18-07-22

Java History:

=============

-> In 1990, Sun Microsystems Inc. (US) has conceived a project to develop software for consumer

electronic devices that could be controlled by a remote. This project was called Green Project.

-> In January of 1991, James Gosling, Bill Joy, Patrick Naughton and several others started to

develop.

-> Gosling thought C and C++ could be used to develop the project. But the problem he

identified that these languages are system dependent and hence could not be used on various processors.

-> So he started to developing a new language, which was completely system independent.

-> This Language was initially called Oak. Later it was changed to Java.

James Gosling is considered as “FATHER OF JAVA”.

The main concept of java is “Write Once, Run Anywhere”.

Java Version History:

====================

JDK Alpha and Beta (1995) was the first releases but they have highly unstable APIs....

JDK 1.0 (January 23, 1996) ...

JDK 1.1 (February 19, 1997) ...

J2SE 1.2 (December 8, 1998) ...

J2SE 1.3 (May 8, 2000) ...

J2SE 1.4 (February 6, 2002) ...

J2SE 5.0 (September 30, 2004) ...

Java SE 6 (December 11, 2006) .......

……………

Java SE 14 (March 17, 2020).

Java SE 17 (Sept 14th , 2021).

Java SE 18 (March 22, 2022)

------------------------------------------------------------------------------

Java Features/ Buzzwords:

-------------------------

1. Simple

2. Platform Independent

3. Architecuter Neutral

4. Portable

5. Compiled and Interpreted

6. Object Oriented

7. Secure

8. Robust

9. Multi Threading

10.Distriubuted

11.High Performance

12.Dynamic

----------------------------------------------------------------------------

19-07-22

========

Java Fundamentals:

------------------

1. Identifiers

2. Keywords/Reserved Words

3. Data Types

4. Variables

5. Commandline Arguments

6. Operators

7. Control Statements

8. Arrays

9. Variable Length Arguments

10.Java Coding Standards

25-07-22:

=========

1. Identifier:

--------------

Identifier is a name that used in the programming. It may be a class name, variable name, interface name,etc...

Rules for define a identifier:

-----------------------------

1. The following are the allowed characters to define an identifier

A - Z

a - z

0 - 9

\_ and $

2. An identifier should be start with and Alphabet and \_

3. An identifier in java is a case Sensitive.

Ex: Marks and MARKS are two different identifiers

4. There is no limit for length of an identifier but it is recommended to take

the length of an identifier is upto 15 characters only.

5. It is not recommended to use reserved words as an identifiers.

6. It is allowed to define all the predefined classes as identifier but it is not recommended.

int Runnable=10;

Ex:

Total\_Marks

Total$Marks

Salary1

1emp

\_x

\_\_x

int

Thread

Min#marks

------------------------------------------------------------------------

2. Reserved Word /Keywords:

----------------------------

The words which are having a specific meaning or assiciated with some specific operation are called Reserved words.

-> Java is a small programming language it contains total of 53 keywords.

-> int,float, double, short, byte, char, long,boolean

-> private, public, protoected, default

-> if, else, switch, for, while, do

-> class, interface, abstract

-> this, static, super, new, instanceof,final

-> try, catch, throw, throws, finally etc...

---------------------------------------------------------------------------

3. Datatype:

------------

Data type is used to specify what type of values it can be stored in a variable.

It can be classified into two types

1. Premitive or Fundamental datatypes

The data types which are given as part of java software installation are called premitive data types.

Ex:

byte, short, int, long

float, double

char

boolean [8 primitive data types]

2. User defined Datatypes

The datatypes which are created by the user by using fundamental datatypes are called user defined datatypes.

Ex:

class, Array, String etc...

datatype size range Wrapper class default\_value

-----------------------------------------------------------------------------

byte 8 bits -128 to +127 Byte 0

short 2 bytes -32768 to +32767 Short 0

int 4 bytes -2^31 to +2^31 -1 Integer 0

long 8 bytes -2^63 to +2^63 -1 Long 0

float 4 bytes -3.4 x e^38 to Float 0.0

+3.4 x e^38

double 8 bytes -1.7 x e^308 to Double 0.0

+1.7 x e^308

char 2 bytes 0 to 65535 Character space

boolean 1 bit true,false Boolean false

---------------------------------------------------------------------------

4. variable:

-----------

A variable is a name of the memory location where the actual value was stored.

Types of Variables:

-------------------

1. Primitive datatype variables

The variables which are used to store the primitive data type values.

int x=10;

x is a primitive datatype variable

2. reference varaibles

The variables which are used to store the address of an object.

Student s1=new Student();

s1 -> reference variable

Variables are classified into:

1. Instance variable

-> The variables which are declared inside a class and outside of a

method or constructor are called instance variables.

-> Every object will maintain a separate copy of memory for these

variables.

-> Instance variables are also called as object dependent variables.

Ex:

class Sample

{

int x;

String name;

public void m1()

{

.....

}

.....

}

Here x and name are called instance variables.

2. static variable

-> A variable which is declared inside a class and outside

of a method or constructor with a keyword "static" are called static variables.

-> Static variables are also called as class variables.

-> If you want to access static variable we have to use

class name.

Ex:

classname.varaible;

-> A single copy of memory is allocated for static variables

and allows to access this by the all the objects.

Ex:

class Sample

{

int x;

static int y;

public void m1()

{

.......

}

......

}

Note: Here x is instance variable and y is static variable.

3. local variable

-> A variable which is declared inside a method or block are

called local variables.

-> The scope of local variables is within the block only.

------------------------------------------------------------------------------------------

Literal:

========

A constant which is assigned to a variable is called literal.

Types of Literals:

------------------

1) Integeral Literals:

----------------------

Integeral literals can be represented in the following forms.

i. Decimal form

int x=10;

ii. Octal form

int y=010;

iii.Hexa Decimal form

int z=0x10

2) character literals:

---------------------

A character is enclosed with single quotes is called character literals.

Ex:

char ch='a';

char ch='A';

char ch='#';

3) floating value literals:

---------------------------

1. float:

float f1=3.142f;

2. double:

double d1=3.142;

Here any real value ends with f treated as float otherwise it is treated as double.

4) String Literal:

------------------

A group of characters enclosed with double quotes is called string literal.

Ex:

String msg="Welcome to Java";

5) Boolean Literal:

-------------------

boolean literal are used to represent a condition is satisfied or not. It includes 2 values (true and false)

Ex:

boolean flag=true;

-------------------------------------------------------------------------------------

Java Program Structure:

==================

1. Documentation section [Optional]

2. Package declaration section [Optional]

3. package importing section [Optional]

4. interface declaration [Optional]

5. class declaration [Optional]

6. class with main method [Mandatory]

------------------------------------------------------------------

1. //program description like name.

2. package packagename;

3. import packagename.\*;

4. interface <interface\_name>

{

........

}

5. class <class\_name>

{

.......

}

6. class Mainclass

{

public static void main(String args[])

{

........

}

}

----------------------------------------------------------------------------------------------

Commandline Arguments:

---------------------

-> The Arguements which are passed to the program at the time of program execution through command

prompt are called Commandline arguments.

-> Commandline arguments are recommended when you know the input to your program in an advance.

1. Program to find the addition of two numbers which are taken as commandline arguments.

class Sample

{

public static void main(String args[])

{

System.out.println(args[0]+args[1]);

}

}

Run:

java Sample 10 20

Output:

--------

1020

--------------------------------------------------------------------------------

class Sample

{

public static void main(String args[])

{

System.out.println(Integer.parseInt(args[0])+Integer.parseInt(args[1]));

}

}

Run:

java Sample 10 20

Output:

--------

30

----------------------------------------------------------------------------------

2. Write a program to print multiplication table by taking n and m values as commandline arguments.

class Sample

{

public static void main(String args[])

{

int a=Integer.parseInt(args[0]);

int b=Integer.parseInt(args[1]);

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

{

System.out.println(a+"\*"+i+"= "+a\*i);

}

}

}

-------------------------------------------------------------------

3. Program to display your personal information which are taken by commandline arguments.

class Sample

{

public static void main(String args[])

{

System.out.println("Name :"+args[0]);

System.out.println("Roll NO:"+args[1]);

System.out.println("Age: "+args[2]);

System.out.println("Branch :"+args[3]);

System.out.println("Section :"+args[4]);

}

}

---------------------------------------------------------

4. Program to find the Simple Interest by taking the input through commandline arguments.

class SimpleInterest

{

public static void main(String args[])

{

int P,T;

float R,I;

P=Integer.parseInt(args[0]);

T=Integer.parseInt(args[1]);

R=Float.parseFloat(args[2]);

I=(P\*T\*R)/100;

System.out.println("Principle Amount="+P);

System.out.println("Time in Months="+T);

System.out.println("Rate of Interest="+R);

System.out.println("Interest = "+I);

}

}

----------------------------------------------------------------

5. Program to find the sum of all the elements which are passed through commandline arguments.

class Demo2

{

public static void main(String args[])

{

System.out.println(args.length);

int sum=0;

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

sum=sum+Integer.parseInt(args[i]);

System.out.println("Sum = "+sum);

}

}

------------------------------------------------------------------

6. Program to find the capacity of a Disk. A Disk has 2 surfaces.

Each surface divided into t tracks.

Each track is divided into s sectors.

Each sector is divided into b blocks.

Each block is of 512 bytes of size.

