**1.Java Exception Handling (Try-catch)**

You will be given two integers ***x***  and ***y*** as input, you have to compute ***x/y***. If ***x*** and ***y***  are not ***32*** bit signed integers or if ***y*** is zero, exception will occur and you have to report it. Read sample Input/Output to know what to report in case of exceptions.

**Sample Input 1**

10

3

**Sample Output 1**

3

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

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 scan = new Scanner(System.in);

try {

int x = scan.nextInt();

int y = scan.nextInt();

System.out.println(x / y);

}

catch(InputMismatchException e) {

// ensure that "java.util.InputMismatchException" is printed only.

System.out.println(e.getClass().toString().replaceFirst("class ", ""));

}

catch(ArithmeticException e) {

// Print F

System.out.println(e);

}

scan.close();

}

}

2.Rithvik likes to play with numbers, every day he used to solve one Brain Teaser, today he attempted to solve the GAPFUL NUMBER.

A gapful number is a number of 3 digits that is divisible by the number formed by the first and last digits of the original number. Help him write a program to check whether the given number is a gapful number or not.

**For Example:**

Input:192

Output : true(192 is a gapful number because it is divisible by 12)

import java.util.Scanner;

class Main

{

public static void main(String[] args) {

final int MINIMUM = 100;

String input = "";

int first\_number = 0;

int second\_number = 0;

input = new Scanner(System.in).nextLine();

try {

first\_number = Integer.parseInt(input);

if (first\_number < MINIMUM) {

System.out.println("Input a number of at least 3 digits!");

return;

}

} catch (NumberFormatException nfe) {

System.out.println("Input a valid number!");

return;

}

second\_number = Integer.parseInt("" + input.charAt(0) + input.charAt(input.length() - 1));

if (first\_number % second\_number == 0) {

System.out.println("True");

} else {

System.out.println("False");

}

}

}

3.Get five integers from user and store it in an Array ARR[],next get an integer 'X' and integer 'Y' from user.

Then print ARR[X] / ARR[Y].

(int ARR[]=new int[5];)

if X or Y is not in the range of index of ARR[] i.e., 0 to 4 ,then the JVM will throw an Exception, handle that Exception with proper Exception Handling class by printing "index must be 0 to 4";

if ARR[Y] is equal to 0(zero),then the JVM will throw an Exception, handle that Exception with proper Exception Handling class by printing "denominator should not be zero";

If there is no Exception in the program then print ARR[X] divided by ARR[Y] (Quotient)

**Use Exception Handling Mechanism :(try, catch..)**

**[Hint:- Multiple catch block must be used]**

**Input Format**

The first five integers are the values for the Array (ARR)

The next integer 'X' represents the index of ARR.

The next integer 'Y' represents the index of ARR.

**Output Format**

An integer i.e. the result of ARR[X] / ARR[Y]

Or

index must be 0 to 4

Or

denominator should not be zero

**Sample Input 1**

3 4 5 1 2

2 4

**Sample Output 1**

2

**Sample Input 2**

5 6 0 1 3

4 2

**Sample Output 2**

denominator should not be zero

**Sample Input 3**

7 8 9 0 1

5 3

**Sample Output 3**

index must be 0 to 4

import java.util.Scanner;

class Main

{

public static void main(String args[])

{

int ARR[]=new int[5];

int X,Y;

Scanner ip=new Scanner(System.in);

for(int i=0;i<5;i++)

ARR[i]=ip.nextInt();

X=ip.nextInt();

Y=ip.nextInt();

try

{

System.out.println(ARR[X]/ARR[Y]);

}

catch(ArithmeticException e)

{

System.out.println("denominator should not be zero");

}

catch(ArrayIndexOutOfBoundsException e)

{

System.out.println("index must be 0 to 4");

}

catch(Exception e)

{

System.out.println("UnKnown Exception");

}

}

}

4.Create your own exception called "IncorrectAgeException". It should display the message as “Please Enter Correct Age”.

Your code should get age as input from the user.

If age is greater than 150, throw an IncorrectAgeException.

