
            System.out.println("Cannot divide by zero");
        }
    }
}
```

Output:



Your Output (stdout)

```
Cannot divide by zero
```

Result: Thus the above program is compiled and executed successfully

2. The prominent exception which you will see is ArrayIndexOutOfBoundsException. It occurs when the program try to access the array beyond its size. As we know arrays have fixed size. So when you try to use array beyond its size it throws this exception. Let's try to handle this exception.

Handling this exception will also prove to be good for our application. For example, if there are only 100 seats in the event and user try to book a 105th seat, it will throw this exception. So you must handle it to do a specific job.

Create an array of size 100 and obtain the input from user. Get the value at specified index. Handle any exception that occurs in Solution Class.

Input Format

First line of the input consist of size of an array

Next input is the elements of the array .

Fina line of the input consist of the index number to be searched.

Constraints

Nil

Output Format

Output prints the value at specified index.

Sample Input 0

```
5
2
3
6
8
4
2
```

Sample Output 0

```
Value at 2 = 6
```

Sample Input 1

```
5
2
3
6
8
4
7
```

Sample Output 1

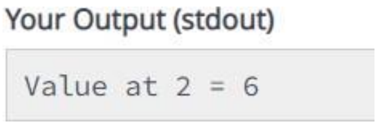
```
Array index out of bound.
```

Program Code:

```
import java.io.*;
import java.util.*;


public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
```

```
         int[] a = new int[n];
        for(int i=0; i<n; i++){
            a[i] = sc.nextInt();
        }
        int x = sc.nextInt();
        try{
            System.out.println("Value at " + x + " = " + a[x]);
        }
        catch(IndexOutOfBoundsException e){
            System.out.println("Array index out of bound.");
        }
    }
}
```

Output:



Your Output (stdout)

```
Value at 2 = 6
```

Result: Thus the above program is compiled and executed successfully

3. Create a class Student with attributes roll no, name, age and course. Initialize values through the parameterized constructor.

If the age of a student is not in between 15 and 21 then generate user-defined exception "AgeNotWithinRangeException". If the name contains numbers or special symbols raise exception "NameNotValidException". Define the two exception classes to display the message as shown in sample output.

Input Format

Student ID

Name

Age

Course

Constraints

Nil

Output Format

Print student details with Exception(if any)

Sample Input 0

```
100

Babu

20

MCA
```

Sample Output 0

```
100 Babu 20 MCA
```

Sample Input 1

```
100

Babu

24

MCA
```

Sample Output 1

```
Age is not between 15 and 21

100 Babu 24 MCA
```

Sample Input 2

```
100

Babu3

20

MCA
```

Sample Output 2

```
Name is not Valid

100 Babu3 20 MCA
```

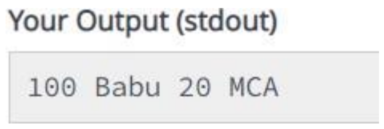
Program Code:

```
import java.io.*;

import java.util.*;


class Student{

    int roll,age;

    String name,course;

    Student(int r,String n,int a,String c){

        this.roll=r;
```

```
     this.course=c;
    this.age = a;
    this.name = n;
    int l,temp=0;
     l = n.length();
     for(int i=0;i<l;i++){
         char ch;
         ch=n.charAt(i);
         if(ch<'A' || ch>'Z' && ch<'a' || ch>'z')
             temp=1;
     }
     try{
         if(temp==1)
             throw new NameNotValidException();
         else
             this.name=n;
     }
     catch(NameNotValidException e2){
         System.out.println(e2.validname());
     }
     try{
         if(a>=15 && a<=21)
             this.age=a;
         else
             throw new AgeNotWithInRangeException();
     }
     catch(AgeNotWithInRangeException e1){
         System.out.println(e1);
     }
 }
 void display(){
     System.out.println(this.roll+" "+this.name+" "+this.age+" " +this.course);
 }
```