Now print the capacity of the disk in KB, MB and GB format.

class Demo3

{

public static void main(String args[])

{

int t,s,b,C;

t=Integer.parseInt(args[0]);

s=Integer.parseInt(args[1]);

b=Integer.parseInt(args[2]);

C=2\*t\*s\*b\*512;

System.out.println(C+" Bytes");

System.out.println(C/1024+" KB");

System.out.println(C/(1024\*1024)+" MB");

System.out.println((float)C/(1024\*1024\*1024)+" GB");

}

}

--------------------------------------------------------------------------

7. Program to print all the even numbers from the input taken through commandline arguments.

class Demo4

{

public static void main(String args[])

{

int temp;

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

{

temp=Integer.parseInt(args[i]);

if(temp%2==0)

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

}

}

}

8. Program to find the count number of even and odd elements are there the input which is taken through commandline arguments.

class Demo4

{

public static void main(String args[])

{

int temp,count=0;

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

{

temp=Integer.parseInt(args[i]);

if(temp%2==0)

{

count++;

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

}

}

System.out.println("The number of even numbers: "+count);

System.out.println("The number of odd numbers: "+(args.length-count));

}

}

----------------------------------------------------------------------------------

9. Write a program to read the Student Marks as commnadline arguments and find the total marks and average marks.

class Total\_Marks

{

public static void main(String args[])

{

String rollno,name;

int m1,m2,m3,m4,m5,m6,tot;

float avg;

rollno=args[0];

name=args[1];

m1=Integer.parseInt(args[2]);

m2=Integer.parseInt(args[3]);

m3=Integer.parseInt(args[4]);

m4=Integer.parseInt(args[5]);

m5=Integer.parseInt(args[6]);

m6=Integer.parseInt(args[7]);

tot=m1+m2+m3+m4+m5+m6;

avg=tot/6;

System.out.println("Roll Number ="+rollno);

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

System.out.println("Total Marks ="+tot);

System.out.println("Average = "+avg);

}

}

-------------------------------------------------------------------------------------------

27-07-22

========

Scanner:

-------

1. Scanner is a class which is available in java.util package.

2. It is used to read the input from the user during program execution.

How to import Scanner class in your program.

import java.util.Scanner;

How to create an object for Scanner class.

Scanner objname=new Scanner(System.in);

Methods:

--------

1. nextByte() - To read a byte value

2. nextShort() - To read a short value

3. nextInt() - To read an integer value

4. nextLong() - To read a long value

5. nextFloat() - To read a float value

6. nextDouble() - To read a double value

7. next() - To read a word

8. nextLine() - To read a line of text [group of words]

9. nextBoolean() - To read a boolean value

10.next().charAt(0) - To read a character

------------------------------------------------------

1. Program to read input of an employee using Scanner class.

import java.util.Scanner;

class ScannerDemo1

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int empid;

String ename,desg;

float salary;

char gender;

long mobileno;

System.out.println("Enter empid, name,desg,salary,gender, mobileno");

empid=sc.nextInt();

sc.nextLine();

ename=sc.nextLine();

desg=sc.nextLine();

salary=sc.nextFloat();

gender=sc.next().charAt(0);

mobileno=sc.nextLong();

System.out.println(empid+" "+ename+" "+desg+" "+salary+" "+gender+" "+mobileno);

}

}

----------------------------------------------------------------------------------------

import java.util.Scanner;

class ScannerDemo2

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int age;

String name;

age=sc.nextInt();

name=sc.next();

System.out.println("Hello "+name+" You are "+age+" years old");

}

}

Output:

D:\DR>java ScannerDemo2

32

Rajesh Naidu

Hello Rajesh You are 32 years old

------------------------------------------------------------

import java.util.Scanner;

class ScannerDemo2

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int age;

String name;

age=sc.nextInt();

name=sc.nextLine();

System.out.println("Hello "+name+" You are "+age+" years old");

}

}

Output:

D:\DR>java ScannerDemo2

32

Hello You are 32 years old

-----------------------------------------------------------

import java.util.Scanner;

class ScannerDemo2

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int age;

String name;

name=sc.nextLine();

age=sc.nextInt();

System.out.println("Hello "+name+" You are "+age+" years old");

}

}

Output:

-------

D:\DR>java ScannerDemo2

Rajesh Naidu

32

Hello Rajesh Naidu You are 32 years old

-----------------------------------------------------------------

Operator:

-> It is a symbol used to perform an operation. Every expression it includes operators and operands.

Types of Operators:

------------------

1. Arithmetic Operators [+, -, \*, /, %]

2. Relative Operators [<, <=, >, >=, ==, !=]

3. Logical Operators [&&, ||, !]

4. Bitwise Operators [&, |, ~, <<, >>]

5. Assignment Operators [=, +=, -=,.....]

6. Conditional Operator [?:]

7. Special Operators [new, instanceof]

Logical Operators:

------------------

These are the operators used to connect two or more expressions.

&& -> Both the exrpessions are true then only it returns true.

|| -> Any one of the expression is true then it returns true.

! -> It returns exactly opposite to the given condition

class OperatorsDemo

{

public static void main(String args[])

{

System.out.println(true && true); //true

System.out.println(false && true); //false

System.out.println(false && false); //false

}

}

--------------------------------------------------------------

class OperatorsDemo

{

public static void main(String args[])

{

System.out.println(true || true); //true

System.out.println(false || true); //false

System.out.println(false || false); //false

}

}

--------------------------------------------------

class OperatorsDemo

{

public static void main(String args[])

{

System.out.println(4 || 5);

System.out.println(4 && 5);

System.out.println(!4);

}

}

OperatorsDemo1.java:5: error: bad operand types for binary operator '||'

System.out.println(4 || 5); //true

Note: Logical Operators will take boolean values only as an arguments.

---------------------------------------------------------------------------------------------

class OperatorDemo2

{

public static void main(String args[])

{

int x=10,y=20;

if(++x<=10 && y++<30)

System.out.println(x+" "+y);

else

System.out.println(x+" "+y);

}

}

Output:

11 20

Note: An expressions contains logical and (&&) then if first expression is evaluated as false then java interpreter will not evaluate the remaining expressions.

---------------------------------------------------------------------------------------------

class OperatorDemo2

{

public static void main(String args[])

{

int x=10,y=20;

if(++x<=10 || y++<30)

System.out.println(x+" "+y);

else

System.out.println(x+" "+y);

}

}

Output:

11 21

-------------------------------------------------------------------------------------------------

class OperatorDemo2

{

public static void main(String args[])

{

int x=10,y=20;

if(x++<=10 || y++<20)

System.out.println(x+" "+y);

else

System.out.println(x+" "+y);

}

}

Output:

------

11 20

-----------------------------------------------------------------------------------------

Bitwise Operators:

Bitwise Operators are used to perform bit by bit manipulation.

& -> It returns true if both the arguments are true

| -> It returns true if any of the argument is true

^ -> It returns true both the arguments are different.

~ -> It change the bits 1 -> 0 or 0 -> 1.

class OperatorsDemo

{

public static void main(String args[])

{

System.out.println(true & true); //true

System.out.println(false | true); //true

System.out.println(true ^ false); //true

System.out.println(4 & 5); //4

System.out.println(4 | 5); //5

System.out.println(4 ^ 5); //1

}

}

--------------------------------------------------------------------

class OperatorsDemo

{

public static void main(String args[])

{

int x=10,y=20;

if(x++>10 & y++<=20)

x++;

else

y++;

System.out.println(x+" "+y);

}

}

Output:

11 22

-------------------------------------------------------------

Differences between Logical and Bitwise operators

Logical Operator(&&, ||) Bitwise Operators(&,|,^)

1. Second Argument evaluation 1. Both arguments evaluation is

is Optional. Mandatory.

2. Performance is good 2. Performance is low

3. It will work on Boolean 3. It will work on boolean and

arguments. integeral values.

int x= 4, y= 5;

0100 0100 0100

0101 0101 0101

---- ---- ----

&-> 0100 |-> 0101 ^-> 0001

---- ---- ----

4 5 1

---- ---- ----

4 -> 0100 4-> 0100

4<<1-> 1000 =>8 4>>1-> 0010 => 2

x<<n or x>>n

1. Write a program to find the given number is even or odd.

5 -> 0000 0101

6 -> 0000 0110

9 -> 0000 1001

12-> 0000 1100

6 -> 0000 0110

1 -> 0000 0001

& ---------

0000 0000

9 -> 0000 1001

1 -> 0000 0001

& ---------

0000 0001

int n=sc.nextInt();

if(n & 1 >0)

System.out.println("odd");

else

System.out.println("Even");

----------------------------------------------------------------

BitManpulation:

---------------

BitManpulation is mainly used in

1. Encryption Algorithm

2. Data Compression

3. Error checking

4. Optimization

a^0 => a

a=4 -> 0100

0 -> 0000

^ ----

0100 = 4 [a]

a^0 => a

-------------------------------

a^a => 0

-------------------------------

a^b = k

a^b = k

a^b^b = k^b

a = k^b

then b^k = a

--------------------------------------------------------

Q. Given an integer, check if ith bit of the integer is set or unset.

Input:

n, ith

n=9 => 0000 1001

0000 0100

---------

0000 0000 = 0 indicates the bit is unset

n=13 =>0000 1101

& 0000 0100

---------

0000 0100 >0 >0 means the bit is set.

i= 2

1 => 0000 0001

1<<1 0000 0010

1<<2 0000 0100

1<<3 0000 1000

if((n & 1<<i)>0)

System.out.println("Bit is set bit");

else

System.out.println("Bit is unset");

--------------------------------------------------------

Q. Given an array, find the xor of all the numbers where ith bit is set.

-------------------------------------------------------

Q. Given an array, all the numbers appear even number of times except one.

Ex: [2, 1, 3, 2, 1, 1, 1, 4, 4]

1st Approach:

------------

Sort(arr)

1 1 1 1 2 2 3 4 4

--- --- ---

Time Complexity: O(nlogn)

Space Complexity: O(1)

2nd Approach:

-------------

HashMap hs;

hs

Key Value

1 4

2 2

3 1

4 2

3rd approach:

class Demo8

{

public static void main(String args[])

{

int x[]={1,2,3,4,1,2,1,1,4};

int result=0;

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

{

result=result^x[i];

}

System.out.println(result);

}

}

-------------------------------------------------------

Q. Given an array,

all the numbers appear even number of times except 2 numbers.