If age is less than or equal to 150, display "welcome" message.

**Input Format**

151

**Output Format**

IncorrectAgeException: Please Enter Correct Age

**Sample Input 1**

151

**Sample Output 1**

IncorrectAgeException: Please Enter Correct Age

import java.util.\*;

class IncorrectAgeException extends Exception

{

public IncorrectAgeException(String m)

{

super(m);

}

}

class person

{

int age;

person (int a)

{

age=a;

}

public void checkage() throws IncorrectAgeException

{

if (age>150)

throw new IncorrectAgeException("Please Enter Correct Age");

else

System.out.println("Welcome");

}

}class Main

{

public static void main(String args[])

{

Scanner scan=new Scanner(System.in);

int a;

a = scan.nextInt();

person p = new person(a);

try

{

p.checkage();

}

catch(IncorrectAgeException e)

{

System.out.println(e);

}

}

}

5.Swap the values in the LinkedList <String>

Get Ten Strings from the user and put them in a LinkedList<String> object, and swap the position second string with fifth string.

**Input Format**

Ten Strings without white space(s)

All the strings must be in small case.

**Output Format**

Ten Strings after swapping the second with fith position

**Sample Input 1**

qwerty

asdfg

mnop

cvbn

poiuy

bhuijn

sge

dgdfgasf

rty

cvbcvb

**Sample Output 1**

qwerty

poiuy

mnop

cvbn

asdfg

bhuijn

sge

dgdfgasf

rty

cvbcvb

**Sample Input 2**

a

b

c

d

e

f

g

h

i

j

**Sample Output 2**

a

e

c

d

b

f

g

h

i

j

import java.util.\*;

class Main

{

public static void main(String[]args)

{

Scanner ip=new Scanner(System.in);

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

for(int i=0;i<10;i++)

{

list.add(ip.next());

}

Collections.swap(list,1,4);

for(String str:list)

System.out.println(str);

}

}

6.Get 10 integers from user,find and print the second largest number.

If all numbers are same i.e no second largest number then print -1.

**Sample Input 1**

5 5 5 5 5 5 5 5 5 5

**Sample Output 1**

-1

**Sample Input 2**

1 2 3 4 5 6 7 8 9 0

**Sample Output 2**

8

**Sample Input 3**

7 7 7 7 7 6 6 6 6 3

**Sample Output 3**

6

import java.util.\*;

class Main

{

public static void main(String[]args)

{

Scanner ip=new Scanner(System.in);

TreeSet<Integer> ts=new TreeSet<Integer>();

for(int i=0;i<10;i++)

{

ts.add(ip.nextInt());

}

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

if(list.size()>1)

{

System.out.println(list.get(list.size()-2));

}

else

{

System.out.println("-1");

}

}

}

7.Write a java program to create a own exception called DivisiblebyFiveException it should display an message called "Number should not be divided by five". Get an input from the user.

If number divisble by 5 it should raise an exception called DivisiblebyFiveException.

Otherwise Print Valid Number.

**Input Format**

565

**Output Format**

DivisiblebyFiveException: Number should not be divide by five

**Sample Input 1**

565

**Sample Output 1**

DivisiblebyFiveException: Number should not be divide by five

import java.util.Scanner;

import java.io.\*;

class DivisiblebyFiveException extends Exception

{

public DivisiblebyFiveException(String message)

{

super(message);

}

}

class exe

{

public static void main(String[] args)

{

Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

try

{

if(x%5==0)

throw new DivisiblebyFiveException("Number should not be divide by five");

else

System.out.print("Valid Number");

}

catch(Exception e)

{

System.out.print(e);

}

}

}

8.Create three threads namely Thread1,Thread2,Thread3 in java.,

The Thread1 should print 2,4,6,....20.

The Thread2 should print 5,10,15,...100.

The Thread3 should print 11,22,33,...110.

