SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

INSTITUTE OF PLACEMENT AND TRAINING

CSA09 –JAVA PROGRAMMING

1. Write a Java Program to Convert a Given Number of Days in Terms of Years, Weeks & Days.

Sample Input&Output::

Enter the number of days:756
No. of years:2
No. of weeks:3
No. of days:5

Test cases:

1. 38

2. 3.6

3. 0

4. -365

5. -45

Given a date, return the corresponding day of the week for that date.

The input is given as three integers representing the day, month and year respectively.

Return the answer as one of the following values {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"}.

Example 1:

Input: day = 31, month = 8, year = 2019

Output: "Saturday"

Example 2:

Input: day = 18, month = 7, year = 1999

Output: "Sunday"

Example 3:

Input: day = 15, month = 8, year = 1993

Output: "Sunday"

Constraints:

- The given dates are valid dates between the years 1971 and 2100.

```
import java.io.*;
import java.util.*;
class days
{
  public static void main(String args[])
  {
    try
    {

      Scanner sc=new Scanner(System.in);
      int days,year,weeks,days1;
      System.out.println("Enter the number of days:");
      days=sc.nextInt();
      if(days>0)
      {
        year=days/365;
        days=days%365;
        weeks=days/7;
        days1=days%7;
        System.out.println("The number of years:"+year);
        System.out.println("The number of weeks:"+weeks);
        System.out.println("The number of days:"+days1);
```

```
          }
          else
          {
            System.out.println("negative value not accepted");
          }
      }
      catch(Exception e)
      {
         System.out.println("floating point/string exception");
      }
   }
}
```

2. Write a program to find the number of student users in the college, get the total users, staff users details from the client. Note for every 3 staff user there is one Non teaching staff user assigned by default.
   Sample Input:
   Total Users: 856
   Staff Users: 126
   Sample Output:

Student Users: 688

Test Cases:

1. Total User: 0

2. Total User: -143

3. Total User: 1026, Staff User: 1026

4. Total User: 450, Staff User: 540

5. Total User: 600, Staff User: 450

```
import java.io.*;
import java.util.*;
class users
{
  public static void main(String args[])
  {
    try
    {
      Scanner sc=new Scanner(System.in);
      int t,s,su,su1,ns;
      System.out.println("Total users:");
      t=sc.nextInt();
      System.out.println("Staff users:");
      s=sc.nextInt();
      if(t<=0||s<=0)
      {
        System.out.println("negative value not accepted/null value not accepted..");
      }
      if(t<=s)
      {
        System.out.println("please enter the valid no.of.total users");
      }
      else
      {
       su=t-s-(s/3);

       System.out.println("Student users:"+su);
      }
    }
   catch(Exception e)
   {
     System.out.println("floating point/string exception");
   }
  }
}
```

3. Write a program to print number of factors and to print nth factor of the given number.

Sample Input:
Given Number: 100
N = 4
Sample Output:
Number of factors = 9
4th factor of 100 = 5
Test Cases:
1. Given Number = 512 , N = 6
2. Given Number = 343 , N = 7
3. Given Number = 1024 , N = 0
4. Given Number = -6561 , N = 3
5. Given Number = 0 , N = 2

```
import java.io.*;
import java.util.*;
class factor
{
   public static void main(String args[])
   {
       try
        {
          Scanner sc=new Scanner(System.in);
          int count=0,n=100,i,j=0,N;
          int []a=new int [10];
```

```
     System.out.println("Enter the number:");
     n=sc.nextInt();
     System.out.println("Enter the value of N:");
     N=sc.nextInt();
     if(n<=0)
     {
      System.out.println("Enter valid number");
     }
     else
     {
       for(i=1;i<=n;i++)
       {
        if(n%i==0)
        {
        a[j] = i;
        count++;
          j++;
        }
       }
       System.out.println("The number of factors:"+count);
       System.out.println(N + " th factor " + a[N-1]);
    }

   }
    catch(Exception e)
    {
      System.out.println("Enter only numbers");
    }
  }
}
```

4. Write a program to print n prime numbers after nth Prime number

   Sample Input:

   N = 3

   Sample Output:

   3rd Prime number is 5

3 prime numbers after 5 are: 7, 11, 13

Test cases:

1. N = P
2. N = 0
3. N = -4
4. N = 11
5. N = 7.2

5. Write a Program to create a list of all numbers in a range which are perfect squares and the sum of the digits of the number is less than 10.