[4, 1, 2, 3, 2, 1, 5, 4]

----------------------------------------------------------------------------

Operators Precedence:

--------------------

In an expression more than one operator is there then which operator is executed first decided by the precedence.

x=4+8/2

x=8

Operator Associativity:

-----------------------

In an expression more than one operator having equla precedence then the order in which is executed decided by associativity. [left -> right or right -> left].

x=y=z=7;

Note: We can identify the precedence and Associativity by using the following rules.

For Precedence: PUMA IS REBL

For Associativity: All operators are left -> right except Unary, Conditional and Assignment.

-------------------------------------------------------------

Control Statements:

-------------------

Control Statements are used to control the flow of execution of a program.

Control Statements are classified into 2 types

1) Conditional Statements

if, if else, if else if, nested if, switch

2) Looping or iterative statements:

for, while, do while

Unconditional Statments:

------------------------

break, continue

Conditional Statements:

-----------------------

1) if

if(condition)

{

statements;

}

2) if else

if(condition)

{

//true block

}

else

{

//false block

}

3) if else if:

If we want to check multiple conditions on the same variable then it is preferable to use if else if.

if(condition1)

{

//block of statements

}

else if(condition2)

{

//block of statements

}

else if(condition3)

{

//block of statements

}

.....

else

{

//block of statements

}

4) Nestted if:

if statements placed in another if statements is called nested if.

if(condition1)

{

if(condition2)

{

//block of statements

}

else

{

//block of statements

}

}

else

{

..........

}

5) switch:

---------

switch(variable)

{

case label1:

statements;

break;

case label2:

statements;

break;

.......

default:

statemetns;

break;

}

------------------------------------------------------------

int x=5;

int y=5;

if(y=x) => if(5)

System.out.println("Y is equal to x");

else

System.out.println("y is not equal to x");

Output:

------

Error: Incompatable types int cannot be converted to boolean

2)

class demo9

{

public static void main(String args[])

{

int x=10;

if(x==10)

System.out.println("Welcome");

System.out.println("Hello");

else

System.out.println("Good Bye");

}

}

Output:

demo9.java:9: error: 'else' without 'if'

else

^

--------------------------------------------------------------------------

3)

class demo9

{

public static void main(String args[])

{

int x=10;

if(x>7)

{

System.out.println("Welcome");

break;

System.out.println("Hello");

}

else

System.out.println("Good Bye");

}

}

output:

demo9.java:9: error: break outside switch or loop

break;

^

------------------------------------------------------

switch:

1) What type of datatype variables will take by switch:

1) byte, short ,int, String(1.7Version),char

2) long, float, double, boolean datatypes are not allowed to taken as switch

variables.

3) In switch case labels are allowed to take an expression but the expression

must be evaluated first that will be used as case label.

4) Duplicate case labels are not allowed.

5)

int x=5;

switch(x)

{

case 1:

System.out.println("hello");

break;

case 2:

System.out.println("Welcome");

break;

case 1+2:

System.out.println("Good Morning");

break;

case 5\*1:

System.out.println("Good Byee");

break;

default:

System.out.println("Guest");

break;

}

----------------------------------------------------

class demo9

{

public static void main(String args[])

{

int x=5;

switch(x)

{

case 1:

System.out.println("hello");

break;

case 2:

System.out.println("Welcome");

break;

case 3+2:

System.out.println("Good Morning");

break;

case 5\*1:

System.out.println("Good Byee");

break;

default:

System.out.println("Guest");

break;

}

}

}

Output:

demo9.java:17: error: duplicate case label

case 5\*1:

^

1 error

-----------------------------------------------------------------------

class demo9

{

public static void main(String args[])

{

int x=5,y=2;

switch(x)

{

case x>3:

System.out.println("Ajay");

break;

case y<x:

System.out.println("Sivaram");

break;

}

}

}

demo9.java:8: error: incompatible types: boolean cannot be converted to int

case x>3:

^

demo9.java:11: error: incompatible types: boolean cannot be converted to int

case y<x:

^

2 errors

------------------------------------------------------------

class demo9

{

public static void main(String args[])

{

char ch='B';

int x=65;

switch(x)

{

case 'A':

System.out.println("Ajay");

break;

case 'B':

System.out.println("Sivaram");

break;

default:

System.out.println("Govind");

break;

}

}

}

Output:

Ajay

------------------------------------------------------------------

class demo9

{

public static void main(String args[])

{

char ch='B';

int x=65;

switch(x)

{

default:

System.out.println("Govind");

break;

case 'A':

System.out.println("Ajay");

case 'B':

System.out.println("Sivaram");

//break;

case 'C':

System.out.println("Kranthi");

//break;

}

}

}

D:\DR>java demo9

Ajay

Sivaram

Kranthi

---------------------------------------------

05-08-22

=======

1) How to read the input

Scanner sc=new Scanner(System.in);

Method:

public void nextXXXX();

2) Type Casting:

Convertion from one data type to another datatype

Implicit Conversion:

It is also called as Automatic Conversion. Without user intervention the convertion will be done automatically.

Conversion from lower datatype to higher datatype is called Implicit. There is no data loss.

byte->short->char->int->long->float->double

Example:

int x=10;

float y=x; => y=10.0f;

Explicit:

Conversion from higher datatype to lower datatype is called Explicit conversion.

double -> float -> long -> int -> char -> byte

float y=3.142f;

int x=y; //Compilation Error: Lossy precession value

Syntax:

lowerdatatype variable=(datatype)variablename;

Ex:

float y=3.142f;

int x=(int)y;

------------------------------------------

3) How to print formatted output.

System.out.format("formatted String",variables...);

class Demo10

{

public static void main(String args[])

{

int x=65;

char ch='A';

float y=3.1426f;

String str="JAVA PROGRAMMING";

System.out.format("%d %c %05.2f %s",x,ch,y,str);

}

}

========================================================================================

CONTROL STATEMENTS

========================================================================================

--------------------------------------------------------------------------------

SELECTIVE STATEMENTS

--------------------------------------------------------------------------------

1. Program to find the biggest of two numbers:

Input:

10

20

Ouput:

20

------------------------------------------------------------------------------

2. Program to find the eligibility for voting by taking age as an input.

Input:

25

Output:

Eligible

import java.util.Scanner;

class Voting

{

public static void main(String args[])

{

int age;

Scanner sc=new Scanner(System.in);

age=sc.nextInt();

if(age>=18)

System.out.println("Eligible");

else

System.out.println("Not Eligible");

sc.close();

}

}

------------------------------------------------------------------------------

3. Program to find the given character is vowel or consonant?

Input:

A

Output:

Vowel

Input:

S

Output:

Consonant

import java.util.Scanner;

class VowelorNot

{

public static void main(String args[])

{

char ch;

Scanner sc=new Scanner(System.in);

ch=sc.nextLine().charAt(0);

if(ch=='a')

System.out.println("Vowel");

else if(ch=='e')

System.out.println("Vowel");

else if(ch=='i')

System.out.println("Vowel");

else if(ch=='o')

System.out.println("Vowel");

else if(ch=='u')

System.out.printn("Vowel");

else

sc.close();

}

}

2nd Approach:

class VowelorNot

{

public static void main(String args[])

{

char ch;

Scanner sc=new Scanner(System.in);

ch=sc.nextLine().charAt(0);

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

System.out.println("Vowel");

else

System.out.println("Consonent");

sc.close();

}

}

3rd approach -using switch case:

class VowelorNot

{

public static void main(String args[])

{

char ch;

Scanner sc=new Scanner(System.in);

ch=sc.nextLine().charAt(0);

switch(ch)

{

case 'a':

case 'e':

case 'i':

case 'o':

case 'u':

System.out.println("Vowel");

break;

default:

System.oout.println("Consonant");

break;

}

sc.close();

}

}

---------------------------------------------------------------------------------

4. Program to input sides of a triangle and check whether triangle is valid or not.

Input:

2

3

4

Output:

Valid

Hint:

A triangle is valid if sum of its two sides is greater than the third side. If three sides are a, b and c, then three conditions should be met.

1. a + b > c

2. a + c > b

3. b + c > a

class ValidTriangleorNot

{

public static void main(String args[])

{

int a,b,c;

Scanner sc=new Scanner(System.in);

a=sc.nextInt();

b=sc.nextInt();

c=sc.nextInt();

if(a+b>c && a+c>b && b+c>a)

System.out.println("Valid triangle");

else

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

sc.close();

}

}

-------------------------------------------------------------------------------

5. Program to check whether a triangle is equilatoral, Isosceles or Scalence.

Input:

2

3

4

Output:

Scalence

Hint:

Equilatoral -> All the sides of the Triangle are equal

Isosceles -> Any two sides of the Triangle are equal

Scalence -> All the sides of the Triangle are different

class TraingleDemo

{

public static void main(String args[])

{

int a,b,c;

Scanner sc=new Scanner(System.in);

a=sc.nextInt();

b=sc.nextInt();

c=sc.nextInt();

if(a==b && b==c)

System.out.println("Equilatoral");

else if(a==b || b==c || a==c)

System.out.println("Isosceles");

else

System.out.println("Scalence")

sc.close();

}

}

--------------------------------------------------------------------------------

6. Write a program find who is best hero based on given input?

Input: Chiru

67 52 27

Balayya

45 33 26

Note: 1 hit movie= 10 points, 1 average movie=5, a flop movie= -5

Input format is: Hero\_Name Hit\_Movies Avg\_Movies Flop\_Movies.