(Its Multithreaded program so,the output order is depends upon the system and time and it's cosidered)

**Sample Input 1**

**Sample Output 1**

2 5 10 15 20 25 30 4 35 40 45 50 11 22 33 44 55 66 77 88 99 110 6 8 10 12 14 16 18 20

**Sample Input 2**

**Sample Output 2**

2 4 6 8 10 12 14 16 18 20 5 10 15 20 25 30 35 40 45 50 11 22 33

class A extends Thread

{

public void run()

{

for(int i=1;i<=10;i++)

{

System.out.print(" "+i\*2);

}

}

}

class B extends Thread

{

public void run()

{

for(int i=1;i<=10;i++)

{

System.out.print(" "+i\*5);

}

}

}

class C extends Thread

{

public void run()

{

for(int i=1;i<=10;i++)

{

System.out.print(" "+i\*11);

}

}

}

class MyClass {

public static void main(String args[]) {

A t1=new A();

B t2=new B();

C t3=new C();

t1.start();

t2.start();

t3.start();

}

}

9.Find the middle of a given linked list

Given a singly linked list, find the middle of the linked list.

For example, if the given linked list is 1->2->3->4->5 then the output should be 3.

If there are even nodes, then there would be two middle nodes, we need to print the second middle element.

For example, if given linked list is 1->2->3->4->5->6 then the output should be 4.

**Input Format**

n - get size of linked list

get number elements in linked list

5

1

2

3

4

5

**Output Format**

Print the elements of the Linked List

Print middle element of the linked List

5->4->3->2->1->NULL

The middle element is [3]

**Sample Input 1**

5

1

2

3

4

5

**Sample Output 1**

5->4->3->2->1->NULL

The middle element is [3]

import java.util.\*;

class LinkedList

{

Node head;

class Node

{

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

/\* Function to print middle of linked list \*/

void printMiddle()

{

Node slow\_ptr = head;

Node fast\_ptr = head;

if (head != null)

{

while (fast\_ptr != null && fast\_ptr.next != null)

{

fast\_ptr = fast\_ptr.next.next;

slow\_ptr = slow\_ptr.next;

}

System.out.println("The middle element is [" +

slow\_ptr.data + "] \n");

}

}

public void push(int new\_data)

{

Node new\_node = new Node(new\_data);

new\_node.next = head;

head = new\_node;

}

public void printList()

{

Node tnode = head;

while (tnode != null)

{

System.out.print(tnode.data+"->");

tnode = tnode.next;

}

System.out.println("NULL");

}

}

class Main {

public static void main(String [] args)

{

LinkedList llist = new LinkedList();

Scanner scan = new Scanner(System.in);

int n = scan.nextInt();

for (int i=1;i<=n;i++)

llist.push(scan.nextInt());

llist.printList();

llist.printMiddle();

}}

10.A string containing only parentheses is balanced if the following is true: 1. if it is an empty string 2. if A and B are correct, AB is correct, 3. if A is correct, (A) and {A} and [A] are also correct.

Examples of some correctly balanced strings are: "{}()", "[{()}]", "({()})"

Examples of some unbalanced strings are: "{}(", "({)}", "[[", "}{" etc.

Given a string, determine if it is balanced or not.

**Input Format**

There will be multiple lines in the input file, each having a single non-empty string. You should read input till the end-of-file.

The part of the code that handles input operation is already provided in the editor.

**Output Format**

For each case, print 'true' if the string is balanced, 'false' otherwise.

**Sample Input 1**

{}()

({()})

{}(

[]

**Sample Output 1**

true

true

false

true

import java.io.\*;

import java.util.\*;

import java.text.\*;

import java.math.\*;

import java.util.regex.\*;

class Solution {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

while(scanner.hasNextLine()) {

Stack<Character> stack = new Stack<>();

String line = scanner.nextLine();

for(char c : line.toCharArray()) {

if(c == '{' || c == '(' || c == '[') {

stack.push(c);

continue;

}

if(c == '}' && !stack.isEmpty() && stack.peek() == '{') {

stack.pop();

continue;

}

if(c == ')' && !stack.isEmpty() && stack.peek() == '('){

stack.pop();

continue;

}

if(c == ']' && !stack.isEmpty() && stack.peek() == '['){

stack.pop();

continue;

}

if(c == '}' || c == ')' || c == ']') {

stack.push(c);

break;

}

}

System.out.println(stack.isEmpty());

}

}

}

Base64.Encoder encoder = Base64.getMimeEnocder();

Str = encoder.encodeToString(str.getBytes());

11.Get a string from user and encode the string using Base64 Algorithm and print encode MIME message for the given string.