Sample Input & Output:

Enter lower range: 1

Enter upper range: 40

[1, 4, 9, 16, 25, 36]

Test case:

1. Enter lower range: 50

   Enter upper range: 100

2. Enter lower range: 5

   Enter upper range: 8

3. Enter lower range: 10

   Enter upper range: 5

4. Enter lower range: 500

   Enter upper range: 500

5. Enter lower range: 0

   Enter upper range: -100

```
import java.util.*;
public class square
```

```
{
public static void main (String[] args)
{
try
{
 Scanner sc=new Scanner(System.in);
 int i;
 System.out.println("Enter lower range: ");
 int l=sc.nextInt();
 System.out.println("Enter upper range: ");
 int u=sc.nextInt();
 if(l>u)
 {
    System.out.println("The upper limit need to be high");

  }
 if(l<=0||u<=0)
 {
    System.out.println("Enter the valid limit");

  }
  if(l==u)
  {
    System.out.println("The both limit not should be equal");
  }
 else
 {
   for (i = l; i <= u; i++)
 {
  if (Math.sqrt(i) == (int)Math.sqrt(i))
  System.out.print(i + " ");
 }
 }
}
catch(Exception e)
{
   System.out.println("Enter the valid limit");
}
}
}
```

6. Write a program to print unique permutations of a given number
Sample Input:
Given Number: 143
Sample Output:

```
Permutations are:
134
143
314
341
413
431
```

Test cases:

1. 0
2. 111

3. 505
4. -143
5. -598

```
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;
class permutation
 {
 public static Set<String> getPermutation(String str)
{
 Set<String> permutations = new HashSet<String>();
 if (str == null)
 {
 return null;
 } else if (str.length() == 0) {
 permutations.add("");
 return permutations;
 }
 char first = str.charAt(0);
 String sub = str.substring(1);
 Set<String> words = getPermutation(sub);
 for (String strNew : words) {
 for (int i = 0;i<=strNew.length();i++){
 permutations.add(strNew.substring(0, i) + first + strNew.substring(i));
 }
 }
 return permutations;
 }
 public static void main(String[] args) {
```

```
Scanner input = new Scanner(System.in);
System.out.print("Enter the number: ");
String data = input.nextLine();
System.out.println("Permutations of " + data + ": \n" + getPermutation(data));
}
}
```

7. Write a Program to create an array with the First Element as the Number and Second Element as the Square of the Number.

Sample Input:

Enter the lower range:45
Enter the upper range:49

Sample Output:

[(45, 2025), (46, 2116), (47, 2209), (48, 2304), (49, 2401)]

Test case:

1. Enter lower range: 50

   Enter upper range: 100

2. Enter lower range: 5

   Enter upper range: 8

3. Enter lower range: 10

   Enter upper range: 5

4. Enter lower range: 500

   Enter upper range: 500

5. Enter lower range: 0

   Enter upper range: -100

```
import java.util.Scanner;
class sqlist
{
 public static void main(String[] args)
```

```
{
 Scanner sc=new Scanner(System.in);
 int ul,ll;
 System.out.print("Enter the lower limit: ");
 ll=sc.nextInt();

 System.out.print("Enter the upper limit: ");
 ul=sc.nextInt();
 if(ll>ul)
 {
   System.out.print("Enter the valid limits");
 }
 if(ll<0 ||ul<0 ||(ll<0 &&ul<0 ))
 {
     System.out.println("\nEnter the valid limits");
 }

 else
 {
 System.out.print("[");
 if(ll<ul){
 for(int j=ll;j<=ul;j++){
 System.out.print("("+j+","+(int)Math.pow(j,2)+")");
 if(j<ul)
 System.out.print(",");
 }
 }
 System.out.print("]");
 if(ul==ll)
 System.out.println("Both the limits are same");
}


 }
 }
```

8. Develop a JAVA code to display the balance. Include the following members:

- Design a class to represent a bank account.
- **Data Members:** Name of the depositor, Account number, Type of account(Savings/Current), Balance amount in the account(Minimum balance is Rs.500.00)
- **Methods:**
  1. To read account number, Depositor name, Type of account.
  2. To deposit an amount (Deposited amount should be added with it)
  3. To withdraw an amount after checking balance(Minimum balance must be Rs.500.00