Source Code:

============

import java.util.Scanner;

class BestHero

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int h1,a1,f1,h2,a2,f2,t1,t2;

String hero1,hero2;

hero1=sc.nextLine();

h1=sc.nextInt();

a1=sc.nextInt();

f1=sc.nextInt();

sc.nextLine();

hero2=sc.nextLine();

h2=sc.nextInt();

a2=sc.nextInt();

f2=sc.nextInt();

t1=h1\*10+a1\*5+f1\*(-5);

t2=h2\*10+a2\*5+f2\*(-5);

if(t1>t2)

System.out.println(hero1+" is the best hero with "+t1+" points");

else

System.out.println(hero2+" is the best hero with "+t2+" points");

sc.close();

}

}

---------------------------------------------------------------------------------

7. Write a program to display the given digit(0 to 9) in words as follows

Input: 9

Output: Nine

--------------------------------------------------------------------------------

8. Write a Program to print the color name by taking the Color code as input.

V -> Violet

I -> Indigo

B -> Blue

G -> Green

Y -> Yellow

O -> Orange

R -> Red

Input:

G

Output:

Green

Note: Implement by using both if else if and switch case.

import java.util.Scanner;

class Sample

{

public static void main(String args[])

{

char code;

Scanner sc=new Scanner(System.in);

code=sc.nextLine().charAt(0);

code=Character.toUpperCase(code);

switch(code)

{

case 'V':

System.out.println("Violet");

break;

case 'I':

System.out.println("Indigo");

break;

case 'B':

System.out.println("Blue");

break;

case 'G':

System.out.println("Green");

break;

case 'Y':

System.out.println("Yellow");

break;

case 'O':

System.out.println("Orange");

break;

case 'R':

System.out.println("Red");

break;

default:

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

break;

}

}

}

-------------------------------------------------------------------------------

9. Write a program to read temperature in centigrade and display a suitable message according to temperature state below:

Temp < 0 then Freezing weather

Temp 0-10 then Very Cold weather

Temp 10-20 then Cold weather

Temp 20-30 then Normal in Temp

Temp 30-40 then Its Hot

Temp >=40 then Its Very Hot

import java.util.Scanner;

class Temprature

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int temp;

temp=sc.nextInt();

if(temp<0)

System.out.println("Freezing Weather");

else if(temp<10)

System.out.println("Very Cold Weather");

else if(temp<20)

System.out.println("Cold Weather");

else if(temp<30)

System.out.println("Normal in Temp");

else if(temp<40)

System.out.println("Its Hot");

else

System.out.println("Its Very Hot");

sc.close();

}

}

--------------------------------------------------------------------------------

10. Write a program to display season by taking the input as month.

Assume:

4, 5, 6                - Summer

        7, 8, 9,10          - Rainy

        11, 12, 1            - Winter

        2, 3                    - Spring

Input: 9

Output: Rainy

Source Code:

============

import java.util.Scanner;

class Sample

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int m;

m=sc.nextInt();

switch(m)

{

case 4:

case 5:

case 6:

System.out.println("Summer");

break;

case 7:

case 8:

case 9:

case 10:

System.out.println("Rainy");

break;

case 11:

case 12:

case 1:

System.out.println("Winter");

break;

case 2:

case 3:

System.out.println("Spring");

break;

default:

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

break;

}

}

}

-------------------------------------------------------------------------------

11. Program to find the roots of the quadratic equation

class Roots

{

public static void main(String[] args) {

// value a, b, and c

double a = 2.3, b = 4, c = 5.6;

double root1, root2;

// calculate the determinant (b2 - 4ac)

double determinant = b \* b - 4 \* a \* c;

// check if determinant is greater than 0

if (determinant > 0) {

// two real and distinct roots

root1 = (-b + Math.sqrt(determinant)) / (2 \* a);

root2 = (-b - Math.sqrt(determinant)) / (2 \* a);

System.out.format("root1 = %.2f and root2 = %.2f", root1, root2);

}

// check if determinant is equal to 0

else if (determinant == 0) {

// two real and equal roots

// determinant is equal to 0

// so -b + 0 == -b

root1 = root2 = -b / (2 \* a);

System.out.format("root1 = root2 = %.2f;", root1);

}

// if determinant is less than zero

else {

// roots are complex number and distinct

double real = -b / (2 \* a);

double imaginary = Math.sqrt(-determinant) / (2 \* a);

System.out.format("root1 = %.2f+%.2fi", real, imaginary);

System.out.format("\nroot2 = %.2f-%.2fi", real, imaginary);

}

}

}

-------------------------------------------------------------------------------

12. Program to input basic salary of an employee and calculate its gross salary according to following.

Basic Salary <=10000 ==> HRA=20%, DA=80%

Basic Salary <=20000 ==> HRA=25%, DA=90%

Basic Salary >20000 ==> HRA=30%, DA=95%

-------------------------------------------------------------------------------

13. Program to input electricity unit charges and calculate total electricity bill according to the given condition:

for first 50 units Rs - 0.50/unit

for next 100 units Rs - 0.75/unit

for next 100 units Rs - 1.20/unit

for unit above 250 Rs - 1.50/unit

An additional surcharge of 20% is added to the bill.

Input:

150

Output:

120

import java.util.Scanner;

class ElectricityBill

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int units;

float bill=0.0f;

units=sc.nextInt();

if(units<=50)

{

bill=units\*0.50f;

}

else if(units>50 && units<=150)

{

bill=50\*0.50f + (units-50)\*0.75f;

}

else if(units<=250)

{

bill=50\*0.50f + 100\*0.75f + (units-150)\*1.20f;

}

else

{

bill=50\*0.50f + 100\*0.75f + 100\*1.20f + (units-250)\*1.50f;

}

bill=bill+ 0.2f \* bill;

System.out.println("Total Bill = "+bill);

System.out.format("%.2f",bill);

sc.close();

}

}

-----------------------------------------------------------------------------

14. Program to implement simple calculator by using switch case.

import java.util.Scanner;

class SimpleCalculator

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int x,y;

char op;

x=sc.nextInt();

y=sc.nextInt();

sc.nextLine();

System.out.println("+ - Addition\n- - Subtraction\n/ - Division\n\* - Multiplication\n% - Modulous\n");

System.out.println("Enter your choice");

op=sc.nextLine().charAt(0);

switch(op)

{

case '+':

System.out.println(x+y);

break;

case '-':

System.out.println(x-y);

break;

case '\*':

System.out.println(x\*y);

break;

case '/':

System.out.println(x/y);

break;

case '%':

System.out.println(x%y);

break;

default:

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

break;

}

}

}

--------------------------------------------------------------------------------

---------------------------------------------------------------------------------

LOOPING STATEMENTS

---------------------------------------------------------------------------------

Looping Statements:

These are the statements used to execute a block of statements repeatedly for some number of times.

i. for

ii. while

iii.do while

2 types

1. Entry controlled loops

for, while

2. Exit Controlled loops

do while

i) for loop:

for(initialization;condition;updatation)

{

//body of the loop

}

Note: The loop is to be executed until the condtion is false.

BASICS:

======

---------------------------------------------------------------------------------

1. Program to print your name 10 times using for loop?

----------------------------------------------------------------------------------

2. Program to print numbers from 1 to 100.

----------------------------------------------------------------------------------

3. Program to print all the even numbers from 1 to 100.

-----------------------------------------------------------------------------------

4. Program to print all the odd numbers from 1 to n.

-----------------------------------------------------------------------------------

5. Program to print all 4 multiple from 1 to n.

import java.util.\*;

class multiplication{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

for(int i=4;i<=n;i=i+4)

{

System.out.println(i);

}

sc.close();

}

}

----------------------------------------------------------------------------------

6. Program to find sum of n natural numbers.

Input:

5

Output: 15

Explanation: sum=1 + 2 + 3 + 4 + 5 => 15

class SumofNumbers{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int sum=0;

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

{

if(i%2==0)

sum=sum+i;

}

System.out.println(sum);

sc.close();

}

}

-----------------------------------------------------------------------------------

7. Program to find sum of squares n natural numbers.

n=5

sum=1^2 + 2^2 + 3^2 + 4^2 + 5^2

class Sample{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int sum=0;

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

{

sum+=Math.pow(i,2);

}

System.out.println(sum);

sc.close();

}

}

----------------------------------------------------------------------------------

8. Program to find the factorial of a given number.

class SumofNumbers{

public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int prod=1;

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

{

prod\*=i;

}

System.out.println(prod);

sc.close();

}

}

----------------------------------------------------------------------------------

9. Program to find the value for the following expression.

1 + x + x^2 + x^3 + x^4 +........+ x^n

import java.util.Scanner;

class Series{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

int n = sc.nextInt();

int sum = 0;

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

sum += (int)Math.pow(x, i);

}

System.out.println(sum);

}

}

----------------------------------------------------------------------------------