**Input Format**

A sting

**Output Format**

Encode MIME String

**Sample Input 1**

HELLO

**Sample Output 1**

SEVMTE8=

**Sample Input 2**

good morning

**Sample Output 2**

Z29vZA==

import java.util.\*;

class Main

{

public static void main(String[] args)

{

// Getting MIME encoder

Scanner ip=new Scanner(System.in);

Base64.Encoder encoder = Base64.getMimeEncoder();

String message = ip.next();

String eStr = encoder.encodeToString(message.getBytes());

System.out.println(eStr);

}

}

12.Neha is carrying out research in English Literature. Her research mainly focuses on English alphabets. As a part of her research, she wanted to know which characters are frequently used in English. She knew that this task would consume more time if carried out manually. Since you are a technical geek she wanted to check with you whether this can be done using computers at no time. You narrow down the problem and first compute the frequency of each character in a given word. And later you can extend this program to compute the character frequency in a novel. For this first, get a word as input and compute the frequency of all the character in a word. The A map will be your best choice to store frequencies for each character. Display all the character in alphabetical order and with their respective frequencies. Space is out of scope for this experiment.

Input Format:

A single line contain string s to perform operation

Output Format:

Refer sample output statement to print proper result

**Input Format**

DivideByZeroException

**Output Format**

Character frequency in given text :

B:1 D:1 E:1 Z:1 c:1 d:1 e:3 i:3 n:1 o:2 p:1 r:1 t:1 v:1 x:1 y:1

**Sample Input 1**

DivideByZeroException

**Sample Output 1**

Character frequency in given text:

B:1 D:1 E:1 Z:1 c:1 d:1 e:3 i:3 n:1 o:2 p:1 r:1 t:1 v:1 x:1 y:1

import java.util.Scanner;

import java.util.\*;

class Main

{

public static void main(String args[])

{

Scanner s=new Scanner(System.in);

String str =s.next();

TreeMap<Character,Integer> map=new TreeMap<Character,Integer>();

int n=str.length();

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

{

char c=str.charAt(i);

Integer val=map.get(c);

if(val!=null)

{

map.put(c,new Integer(val+1));

}

else

{

map.put(c,1);

}

}

System.out.println("Character frequency in given text:");

for(Map.Entry<Character,Integer>entry: map.entrySet())

{

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

}

}

}

13.You are required to compute the power of a number by implementing a calculator. Create a class MyCalculator which consists of a single method long power(int, int). This method takes two integers n, and p, as parameters and finds np. If either n or p is negative, then the method must throw an exception which says " n or p should not be negative". Also, if n both p and are zero, then the method must throw an exception which says "n and p should not be zero"

For example, -4 and -5 would result in

java.lang.Exception: n or p should not be negative

**Input Format**

Each line of the input contains two integers, n and p. The locked stub code in the editor reads the input and sends the values to the method as parameters.

**Output Format**

Each line of the output contains the result np , if both n and p are positive. If either n or p is negative, the output contains "n and p should be non-negative". If both n and p are zero, the output contains "n and p should not be zero.". This is printed by the locked stub code in the editor.