  Note : Assume that balance amount = 10000

Test Cases

1. 100, Raja, S, 8000
2. Raja, 100, S, 9000
3. 101, Rani, S, 12000
4. 102, Ragu, W, 8000
5. 103, Ravi, C, 10000

```
import java.io.*;

import java.util.*;

class Account

{

  double balance;
```

```
Account()
{
  balance = 0;
}
Account(double sum)
{
 balance = sum;
}
double add(double sum)
{
  balance += sum;
  return sum;
 }
double withdraw(double sum)
{
  if (sum > balance) {
    balance -= 5;
    return -5;
  }
  else {
    this.balance -= sum;
    return balance; // Notice: always >= 0 (never < 0)
  }
}
double inquire()
{
  return balance;
}
  double interest (double rate)
 {
  return rate * balance;
 }
}
class bank1
{
       public static void main(String args[])
       {
     try
     {
        Scanner s=new Scanner(System.in);


        System.out.println("Enter account holder name:");
        String s1=s.next();
        System.out.println("Enter account type:");
        String s2=s.next();
              int b=0;
        System.out.println("Enter the initial value");


              b=s.nextInt();
              Account A;
              if (b==0){
                     A = new Account();
              }
              else{
                     A = new Account(b);
              }


              System.out.println("Enter the amount to withdraw");
```

```
                b=s.nextInt();


                double d = A.withdraw(b);
            System.out.println("Account holder name:"+s1);
            System.out.println("Account type:"+s2);
                if (d == -5) {
                        System.out.println("Penaly RS. -5 is charged since insufficient
balance");
                        System.out.println("Currrent balance" + A.inquire());
                }
                else{
                System.out.println("Now balance after withdraw of"+ A.inquire() + "is" + d);
                }
                System.out.println("Interest for current balance" + A.inquire() + " is " +
                        A.interest(0.006));
            }
        catch(Exception e)
        {
          System.out.println("Due to character exception");
        }
    }
    }
```

9. Develop a code to Reverse and Add a Number until you get a Palindrome.

Sample Input   If 7325 is input number, then
7325 (Input Number) + 5237 (Reverse Of Input Number) = 12562
12562 + 26521 = 39083
39083 + 38093 = 77176
77176 + 67177 = 144353
144353 + 353441 = 497794 (Palindrome)

Test Cases

1. 8765

2. -8765
3. 0
4. EIGHT FIVE
5. 87.57

```
import java.util.Scanner;

public class palinadd
{

        static int reverseNumber(int number)
        {
                int reverse = 0;

                int rem = 0;

                while (number != 0)
                {
                        rem = number % 10;

                        reverse = (reverse*10) + rem;

                        number = number/10;
                }

                return reverse;
        }


        static boolean checkPalindrome(int number)
        {
                int reverse = reverseNumber(number);

                if(reverse == number)
                {
                        return true;
                }
                else
                {
                        return false;
                }
        }
```

```
    static void reverseAndAdd(int number)
    {
        if(checkPalindrome(number))
        {
            System.out.println("Given Number is already a
palindrome");
        }
        else
        {
            while (!checkPalindrome(number))
            {
                int reverse = reverseNumber(number);

                int sum = number + reverse;

                System.out.println(number+" + "+reverse+" =
"+sum);

                number = sum;
            }
        }
    }

    public static void main(String[] args)
    {
      try
      {
        Scanner sc = new Scanner(System.in);

        System.out.println("Enter Number : ");

        int inputNumber = sc.nextInt();
        if(inputNumber<0)
      {
         System.out.println("Enter positive number");
      }
      else
      {
        reverseAndAdd(inputNumber);
      }
     }
     catch(Exception e)
```

```
                {
                   System.out.println("Enter a valid number");
                }
              }
     }
```

10. Create Customer class with deposit() and withdraw() as synchronized methods. Declare AccountNo, AccName and Balance as Instance Variables inside the class. From the main class, Input the amount for withdraw() operation and if requested amount is not available in existing Balance amount, withdraw() method should be temporarily suspended using wait() method until deposit() method receives the input for amount, to be added in the existing Balance amount and then withdraw() would be completed in a successful manner. Develop the above scenario using Synchronization and Inter-Thread Communication.