```
}
class NameNotValidException extends Exception{
    public String validname(){
        return ("Name is not Valid");
    }
}
class AgeNotWithInRangeException extends Exception
{
    public String toString()
    {
        return ("Age is not between 15 and 21");
    }
}
public class Solution {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        String s = sc.next();
        int x = sc.nextInt();
        String d = sc.next();
        Student ss = new Student(n,s,x,d);
        ss.display();
    }
}
```

Output:



Your Output (stdout)

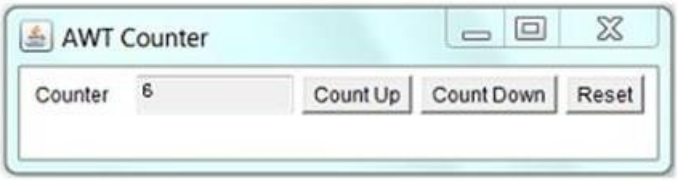
```
100 Babu 20 MCA
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 10 **AWT AND SWING COMPONENT**

DATE: 14-06-2022

1. Develop the following using AWT and swing component:

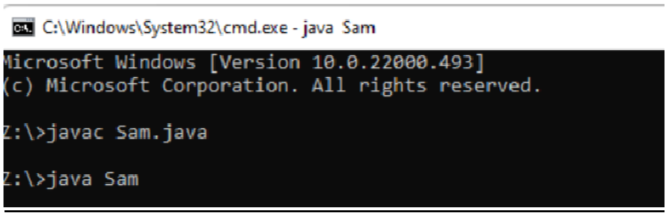


Program Code:

```
import java.util.*;
import javax.swing.*;
import java.awt.*;
 import java.awt.event.*;
public class Sam extends Frame implements ActionListener {
TextField tf; Button b1,b2,b3; Label l1; int count=0; Sam() {
tf=new TextField("Awt counter");
tf.setBounds(130,50,100,30);
b1 = new Button("Counter up");
b2 = new Button("Counter down");
b3 = new Button("Reset");
l1 = new Label("Counter");
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b1.setBounds(300,50,130,30);
b2.setBounds(450,50,130,30);
b3.setBounds(600,50,130,30);
add(b1);
 add(b2);
add(b3);
add(l1);
add(tf);
setVisible(true);
setSize(400,400);
```

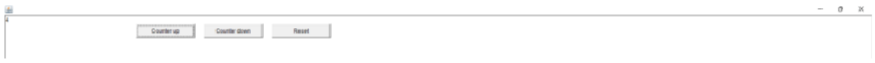
```
 }
 public void actionPerformed(ActionEvent e) {
 if(e.getSource() == b1) {
count++;
String plus = String.valueOf(count); tf.setText(plus);
}
if(e.getSource() == b2) {
count--;
 String minus = String.valueOf(count);
tf.setText(minus);
}
if(e.getSource() == b3)
{ count=0;
String reset = String.valueOf(count); tf.setText(reset);
}
}
public static void main(String[] args) {
new Sam();
}
}
```

Output:



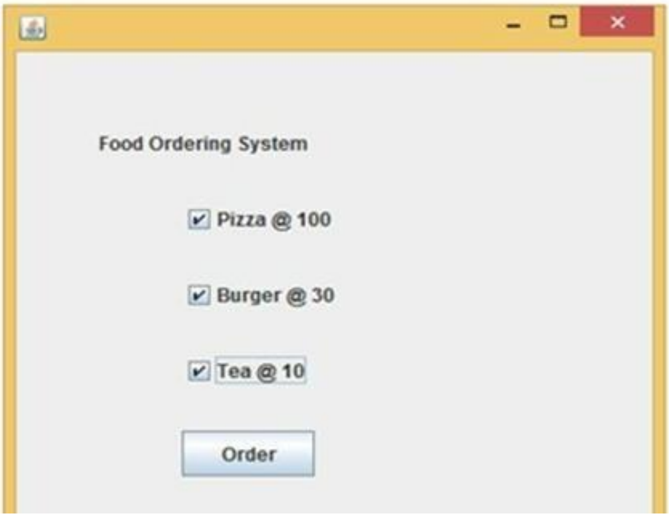
C:\Windows\System32\cmd.exe - java Sam

```
Microsoft Windows [Version 10.0.22000.493]
(c) Microsoft Corporation. All rights reserved.

Z:\>javac Sam.java

Z:\>java Sam
```



Result: Thus the above program is compiled and executed successfully.