10. Program to find the value for the following expression.

1 - x + x^2 - x^3 + x^4 -........+(-1)^n\*x^n

import java.util.Scanner;

class Series{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

int n = sc.nextInt();

int sum = 0;

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

sum += (int)Math.pow(-1,i)\*(int)Math.pow(x, i);

}

System.out.println(sum);

}

}

-----------------------------------------------------------------------------

11. Program to find the value for following expression.

1 + 1/x + 1/x^2 + 1/x^3 +......+ 1/x^n

import java.util.Scanner;

class Series{

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

int n = sc.nextInt();

double sum = 0;

if(x==0)

{

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

System.exit(0);

}

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

sum += 1/Math.pow(x, i);

}

System.out.println(sum);

}

}

-----------------------------------------------------------------------------

12. Program to find the value for following expression.

1 - 1/x + 1/x^2 - 1/x^3 +......+(-1)^n\*1/x^n

------------------------------------------------------------------------------

13. Program to evaluate the following expression

x^0 + x^2 + x^4 +....+ x^n

----------------------------------------------------------------------------

14. Program to print k consicutive multiplication tables upto t rows of each.

Input:

n k t

4 3 12

Output:

n X 1 = n n+1 X 1 n+2 X 1

4 X 1 = 4 | 5 X 1 = 5 | 6 X 1 = 6

4 X 2 = 8 | 5 X 2 =10 | 6 X 2 = 12

......

import java.util.Scanner;

class Multiplication

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,k,t;

n=sc.nextInt();

k=sc.nextInt();

t=sc.nextInt();

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

{

/\* System.out.print(n+" X "+i+" = "+(n\*i));

System.out.print("\t|\t");

System.out.print((n+1)+" X "+i+" = "+((n+1)\*i));

System.out.print("\t|\t");

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

System.out.println();

\*/

for(int j=0;j<k;j++)

{

System.out.print((n+j)+" X "+i+" = "+((n+j)\*i));

System.out.print("\t|\t");

}

System.out.println();

}

sc.close();

}

}

------------------------------------------------------------------------------

15. Program to print n fibonacci series?

import java.util.\*;

class Solution{

public static void main(String []ar){

Scanner s = new Scanner(System.in);

int n = s.nextInt();

int x=0,y=1;

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

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

int sum=x+y;

x=y;

y=sum;

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

}

}

}

----------------------------------------------------------------------------

10-08-22:

----------

Working with Numbers

Method:

A block of statements which are used to do a specific task is called a method.

Syntax:

------

AccessSpecifier return\_type method\_name(datatype arg1,datatype arg2,....)

{

//block of statements

}

Ex:

Task: To display welcome message to the given name as a argument.

public void display(String name)

{

System.out.println("Welcome to "+name);

}

Types of Methods:

-----------------

1. Method has return value and have arguments [Frequently used]

2. Method has return value and doesn't have the arguments [rarely used]

3. Method with no return value and have the arguments. [Frequently used]

4. Method with no return value and without arguments. [rarely used]

1. Write a program to find the factorial of a given number through methods.

import java.util.Scanner;

class Sample

{

public static int factorial(int m)

{

int fact=1;

for(int i=2;i<=m;i++)

fact=fact\*i;

return fact;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int result=factorial(n);

System.out.println(result);

sc.close();

}

}

working with break and continue:

-------------------------------

break:

break is used to exit from a loop.

continue:

continue is used to skip the current iteration and go for the next iteration.

------------------------------------------------------------------------------------

1. Write a program to print the first 20 numbers of the Pell series.

In mathematics, the Pell numbers are an infinite sequence of integers. The sequence of Pell numbers starts with 0 and 1, and then each Pell number is the sum of twice the previous Pell number and the Pell number before that

Expected Output :

The first 20 numbers of Pell series are:

0 1 2 5 12 29 70 169 408 985 2378 5741 13860 33461 80782 195025 470832 1136689 2744210 6625109

------------------------------------------------------------------------------

2. Write a program to check whether a given number is a perfect cube or not.

Expected Output :

Input: 125

Output: YES

import java.util.Scanner;

class Cube

{

public static boolean isCube(int n)

{ int k=(int)Math.ceil(Math.pow(n,1.0/3));

System.out.println(k);

if(k\*k\*k==n)

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isCube(n))

System.out.println(n+" is a Perfect Cube");

else

System.out.println(n+" is not a Perfect Cube");

}

}

------------------------------------------------------------------------------

3. Write a program to check if a number is Authomorphic or not.

Input: 76

Output: The given number 76 is an Automorphic Number.

Source Code:

============

import java.util.Scanner;

class Automorphic

{

public static boolean isAutomorphic(int n)

{

int d=(int)Math.log10(n)+1;

if((n\*n)%(int)Math.pow(10,d)==n)

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isAutomorphic(n))

System.out.println(n+" is a Automorphic Number");

else

System.out.println(n+" is not a Automorphic Number");

}

}

------------------------------------------------------------------------------

4. Program to find the number of digits of a given number.

Input: 123

Output: 3

import java.util.Scanner;

class Sample

{

public static int countofDigits(int n)

{

int count=0;

while(n>0)

{

count++;

n=n/10;

}

return count;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int result=countofDigits(n);

System.out.println(result);

}

}

or

import java.util.Scanner;

class Sample

{

public static int countofDigits(int n)

{

int count=(int)Math.log10(n)+1;

return count;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

int result=countofDigits(n);

System.out.println(result);

}

}

------------------------------------------------------------------------------

5. Program to find the sum of the digits of a given number.

Input: 1234

Output:10

------------------------------------------------------------------------------

6. Program to find the product of a given number.

Input: 123

Output:6

------------------------------------------------------------------------------

7. Program to find the given number is spy number or not.

Spy Number:

A spy number is a number where the sum of its digits equals the product of its digits.

For example, 1124 is a spy number, the sum of its digits is 1+1+2+4=8 and the product of its digits is 1\*1\*2\*4=8

Input: 22

Output: YES

------------------------------------------------------------------------------

9. Program to find the reverse of a given number.

Input: 123

Output:321

------------------------------------------------------------------------------

10. Program to find the given number is palindrome or not.

Palindrome: Reverse of the given number is equal to number itself is called Palindrome

Input: 323

Output: YES

Input: 123

Output: NO

Source Code:

============

import java.util.Scanner;

class Palindrome

{

public static boolean isPalindrome(int n)

{

int d=(int)Math.log10(n);

int found=0;

while(n>9)

{

if(n/(int)Math.pow(10,d) != n%10)

{

found=1;

break;

}

n=n%(int)Math.pow(10,d);

n=n/10;

d=d-2;

}

if(found==1)

return false;

else

return true;

}

public static boolean isPrime(int n)

{

for(int i=2;i<=(int)Math.sqrt(n);i++)

{

if(n%i==0)

{

return false;

}

}

return true;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isPalindrome(n))

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

else

System.out.println(n+" is not a Palindrome");

if(isPrime(n))

System.out.println(n+" is Prime Number");

else

System.out.println(n+" is not a Prime Number");

sc.close();

}

}

D:\DR>java Palindrome

121

121 is Palindrome

D:\DR>java Palindrome

5

5 is Palindrome

------------------------------------------------------------------------------

11. Write a program to find the given number is neon number or not.

Neon number:

A neon number is a positive integer, which is equal to the sum of the digits of its square. For example, 9 is a neon number, because 9 squared = 81, and the sum of the digits 8 + 1 = 9, which is the same as the original number.

Input: 9

Output: YES

------------------------------------------------------------------------------

12. Write a program to check whether a given number is Abundant or not.

An abundant number or excessive number is a number for which the sum of its proper divisors is greater than the number.

Input: 18

Output: YES

------------------------------------------------------------------------------

13. Write a program to check whether a given number is Armstrong number or not.

Armstrong Number:

A number is thought of as an Armstrong number if the sum of its own digits raised to the power number of digits gives the number itself.

Input: 153

Output: YES

Explanation: 1^3+5^3+3^3 = 1+125+27 = 153

Input: 1634

Output: YES

Explanation: 1^4 + 6^4 + 3^4 + 4^4 => 1634

Source Code:

============

import java.util.Scanner;

class Sample

{

public static boolean isArmstrong(int n)

{

int d=(int)Math.log10(n)+1;

int sum=0,r,temp=n;

while(n>0)

{

r=n%10;

sum=sum+(int)Math.pow(r,d);

n=n/10;

}

if(sum==temp)

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isArmstrong(n))

System.out.println(n+" is a Armstrong Number");

else

System.out.println(n+" is not a Armstrong Number");

}

}

------------------------------------------------------------------------------

14. Write a program to check whether a number is a Pronic Number or Heteromecic Number or not.

A pronic number is a number which is the product of two consecutive integers, that is, a number of the form n(n + 1).

Input: 132

Output: YES

Explanation: 132 => 11 \* 12 => 132

Source Code:

============

import java.util.Scanner;

class PronicNumber

{

public static boolean isPronic(int n)

{

int k=(int)Math.sqrt(n);

if(n==k\*(k+1))

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isPronic(n))

System.out.println(n+" is Pronic Number");

else

System.out.println(n+" is not a Pronic Number");

sc.close();

}

}

D:\DR>java PronicNumber

132

132 is Pronic Number

D:\DR>java PronicNumber

144

144 is not a Pronic Number

------------------------------------------------------------------------------

15. Write a program to check whether a given number is an ugly number or not.

An ugly number is a positive integer whose prime factors are limited to 2 , 3 , and 5.

Ugly Number:

Ugly numbers are numbers whose only prime factors are 2, 3 or 5.