Note : existing Bank balance amount 10000
Sample Input : 12000, 3000
Sample Output : Withdraw operation success, balance amount 1000

Test Cases

1. 11000, 4000
2. -10000, -2000
3. 0, 0
4. EIGHT SEVEN, FIVE
5. 100.67, 200.68

GFG2

11. Given an integer n, return a string array answer (1-indexed) where:
answer[i] == "FizzBuzz" if i is divisible by 3 and 5.
answer[i] == "Fizz" if i is divisible by 3.
answer[i] == "Buzz" if i is divisible by 5.
answer[i] == i (as a string) if none of the above conditions are true.

Example 1:
Input: n = 3
Output: ["1","2","Fizz"]

Test Case

Test Case	Inputs
1.	n = 5
2.	n = 10
3.	n = 12
4.	n = 18
5.	n = 20

```
import java.util.*;
class fizz
{
        public static void main(String[] args)
         {
       try
       {
                int n;
           Scanner sc=new Scanner(System.in);
           System.out.println("Enter the value");
           n=sc.nextInt();
                for (int i=1; i<=n; i++)

                {
                        if (i%15==0)

                        System.out.println("FizzBuzz");

                        else if (i%5==0)
                        System.out.println("Buzz");

                        else if (i%3==0)
                                System.out.println("Fizz");

                        else
                System.out.println(i);
                }
      }
      catch(Exception e)
      {
        System.out.println("Due to character exception");
       }
        }
}
```

12. Write a Java program to find the common elements in two array of Positive integer

Sample Input:

[1, 2, 3, 4]
[2, 4, 5, 6, 7]

Expected output: [2, 4]

Test Case

Test Case	Inputs-1	Inputs-2
1.	[1, 2, 3, 4]	[4,5,6,7,8]
2.	[a, b, c, d]	[a, b, c, d]
3.	[1, -2, 3, 4]	[1,-2,5,7,8]

4.	[@, #, 34, 45]	[@,#,%,$,]
5.	[45,78,56,89]	[92,34,56,-78,-90]

```
import java.util.Arrays;
import java.util.*;
public class common
{
public static void main(String[] args)
   {
      Scanner sc=new Scanner(System.in);
      String [] array1;
      String [] array2;
      int a,b;
      int n1,n2,i,j;
      System.out.print("Enter the no. of elements for array 1: ");
      n1=sc.nextInt();
      array1 = new String[n1];
      System.out.print("Enter the elements:");
      for(a=0;a<n1;a++)
      {
        array1[a]=sc.next();
      }
      System.out.print("Enter the no. of elements for array 2: ");
      n2=sc.nextInt();
      array2 = new String[n2];
      System.out.print("Enter the elements:");
      for(b=0;b<n2;b++)
      {
        array2[b]=sc.next();
      }
      System.out.println("Array1 : "+Arrays.toString(array1));
      System.out.println("Array2 : "+Arrays.toString(array2));
      for (i = 0; i < n1; i++)
       {
          for (j = 0; j < n2; j++)
          {
             if(array1[i].equals(array2[j]))
             {

              System.out.println("Common element is : "+(array1[i]));
              }

          }
       }

   }
}
```

13. **Given a string** s **consisting of words and spaces, return the length of the** last **word** ***in the string.*** **A** word **is a maximal substring consisting of non-space characters only. There will be at least one word, consists of only English letters and spaces ' '.**

Example 1:
Input: s = "Hello World"
Output: 5
Explanation: The last word is "World" with length 5.