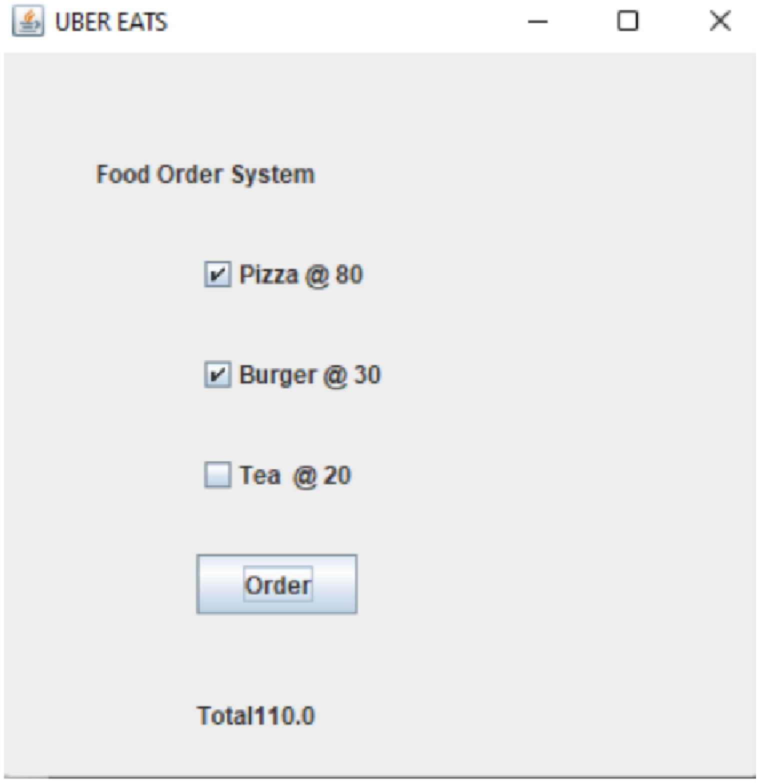
2. Develop food ordering system using checkbox component in swing. Calculate the total bill amount based on the user selection. Add a label control or a textfield control below to the order button to display the bill amount when the user clicks the order button.



Program Code:

```
import javax.swing.*;
import java.awt.event.*;
public class DisplayCheckBox extends JFrame implements ActionListener{
JFrame f;
JLabel l,l1;
 JCheckBox cb1,cb2,cb3;
JButton b;
DisplayCheckBox() {
f=new JFrame();
setTitle("UBER EATS");
l= new JLabel("Food Order System");
l.setBounds(50,50,300,20);
l1= new JLabel();
l1.setBounds(100,230,200,200);
cb1=new JCheckBox("Pizza @ 80");
cb1.setBounds(100,100,150,20);
cb2=new JCheckBox("Burger @ 30");
cb2.setBounds(100,150,150,20);
cb3=new JCheckBox("Tea @ 20");
cb3.setBounds(100,200,150,20);
b=new JButton("Order");
b.setBounds(100,250,80,30);
b.addActionListener(this);
add(l);
add(l1);
add(cb1);
add(cb2);
add(cb3);
add(b);
setSize(400,400);
setLayout(null);
setVisible(true);
setDefaultCloseOperation(EXIT_ON_CLOSE);
}
public void actionPerformed(ActionEvent e) {
float amount=0;
String msg="Total"; if(cb1.isSelected()) {
amount+=80;
}
if(cb2.isSelected()) {
amount+=30;
}
if(cb3.isSelected()) {
amount+=20;
}
msg+=Float.toString(amount);
l1.setText(msg);
}
public static void main(String[] args) {
new DisplayCheckBox();
}
}
```

Output:



UBER EATS

Food Order System

☑ Pizza @ 80

☑ Burger @ 30

□ Tea @ 20

Order

Total110.0

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 11 **THREADS**

DATE: 17-06-2022

1. Write a program WordCount that counts the words in one or more files. Start a new thread for each file. For example, if you call java WordCount report.txt address.txt Homework.java then the program might print

address.txt: 1052

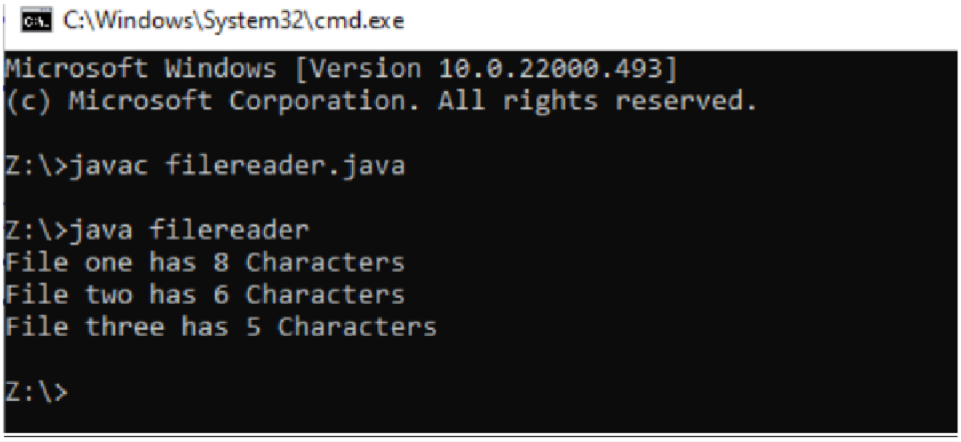
Homework.java: 445

report.txt: 2099

Program Code:

```
import java.io.*;
import java.util.Scanner;
public class filereader implements Runnable {
static int count = 0;
static int count1 = 0;
static int count2 = 0;
int c;
public void run() {
try {
File file1 = new File("test1.txt");
File file2 = new File("test2.txt");
File file3 = new File("test3.txt");
FileInputStream f1 =new FileInputStream(file1);
FileInputStream f2 =new FileInputStream(file2);
FileInputStream f3 =new FileInputStream(file3);
while ((c = f1.read()) != -1) {
 count++;
}
 while ((c = f2.read()) != -1) {
count1++;
}
 while ((c = f3.read()) != -1) {
 count2++;
}
}
catch(Exception e) {
 count = -1;
count1 = -1;
count2 = -1;
System.out.println("Error: " + e);
}
}
public static void main(String[] args) {
try{
Thread tr = new Thread(new filereader());
 tr.start();
 tr.join();
System.out.println("File one has " + count + " Characters");
System.out.println("File two has " + count1 + " Characters");
System.out.println("File three has " + count2 + " Characters");
}
catch (InterruptedException ex) { System.out.println("Error: " + ex);
}
}
}
```

<u>Output:</u>



```
Microsoft Windows [Version 10.0.22000.493]
(c) Microsoft Corporation. All rights reserved.

Z:\>javac filereader.java

Z:\>java filereader
File one has 8 Characters
File two has 6 Characters
File three has 5 Characters

Z:\>
```

<u>Result:</u> Thus the above program is compiled and executed successfully.

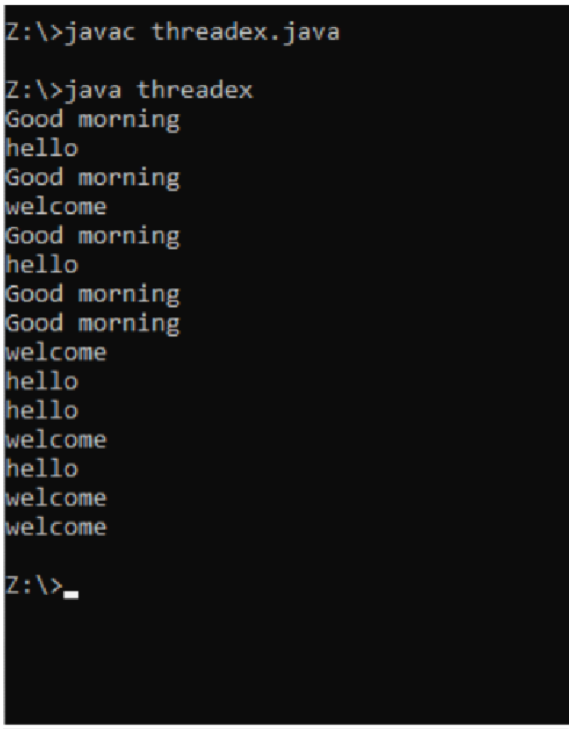
2. Write a Java program that creates three threads. First thread displays “Good Morning” every one second, the second thread displays “Hello” every two seconds and the third thread displays “Welcome” every three seconds.