Input: 25

Output:YES

Explanation:

------------

25 => 5 X 5, its prime factors include 5 only.

-------------------------------------------------------------------------------

16. Write a program to check two numbers are Amicable numbers or not.

Amicable numbers are two different numbers so related that the sum of the proper divisors of each is equal to the other number.

Input : x = 220, y = 284

Output : Yes

Proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110.

Sum of these is 284.

Proper divisors of 284 are 1, 2, 4, 71 and 142 with sum 220.

Input:

1184

1210

Output:

YES

-------------------------------------------------------------------------------

17. Write a program to check whether a given number is Deficient or not.

A deficient number or defective number is a number n for which the sum of divisors of n is less than 2n Equivalently, it is a number for which the sum of proper divisors (or aliquot sum) is less than n.

For example, the proper divisors of 8 are 1, 2, and 4, and their sum is less than 8, so 8 is deficient.

Input: 15

Output: YES

-------------------------------------------------------------------------------

18. Write a program to check whether a number is Disarium or not.

A disarium number is a number in which the sum of the digits to the power of their respective position is equal to the number itself (position is counted from left to right starting from 1).

Hence,175 is a disarium number. i.e 1^1 + 7^2 + 5^3 => 175

Input: 135

Output:YES

import java.util.Scanner;

class DisariumDemo

{

public static boolean isDisarium(int n)

{

int d=(int)Math.log10(n)+1;

int sum=0,r,temp=n;

while(n>0)

{

r=n%10;

sum=sum+(int)Math.pow(r,d);

n=n/10;

d--;

}

if(sum==temp)

return true;

else

return false;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n;

n=sc.nextInt();

if(isDisarium(n))

System.out.println(n+" is a Disarium Number");

else

System.out.println(n+" is not a Disarium Number");

}

}

-------------------------------------------------------------------------------

19. Write a program to check a number is a Happy or not.

Input a number: 13

13 is a Happy number.

-------------------------------------------------------------------------------

20. Write a program to check if a number is Harshad Number or not.

Input a number: 9

The given number is a Harshad Number.

-------------------------------------------------------------------------------

21. Write a program to check whether a number is a Duck Number or not.

Duck number is another special positive non-zero number that contains zero in it. The digit zero should not be presented at the starting of the number.

Input a number: 3210

The given number is a Duck Number.

-------------------------------------------------------------------------------

22. Write a program to check if a given number is circular prime or not.

Input a Number: 1193

The given number is a circular prime Number.

-------------------------------------------------------------------------------

23. Write a program to Check if a number is Mersenne number or not.

Mersenne number is a number that is in the form of M(n)=2^n-1. Where n is an integer.

If n is prime, the corresponding Mersenne number may be prime and the number called Mersenne prime number.

Input a number: 127

127 is a Mersenne number.

-------------------------------------------------------------------------------

24. Write a program to check if a number is Keith or not(with explanation).

Expected Output :

Input a number : 1537

1 + 5 + 3 + 7 = 16

5 + 3 + 7 + 16 = 31

3 + 7 + 16 + 31 = 57

7 + 16 + 31 + 57 = 111

16 + 31 + 57 + 111 = 215

31 + 57 + 111 + 215 = 414

57 + 111 + 215 + 414 = 797

111 + 215 + 414 + 797 = 1537

-------------------------------------------------------------------------------

25. Write a program in Java to check whether a number is Lychrel number or not.

Lychrel Number is a natural number that cannot form a palindrome through the iterative process of repeatedly reversing its digits and adding the resulting numbers.

Input : 196

Output : 196 is lychrel : true

Explanation : 196 becomes palindromic after 19 iterations :

196 + 691 = 887

887 + 788 = 1675

1675 + 5761 = 7436

7436 + 6347 = 13783

13783 + 38731 = 52514

....

16403234045 + 54043230461

70446464506 + 60546464407

Input a number: 196

The given number is Lychrel.

-------------------------------------------------------------------------------

26. Write a program in Java to check whether a given number is a Kaprekar number or not.

A Kaprekar number is a number whose square when divided into two parts and such that sum of parts is equal to the original number and none of the parts has value 0.

Input : n = 45

Output : Yes

Explanation : 452 = 2025 and 20 + 25 is 45

Input : n = 13

Output : No

Explanation : 132 = 169. Neither 16 + 9 nor 1 + 69 is equal to 13

Input : n = 297

Output : Yes

Explanation: 2972 = 88209 and 88 + 209 is 297

Input : n = 10

Output : No

Explanation: 102 = 100. It is not a Kaprekar number even if

sum of 100 + 0 is 100. This is because of the condition that

none of the parts should have value 0.

---------------------------------------------------------------------------------

27. Program to find the Unique digits in the given number.

Input: 1223

Output: 1 3

Input: 91232

Output:1 3 9

import java.util.Scanner;

class UniqueDigits

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,x[],r;

n=sc.nextInt();

x=new int[10];

while(n>0)

{

r=n%10;

x[r]++; //x[r]=x[r]+1

n=n/10;

}

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

{

if(x[i]==1)

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

}

sc.close();

}

}

D:\DR>java UniqueDigits

9125625

Output:

=======

1 6 9

-------------------------------------------------------------------------

28. Program to find the digits which are repeated in the given number.

Input: 912321

Output:1 2

import java.util.Scanner;

class UniqueDigits

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,x[],r;

n=sc.nextInt();

x=new int[10];

while(n>0)

{

r=n%10;

x[r]++; //x[r]=x[r]+1

n=n/10;

}

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

{

if(x[i]==1)

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

}

sc.close();

}

}

D:\DR>java DuplicateDigits

9125625

Output:

=======

2 5

--------------------------------------------------------------

29. Program to find the maximum repeatative digit in the given number.

Note: If in the number more than one digit having same repeatative count then display largest digit among them.

Input:9125625

Output: 5

import java.util.Scanner;

class DuplicateDigits

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,x[],r;

n=sc.nextInt();

x=new int[10];

while(n>0)

{

r=n%10;

x[r]++; //x[r]=x[r]+1

n=n/10;

}

int max=x[0];

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

{

if(x[i]>=x[max])

{

max=i;

}

}

System.out.print(max);

sc.close();

}

}

D:\DR>java DuplicateDigits

9125625

5

D:\DR>java DuplicateDigits

123456

6

D:\DR>java DuplicateDigits

123345

3

------------------------------------------------------------------------

30.

=================================================================================

ARRAYS

=================================================================================

13-08-22:

=========

Array:

Array is a fixed collection of homogenious elements stored in continueous memory locations and referred through a single name.

Declaration of an Array:

========================

datatype array\_name[]=new datatype[size];

Ex:

int x[]=new int[10];

String y[]=new String[10];

float z[]=new float[10];

....

Student s[]=new Studnet[10]; //Array of Objects

Note:

Once an array is declared with size then all the memory locations of that array are filled with default values.

int x[]=new int[5];

System.out.println(x[2]); // 0

Initialization:

1. Compiletime Initialization

Assining of vlaues into an array at the time of array declaration is called Compiletime Initialization.

int x[]={10,20,30,40};

System.out.println(x.length); // 4

2. Runtime Initialization

At the time of execution reading data from the user and stored in an array is called runtime Initialization.

class Array1

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int ages[],n;

n=sc.nextInt();

ages=new int[n];

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

ages[i]=sc.nextInt();

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

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

System.out.println();

System.out.println(Arrays.toString(ages)); //[ele1, ele2,.....]

for(int ele:ages)

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

}

}

----------------------------------------------------------------------------------

1. Program to find the Minimum and Maximum element in array.

import java.util.Scanner;

import java.util.Arrays;

class FindMinandMax

{

public static int getMinimum(int x[],int n)

{

int min=Integer.MAX\_VALUE;

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

{

if(x[i]<min)

min=x[i];

}

return min;

}

public static int getMaximum(int x[],int n)

{

int max=Integer.MIN\_VALUE;

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

{

if(x[i]>max)

max=x[i];

}

return max;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

int minimum=getMinimum(arr,n);

int maximum=getMaximum(arr,n);

System.out.println(minimum+" "+maximum);

sc.close();

}

}

D:\DR>java FindMinandMax

5

2 1 5 3 8

1 8

-----------------------------------------------------------------------------

2. Program to find the index of any one peak element in an array.

Peak Element: An element of its neighbours are smaller than it is called peak element.

Note: For first element only need to compare with next element.

For last element only need to compare with previous element.

Input:

6

1 2 3 4 5 6

Output: 5

Input:

8

2 1 5 4 3 2 3 1

Output: 0

import java.util.Scanner;

class FindPeakElement

{

public static int peakElement(int x[],int n)

{ //logic for finding peak element

if(n==1)

return 0;

if(x[0]>x[1])

return 0;

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

{

if(x[i]>x[i-1] && x[i]>x[i+1])

return i;

}

if(x[n-1]>x[n-2])

return n-1;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

int peak=peakElement(arr,n);

System.out.println(peak);

sc.close();

}

}

-----------------------------------------------------------------------------

3. Program to find the reverse of a given array.

import java.util.Scanner;

class Reverse

{

public static void reverseofArray(int x[],int n) //x is called as formal argument

{

int i=0,j=n-1;

while(i<j)

{

int temp;

temp=x[i];

x[i]=x[j];

x[j]=temp;

i++;

j--;

}

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

reverseofArray(arr,n); //arr is called as actual argument

for(int ele:arr)

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

sc.close();

}

}

-------------------------------------------------------------------------------------

4. Program to find the Kth maximum in an array.

Input:

8

1 2 3 4 5 6 7 8

3

Output:

6

8 7 6 5 4 3 2 1 => 6

Input2:

8

1 2 3 4 5 6 6 7

4

7 6 6 5 4 3 2 1

import java.util.Scanner;

import java.util.Array;

import java.util.Collections;

class KthMaximum

{

public static int kthMax(int x[],int n,int k)

{

Arrays.sort(x,Collections.reverseOrder());

int count=0;

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

{

if(x[i]!=x[i+1])

count++;

if(count==k-1)

break;

}

return x[i+1];

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i,k;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

k=sc.nextInt();

int max=kthMax(arr,n); //arr is called as actual argument

System.out.println(max);

sc.close();

}

}

Note:

In the each iteration you need to find the maximum element in that array and replaced with minimum value of integer and continue this process until you will get the kth maximum element.