**Constraints**

-10<= n <= 10

-10<= p <= 10

**Sample Input 1**

3 5

2 4

0 0

-1 -2

-1 3

**Sample Output 1**

243

16

java.lang.Exception: n and p should not be zero.

java.lang.Exception: n or p should not be negative.

java.lang.Exception: n or p should not be negative.

import java.util.Scanner;

import static java.lang.Math.pow;

class MyCalculator {

Create the method long power(int, int) here.

public long power(final int n, final int p) throws Exception {

if (n < 0 || p < 0) {

throw new Exception("n or p should not be negative.");

} else if (n == 0 && p == 0) {

throw new Exception("n and p should not be zero.");

}

return (long)pow(n, p);

}

}

class MyClass {

public static final MyCalculator my\_calculator = new MyCalculator();

public static final Scanner in = new Scanner(System.in);

public static void main(String[] args) {

while (in .hasNextInt()) {

int n = in .nextInt();

int p = in .nextInt();

try {

System.out.println(my\_calculator.power(n, p));

} catch (Exception e) {

System.out.println(e);

}

}

}

}

14.Consider you have two sets and each set contains 5 integers,your task is to print the non common elements in both the sets.

if there is no non common elements in both the stes,then print -1.

**Input Format**

The first 5 integers are the values for Set1

The next 5 integers are the values for Set2.

**Output Format**

Print integers which are non common in both the sets,otherwise -1.

**Sample Input 1**

1 2 3 4 5

3 4 5 6 7

**Sample Output 1**

1 2 6 7

**Sample Input 2**

2 2 2 2 2

2 2 2 2 2

**Sample Output 2**

-1

import java.util.\*;

class Main

{

public static void main(String args[])

{

Scanner ip=new Scanner(System.in);

Set<Integer> a = new HashSet<Integer>();

Set<Integer> b = new HashSet<Integer>();

for(int i=0;i<5;i++)

a.add(ip.nextInt());

for(int i=0;i<5;i++)

b.add(ip.nextInt());

Set<Integer> union = new HashSet<Integer>(a);

union.addAll(b);

Set<Integer> intersection = new HashSet<Integer>(a);

intersection.retainAll(b);

union.removeAll(intersection);

if(union.size()==0)

System.out.println("-1");

else

{

for (int x:union)

System.out.print(" "+x);

}

}}

15.Get 10 integers from user,find and print the second largest number.

If all numbers are same i.e no second largest number then print -1.

[ Hint:- you may use any Collection classes like ArrayList,TreeSet,Hashset,LinkedList..etc]

**Sample Input 1**

7 7 7 7 7 6 6 6 6 3

**Sample Output 1**

6

**Sample Input 2**

4 4 4 4 4 4 4 4 4 4

**Sample Output 2**

-1

import java.util.\*;

class Main

{

public static void main(String[]args)

{

Scanner ip=new Scanner(System.in);

TreeSet<Integer> ts=new TreeSet<Integer>();

for(int i=0;i<10;i++)

{

ts.add(ip.nextInt());

}

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

if(list.size()>1)

{

System.out.println(list.get(list.size()-2));

}

else

{

System.out.println("-1");

}

}

}

16.Raju is a little boy studying the fifth standard in ABC matric higher secondary school. Next week is annual at his school. He wants to participate in a contest. His friend Babu registered himself and Raju in a contest. The contest was named as Memory Zone. Raju has no idea about the contest. So he willing to practice in order to prevent humiliation on an annual day. As a boy studying 5th standard, he not able to practice for the contest. The contest Memory Zone is about memorizing the words. Before the contest starts Judge give some words. Contestants have to memorize them and write them in the same order. Help Raju to practice this using a computer program. Note: Using LinkedList and Iterator

Input Format:

The first line contains value n denoting no of a word.

Next n line containing the word (One word per each line)

Output Format:

Print the words line by line as the given order

**Input Format**

3

Ball

Cap

Computer`c

**Output Format**

Ball

Cap

Computer

**Sample Input 1**

3

Ball

Cap

Computer

**Sample Output 1**

Ball

Cap

Computer

import java.util.LinkedList;

import java.util.Scanner;

import java.util.Iterator;

class Main

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

LinkedList<String> a=new LinkedList<String>();

int n,i;

n=sc.nextInt();

//sc.nextLine();

for(i=1;i<=n;i++)

{

String str=sc.next();

a.add(str);

}

Iterator<String> it=a.iterator();

while(it.hasNext())

{

System.out.print(it.next()+"\n");

}

}

}

17.Program to remove duplicate elements in an array (sorted and unsorted array cases) is discussed here. Given an array, all the duplicate elements of the array are removed.