Program Code:

```
class A extends Thread
{
synchronized public void run()
{
try
 {
 int i=0; while (i<5) { sleep(1000);
 System.out.println("Good morning "); i++;
 }
 }
catch (Exception e)
{
 }
 }
}
class B extends Thread
{
synchronized public void run()
{
 try
 {
 int i=0; while (i<5)
{ sleep(2000);
 System.out.println("hello"); i++;
 }
 } catch (Exception e) {
 }
 }
}
class C extends Thread { synchronized public void run() { try
 {
```

```
 int i=0; while (i<5)
{
sleep(3000);
 System.out.println("welcome"); i++;
 }
 } catch (Exception e) {
 }
 }
}
class threadex { public static void main(String args[])
{
A a = new A();
B b = new B();
C c = new C();
a.start();
 b.start();
 c.start();
 }
}
```

Output:



```
Z:\>javac threadex.java

Z:\>java threadex
Good morning
hello
Good morning
welcome
Good morning
hello
Good morning
Good morning
welcome
hello
hello
welcome
hello
welcome
welcome

Z:\>
```

Result: Thus the above program is compiled and executed successfully.

EXERCISE : 12 **JAVA JDBC**

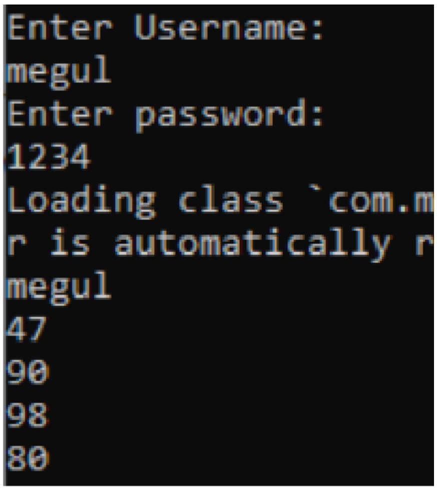
DATE: 20-06-2022

1. Write a program to display student exam scores. The program prompts the user to enter the user's first name and the password in a dialog box. Upon receiving the correct user name and password, the program displays the student's full name and the exam score using java JDBC concepts.

Program Code:

```
import  java.sql.*;
import java.util.*;
class DBConn {
public static void main(String[] args) {
String name, pass;
Scanner sc=new Scanner(System.in);
System.out.println("Enter Username: ");
name=sc.next();
System.out.println("Enter password: ");
pass = sc.next();
try {
Class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/loginuser","root","");
Statement stmt = con.createStatement();
String query="select * from login_table where userid='"+name+"' AND pwd='"+pass + "'";
//System.out.println(query);
 ResultSet rs = stmt.executeQuery(query);
//rs.last();
//System.out.println(rs.getRow());
while(rs.next()) {
System.out.println(rs.getString("name"));
System.out.println(rs.getInt("regno"));
System.out.println(rs.getInt("sub1"));
System.out.println(rs.getInt("sub2"));
System.out.println(rs.getInt("sub3"));
}
}
catch(Exception e)
{ e.printStackTrace();
}
 }
 }
```

Output:



```
Enter Username:
megul
Enter password:
1234
Loading class `com.m
r is automatically r
megul
47
90
98
80
```

Result: Thus the above program is compiled and executed successfully.