--------------------------------------------------------------------------------

5. Program to move all negative numbers to beginning of an array.

Input:

8

1 -2 3 -4 -5 6 -7 -8

Output:

-2 -4 -5 -7 -8 .....

import java.util.Scanner;

class NegativeValuesMovement

{

public static void MoveNegativeNumbers(int x[],int n)

{

int i=0,j=0;

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

{

if(x[i]<0)

{

if(i!=j)

{

int temp=x[i];

x[i]=x[j];

x[j]=temp;

j++;

}

}

}

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

MoveNegativeNumbers(arr,n);

for(int ele:arr)

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

sc.close();

}

}

------------------------------------------------------------------------------

6. Program to find the missing number in an array.

Input:

5 sum1=15

1 2 4 5 [n-1 elements] sum2=12

diff=15-12 => 3

Output:

3

Shortest Way:

n => sum1= n\*(n+1)/2 (sum of n natural numbers)

sum of Array elements => sum2;

diff=sum1-sum2

I want to say diff is missing number?

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

{ found=0;

for(j=0;j<x.length;j++)

{

if(x[j]==i)

{

found=1;

break;

}

}

if(found==0)

{

System.out.println(i);

break;

}

}

2nd Approach: O(n^2);

3rd Approach:

sort(arr);

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

{

if(x[i]+1!=x[i+1])

{

System.out.println(x[i]+1);

System.exit(0);

}

}

-----------------------------------------------------------------------

7. Program to find the count pairs with given sum value.

Input:

5

1 5 4 5 -7

6

import java.util.Scanner;

class CountPairSum

{

public static void pairs(int x[],int n,int target)

{ int count=0;

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

{

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

{

if(x[i]+x[j]==target)

count++;

}

}

return count;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i,sum;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

sum=sc.nextInt();

int c=pairs(arr,n,sum);

System.out.println(c);

sc.close();

}

}

-----------------------------------------------------------------------------

8. Program to print Distnict/Unique values of an array [Remove the duplicate elemetns].

Input:

8

1 2 4 2 1 5 6 4

Output:

1 2 4 5 6

import java.util.Scanner;

import java.util.Arrays;

class DistnictElements

{

public static void UniqueElements(int x[],int n)

{

int y[]=new int[n];

int j=0;

Arrays.sort(x);

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

{

if(x[i]!=x[i+1])

System.out.println(x[i]);

y[j++]=x[i];

}

System.out.println(x[n-1]);

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int arr[],n,i;

n=sc.nextInt();

arr=new int[n];

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

arr[i]=sc.nextInt();

UniqueElements(arr,n);

sc.close();

}

}

------------------------------------------------------------------------------

String:

=======

1. Program to check the given Strings are Anagrams or not.

SILENT

LISTEN

anagrams or not?

S1= EILNST

S2= EILNST

char x[]=S1.toCharArray(); => to convert a string to character array.

char y[]=S2.toCharArray();

Arrays.sort(x);

Arrays.sort(y);

if(Arrays.equals(x,y))

System.out.println("Anagrams");

else

System.out.println("Not anagrams");

2. Program to check if the Strings are rotations of each other or not.

S1="ABCD"

DABC

CDAB

BCDA

ABCD

S2="BCDA"

S1=S1+S1;

S1="ABCDABCD"

S2="BCDA"

// S2 is part of S1 => it returns index of where it was matched.

// S2 is not part of S1 => It return -1.

if(S1.indexOf(S2)!=-1)

System.out.prinln("S2 is part of S1 in circular rotation");

else

System.out.println("S2 is not part of it in circular rotation");

========================================================================================

17-08-22

--------

Almost Prime:

=============

import java.util.Scanner;

class Sample

{

public static boolean isPrime(int n)

{

for(int i=2;i<=(int)Math.sqrt(n);i++)

{

if(n%i==0)

return false;

}

return true;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int t,m;

t=sc.nextInt();

while(t-->0)

{

m=sc.nextInt();

int found=0;

for(int j=2;j<=(int)Math.sqrt(m);j++)

{

if(m%j==0)

{

if(isPrime(j) && isPrime(m/j) && j!=m/j)

{

found=1;

break;

}

}

}

if(found==0)

System.out.println("NO");

else

System.out.println("YES");

}

sc.close();

}

}

-----------------------------------------------------------------------

Three Divisible upto N:

------------------------

import java.util.Scanner;

class ThreeDivisors

{

public static boolean isPrime(int n)

{

for(int i=2;i<=(int)Math.sqrt(n);i++)

{

if(n%i==0)

return false;

}

return true;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int m,count=0;

m=sc.nextInt();

for(int i=2;i<=(int)Math.sqrt(m);i++)

{

if(isPrime(i))

count++;

}

System.out.println(count);

sc.close();

}

}

------------------------------------------------------------------------------------

Arrays:

Arrays.sort(a)

Arrays.sort(a, Collections.reverseOrder());

Arrays.sort(originalArray,start\_index,end\_index)

Arrays.binarySearch(a,int)

Arrays.binarySearch(a,start\_index,end\_index,key)

Arrays.equals(a1,a2)

Arrays.fill(a,value)

Arrays.copyOfRange(originalArray,start\_index,end\_index)

Arrays.copyOfRange(originalArray,new\_Length)

Arrays.mismatch(array1,array2)

Arrays.asList(String[])

import java.util.Scanner;

class Fancy

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,k,r,m,d=1,c=1,i=1,found=0;

int x[]=new int[10];

n=sc.nextInt();

int len=(int)Math.log10(n)+1;

if(len>10)

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

else

{

r=n%10;

n=n/10;

x[r]++;

while(n>0)

{

k=n%10;

n=n/10;

if(k==r)

c++;

else if(k==r+1)

d++;

else if(k==r-1)

i++;

else

{

c=d=i=1;

}

if(c==3 || d==3 || i==3)

{

found=1;

break;

}

x[k]++;

r=k;

n=n/10;

}

if(found==0)

{

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

{

if(x[p]>=5)

{

found=1;

break;

}

}

}

if(found==0)

System.out.println("NOT a Fancy Number");

else

System.out.println("Fancy Number");

}

sc.close();

}

}

-----------------------------------------------------------------------------------------

22-08-22

========

Control Statements - Appy and Contest - 1

Control Statements - Excel Sheet Column - 1

Input Output

26 Z

51 AY

52 AZ

80 CB

676 YZ

702 ZZ

705 AAC

Arrays - Predict the Winner - 16

- Decompress Run-Length Encoded List - 15

- Create Target Array in the Given Order - 14

- Majority Element - 14

- Make Two Arrays Equal by Reversing Sub-arrays - 13

- Move Zeros - 13

- Replace Elements with Greatest Element on Right Side - 13

- Search in Rotated Sorted Array - 12

- Maximum Product of Two Elements in an Array - 10

- Rotation - 15

===========================================================================================

Program to find the pair of elements exists in an array their sum equal to target value.

Input:

1 2 3 4 5

6

Output:

Yes

Input:

1 2 3 4 5

12

Output:

No

import java.util.Scanner;

class Sample

{

public static boolean findPair(int x[],int sum)

{

HashMap<Integer,Integer> hm=new HashMap<>();

for(int i=0;i<x.length;i++) // 6 2 7 9 4 3 10

{

if(hm.containsKey(sum-x[i])))

return true;

else

hm.put(x[i],i); 6 0

2 1

} 7 2

return false; 9 3

}

public static void main(String args[])

{

int n,x[],target,i,j,found=0;

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

x=new int[n];

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

{

x[i]=sc.nextInt();

}

target=sc.nextInt();

boolean result=findPair(x,target);

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

{

for(j=0;j<n;j++)

{

if(x[i]+x[j]==target && i!=j)

{

found=1;

break;

}

}

}

if(found==1)

System.out.println("Yes");

else

System.out.println("No");

}

}

-------------------------------------------------------------------------------

Excel Sheet Column - 1

r

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int N;

N=sc.nextInt();

System.out.println(getCode(N));

sc.close();

}

}

--------------------------------------------------------

Appy and Contest - 1

import java.util.Scanner;

class Sample

{

public static int LCM(int m,int n)

{

int max=m>n?m:n;

int lcm=max;

while(true)

{

if(lcm%m==0 && lcm%n==0)

break;

lcm=lcm+max;

}

return lcm;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int t,A,B,K;

long N;

t=sc.nextInt();

while(t-- > 0)

{

N=sc.nextLong();

A=sc.nextInt();

B=sc.nextInt();

K=sc.nextInt();

int count=(int)N/A + (int)N/B - 2\*(int)N/LCM(A,B);

if(count>=K)

System.out.println("Win");

else

System.out.println("Lose");