For example, consider the array

case 1: Remove duplicates from sorted array

Input: arr = {1, 2, 3, 4, 4}

Output: arr = {1, 2, 3, 4}

case 2: Remove duplicates from unsorted array

Input: arr = {9, 2, 7, 4, 7}

Output: arr = {9, 2, 7, 4}

Input:

n - size of the array

Get n number of values

Output:

Array after removing duplicates

**18.**A palindrome is a word, phrase, number, or other sequence of characters which reads the same backwards and forwards. Can you determine if a given string, s, is a palindrome?

To solve this challenge, we must first take each character in s, enqueue it in a queue, and also push that same character onto a stack. Once that's done, we must dequeue the first character from the queue and pop the top character off the stack, then compare the two characters to see if they are the same; as long as the characters match, we continue dequeueing, popping, and comparing each character until our containers are empty (a non-match means s isn't a palindrome).

Write the following declarations and implementations:

Two instance variables: one for your stack, and one for your queue

A void pushCharacter(char ch) method that pushes a character onto a stack.

A void enqueueCharacter(char ch) method that enqueues a character in the queue instance variable.

A char popCharacter() method that pops and returns the character at the top of the stack instance variable.

A char dequeueCharacter() method that dequeues and returns the first character in the queue instance variable.

**Input Format**

You do not need to read anything from stdin. The locked stub code in your editor reads a single line containing string s. It then calls the methods specified above to pass each character to your instance variables.

**Output Format**

You are not responsible for printing any output to stdout.

If your code is correctly written and s is a palindrome, the locked stub code will print The word s, is a palindrome; otherwise, it will print The word s, is not a palindrome

**Constraints**

S is composed of lowercase English letters.

**Sample Input 1**

Racecar

**Sample Output 1**

The word, Racecar, is not a palindrome.

import java.io.\*;

import java.util.\*;

class MyClass {

// Write your code here.

Queue<Character> queue;

Stack<Character> stack;

MyClass(){

this.queue = new LinkedList<Character>();

this.stack = new Stack<Character>();

}

void pushCharacter(char ch){

this.stack.push(ch);

}

void enqueueCharacter(char ch){

this.queue.add(ch);

}

char popCharacter(){

return this.stack.pop();

}

char dequeueCharacter(){

return this.queue.remove();

}

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

String input = scan.nextLine();

scan.close();

// Convert input String to an array of characters:

char[] s = input.toCharArray();

// Create a Solution object:

MyClass p = new MyClass();

// Enqueue/Push all chars to their respective data structures:

for (char c : s) {

p.pushCharacter(c);

p.enqueueCharacter(c);

}

// Pop/Dequeue the chars at the head of both data structures and compare them:

boolean isPalindrome = true;

for (int i = 0; i < s.length/2; i++) {

if (p.popCharacter() != p.dequeueCharacter()) {

isPalindrome = false;

break;

}

}

//Finally, print whether string s is palindrome or not.

System.out.println( "The word, " + input + ", is "

+ ( (!isPalindrome) ? "not a palindrome." : "a palindrome." ) );

}

}

**19.Task:Collection Interface - Sort an ArrayList<Integer> in descending chronological order.**

Get ‘N’ Integers from the user and store it in ArrayList class object, and sort the elements in descending chronological order and print it.

**Input Format:**

* The first Line ‘N’ represents number of elements in the ArrayList.
* The next N-Lines represents values of the ArrayList object.

**Output Format:**

* The sorted ArrayList

**Sample Input :**

8

5

-23

12

98

34

78

0

92

**Sample Output :**

[98, 92, 78, 34, 12, 5, 0, -23]

**Constraints**

0<N<=100

**Sample Input 1**

6

1

4

78

-2

78

0

**Sample Output 1**

[78, 78, 4, 1, 0, -2]

import java.util.\*;

class Main

{

public static void main(String args[])

{

int N;

Scanner ip=new Scanner(System.in);

N=ip.nextInt();

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

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

{

num.add(ip.nextInt());

}

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

System.out.println(num);

}

}