}

}

}

-------------------------------------------------------------------------

Predict the Winner - 16

import java.util.Scanner;

class Sample

{

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,sum1=0,sum2=0;

n=sc.nextInt();

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

{

if(i%2==0)

sum1+=sc.nextInt();

else

sum2+=sc.nextInt();

}

if(Math.abs(sum1-sum2)%4==0)

System.out.println("X");

else

System.out.println("Y");

}

}

----------------------------------------------------------------------

Rotation - 15

class Solution

{

public static void reverse(int x[],int si,int ei)

{

while(si<ei)

{

int temp=x[si];

x[si]=x[ei];

x[ei]=temp;

si++;

ei--;

}

}

public void rotate(int[] nums, int k) {

if(k==0 || k==nums.length || nums.length==1)

return;

k=k%nums.length;

reverse(nums,nums.length-k,nums.length-1);

reverse(nums,0,nums.length-k-1);

reverse(nums,0,nums.length-1);

}

}

-------------------------------------------------------------------------

Decompress Run-Length Encoded List - 15

import java.util.Scanner;

class Sample

{

public static void main(String args[])

{

int n,x[];

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

x=new int[n];

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

{

x[i]=sc.nextInt();

}

for(int i=0;i<n;i=i+2)

{

int k=x[i];

while(k-->0)

System.out.print(x[i+1]+" ");

}

sc.close();

}

}

-------------------------------------------------------------------------

Create Target Array in the Given Order - 14

import java.util.Scanner;

class Sample

{

public static int[] getArray(int x[],int y[])

{

int output[]=new int[y.length];

int i,j=0;

for(i=0;i<y.length;i++)

{

if(y[i]<j)

{

int k=j;

while(k>y[i])

{

output[k]=output[k-1];

k--;

}

output[y[i]]=x[i];

j++;

}

else

{

output[y[i]]=x[i];

j++;

}

}

return output;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int m,n,x[],y[];

m=sc.nextInt();

x=new int[m];

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

x[i]=sc.nextInt();

n=sc.nextInt();

y=new int[n];

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

y[i]=sc.nextInt();

int result[]=getArray(x,y);

for(int ele:result)

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

sc.close();

}

}

--------------------------------------------------------------------------

Majority Element

import java.util.\*;

class Sample

{

public static int findMajorityElement(int x[])

{

HashMap<Integer,Integer> hm=new HashMap<>();

int key=0;

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

{

if(!hm.containsKey(x[i]))

hm.put(x[i],1);

else

hm.put(x[i],(int)hm.get(x[i])+1);

}

int n=(int)x.length/2;

for(Map.Entry<Integer,Integer> e:hm.entrySet())

{

int l=(int)e.getKey();

if((int)hm.get(l)>n)

{

key=l;

break;

}

}

return key;

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int n,x[];

n=sc.nextInt();

x=new int[n];

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

x[i]=sc.nextInt();

int c=findMajorityElement(x);

System.out.println(c);

sc.close();

}

}

--------------------------------------------------------------------------

Make Two Arrays Equal by Reversing Sub-arrays - 13

Note:

12 5 3 4

4 5 3 12

12 4 5 3

3 12 4 5

5 3 12 4

4 5 3 12

-------------------------------------------------------------------------------

Move Zeros - 13

import java.util.Scanner;

class Sample

{

public static void main(String args[])

{

int n,x[];

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

x=new int[n];

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

x[i]=sc.nextInt();

int j=0,found=0;

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

if(x[i]==0)

{

j=i;

found=1;

break;

}

if(found==1)

{

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

{

if(x[i]!=0 && i!=j)

{

int temp=x[i];

x[i]=x[j];

x[j]=temp;

j++;

}

}

}

for(int ele:x)

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

}

}

----------------------------------------------------------------------

Replace Elements with Greatest Element on Right Side - 13

import java.util.Scanner;

class Sample

{

public static void findArray(int x[],int n)

{

int rightmax=-1,newmax;

for(int i=n-1;i>=0;i--)

{

newmax=Math.max(rightmax,x[i]);

x[i]=rightmax;

rightmax=newmax;

}

}

public static void main(String args[])

{

int n,x[];

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

x=new int[n];

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

x[i]=sc.nextInt();

findArray(x,n);

for(int ele:x)

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

}

}

--------------------------------------------------------------------

Search in Rotated Sorted Array - 12

-------------------------------------------------------------------

Maximum Product of Two Elements in an Array - 10

1 5 4 3 2 6

public class ArrayApp {

public static int getMaxMultiplicationOfTwoNumbers(int[] arr) {

if(arr.length < 2) {

return -1;

}

if(arr.length == 2) {

return arr[0] \* arr[1];

}

int smallest = arr[0] < arr[1] ? arr[0] : arr[1];

int secondSmallest = arr[0] > arr[1] ? arr[0] : arr[1];

int greatest = arr[0] > arr[1] ? arr[0] : arr[1];

int secondGreatest = arr[0] < arr[1] ? arr[0] : arr[1];

for(int i = 2; i < arr.length; i++) {

if(arr[i] < smallest) {

secondSmallest = smallest;

smallest = arr[i];

} else if(arr[i] < secondSmallest) {

secondSmallest = arr[i];

}

if(arr[i] > greatest) {

secondGreatest = greatest;

greatest = arr[i];

} else if(arr[i] > secondGreatest) {

secondGreatest = arr[i];

}

}

int firstTwoMultiplication = smallest \* secondSmallest;

int lastTwoMultiplication = greatest \* secondGreatest;

return firstTwoMultiplication > lastTwoMultiplication ? firstTwoMultiplication : lastTwoMultiplication;

}

public static void main(String[] args) {

int n,x[];

Scanner sc=new Scanner(System.in);

n=sc.nextInt();

x=new int[n];

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

x[i]=sc.nextInt();

System.out.println(getMaxMultiplicationOfTwoNumbers(x));

}

}

--------------------------------------------------------------------

Count Element-I

Count the common elements between two arrays.

import java.util.Scanner;

import java.util.Arrays;

class Sample

{

public static int findCommonCount(int x[],int y[])

{

int c=0;

Arrays.sort(x);

for(int i=0;i<x.length-1;i++) 2 4 5 5 6 7 10

{ 6 5 9 8 1 3 4

if(x[i]==x[i+1])

continue;

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

{

if(x[i]==y[j])

{

c++;

break;

}

}

}

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

{

if(x[x.length-1]==y[j])

{

c++;

break;

}

}

return c;

}

public static void main(String args[])

{

int m,n,x[],y[];

Scanner sc=new Scanner(System.in);

m=sc.nextInt();

n=sc.nextInt();

x=new int[m];

y=new int[n];

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

x[i]=sc.nextInt();

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

y[i]=sc.nextInt();

int count=findCommonCount(x,y);

System.out.println(count);

sc.close();

}

}

--------------------------------------------------------------------------------------

1) Program to find unique elements in an array

2) Program to find union of two arrays

3) Program to find intersection of two arrays.

----------------------------------------------------------------------------------

4) Program to find count of each element in an array.

----------------------------------------------------------------------------------

5) Write a program which reads square matrix and then prints "Yes" if its Row Sums, Column Sums and Diagonal Sums all are same, otherwise prints "No".

Input:

3 3

1 2 3

4 5 6

7 8 9

Output:

No

import java.util.Scanner;

class MatixDemo

{

public static String matrixSumCalculation(int m[][],int row,int col)

{

int s1=0,s2=0,s3;

for(int i=0;i<r;i++) 6 3 6

{ 5 5 5

s1+=m[i][i]; 4 7 4

s2+=m[i][r-i-1];

}

if(s1!=s2)

return "No";

else

{

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

{

s2=0,s3=0;

for(int j=0;j<c;j++)

{

s2+=m[i][j];

s3+=m[j][i];

}

if(s1!=s2 || s1!=s3)

return "No";

}

}

return "Yes";

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in);

int r,c,x[][];

r=sc.nextInt();

c=sc.nextInt();

x=new int[r][c];

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

{

for(int j=0;j<c;j++)

x[i][j]=sc.nextInt();

}

System.out.println(matrixSumCalculation(x,r,c));

}

}

----------------------------------------------------------------------------------

6) Write a program which reads a 2D Matrix and then print the elements of the matrix by traversing in a row wise raster fashion.

Input:

4 5

4 5 5 5 5

1 4 4 5 6

2 2 2 2 2

1 4 4 4 8

Output:

4 5 5 5 5 6 5 4 4 1 2 2 2 2 2 8 4 4 4 1

import java.util.Scanner;

class Sample{

public static void printRasterForm(int[][] arr,int r,int c){

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

if(i % 2 == 0){

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

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

}

}

else{

for(int j = c-1; j >= 0; j--){

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

}

}

}

}

public static void main(String[] args){

Scanner sc = new Scanner(System.in);

int rows = sc.nextInt();

int cols = sc.nextInt();

int arr[][] = new int[rows][cols];

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

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

arr[i][j] = sc.nextInt();

}

}

printRasterForm(arr, rows, cols);

}

}

-----------------------------------------------------------------------------------

7) Program to print spiral matrix of a 2 Dimensional Matrix.

Input:

4 4

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 16

Output:

1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10

-------------------------------------------------------------------------------

8) Given a 2d grid map of 1's(land) and 0's (water), count the number of islands. An island is surrounded by water and is formed by connecting adjacent lands horizontally or vertically. You may assume all four edges of the grid are all surrounded by water.

Input:

4 5

1 1 1 1 0

1 1 0 1 0

1 1 0 0 0

0 0 0 0 0

Output: 1

-------------------------------------------------------------------------------------