Test Case

Test Case	Inputs-1
1.	Maximal Substring Consisting
2.	**lea@st one wor2d**
3.	1254 98076
4.	& * ( ) % # $
5.	letters and spaces

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

     Scanner sc=new Scanner(System.in);
     String str1;

         System.out.println("Enter the string:");
     str1=sc.nextLine();
         System.out.println("Original String: "+str1);
     System.out.println("Length of the last word of the above string: "+length_Of_last_word(str1));
   }

   public static int length_Of_last_word(String str1) {
     int length_word = 0;
     String[] words = str1.split(" ");
     if(words.length>0) {
       length_word = words[words.length-1].length();
     } else {
       length_word = 0;
     }
     return length_word;
   }
}
```

14. Write a program to read a character until a ***** is encountered. Also count the number of uppercase, lowercase, and numbers entered by the users.

Sample Input:
Enter * to exit…

```
Enter any character: W
Enter any character: d
Enter any character: A
Enter any character: G
Enter any character: g
Enter any character: H
Enter any character: *
Sample Output:
Total count of lower case:2
Total count of upper case:4
Total count of numbers =0
```

Test Case

Test Case	Inputs-1
1.	1,7,6,9,5
2.	S, Q, l, K,7, j, M
3.	M, j, L, &, @, G
4.	D, K, I, 6, L, *
5.	*, K, A, e, 1, 8, %, *

```
import java.util.Scanner;
class alpha
{
 public static void main(String[] args)
 {
 Scanner s= new Scanner(System.in);
 System.out.print("Enter any Character: ");
 char ch=s.next().charAt(0);
 int l_count=0,u_count=0,num_count=0;
 while (ch!='*')
 {
 if(ch>='A'&&ch<='Z')
 u_count++;
 if(ch>='a'&&ch<='z')
 l_count++;
 if(Character.isDigit(ch))
 num_count++;
 System.out.print("Enter any Character: ");
 ch=s.next().charAt(0);
 }
 System.out.println("Total count of lower case: "+l_count);
 System.out.println("Total count of upper case: "+u_count);
 System.out.println("Total count of numbers: "+num_count);

 }
```

```
    }
```

15. Roman numerals are represented by seven different symbols: I, V, X, L, C, D and M.

Symbol	**Value**
I	**1**
V	**5**
X	**10**
L	**50**
C	**100**
D	**500**
M	**1000**

For example, 2 is written as II in Roman numeral, just two ones added together. 12 is written as XII, which is simply X + II. The number 27 is written as XXVII, which is XX + V + II.

Roman numerals are usually written largest to smallest from left to right. However, the numeral for four is not IIII. Instead, the number four is written as IV. Because the one is before the five we subtract it making four. The same principle applies to the number nine, which is written as IX. There are six instances where subtraction is used:

- **I can be placed before V (5) and X (10) to make 4 and 9.**
- **X can be placed before L (50) and C (100) to make 40 and 90.**
- **C can be placed before D (500) and M (1000) to make 400 and 900.**

Given a roman numeral, convert it to an integer.

Example:
Input: s = "III"
Output: 3

Test Case	**Inputs**
1.	LVIII
2.	MCMXCI
3.	V
4.	LZAII
5.	MCCDTIV

```
import java.util.*;

class rome1

{

static int romanToInt(String s)

 {

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



   map.put('I',1);
```

```
     map.put('V',5);
     map.put('X',10);
     map.put('L',50);
     map.put('C',100);
     map.put('D',500);
     map.put('M',1000);
    int result=map.get(s.charAt(s.length()-1));
    for(int i=s.length()-1;i>0;i--)
    {
        if(map.get(s.charAt(i))<=map.get(s.charAt(i-1)))
        {
            result+=map.get(s.charAt(i-1));
        }
        else
        {
            result-=map.get(s.charAt(i-1));
        }
    }
    return result;
}
public static void main(String args[])
{
 try
 {
 Scanner sc=new Scanner(System.in);
 String s1;
 System.out.println("Enter the string:");
 s1=sc.next();
 int b=romanToInt(s1);
 System.out.println(b);

 }

catch(Exception e)

{

  System.out.println("Enter the correct string");

}

}

}
```

16. Given two strings ransomNote and magazine, return true if ransomNote can be constructed by using the letters from magazine and false otherwise. Each letter in magazine can only be used once in ransomNote.

Example 1:
Input: ransomNote = "a", magazine = "b"
Output: false

Test Case

Test Case	Inputs
1.	ransomNote = "aa", magazine = "ab"
2.	ransomNote = "aa", magazine = "aab"
3.	ransomNote = "abc", magazine = "abc"
4.	ransomNote = "good", magazine = "better"
5.	ransomNote = "xyz", magazine = "123"

```
import java.io.*;

import java.util.*;

class magazine


{


   static boolean canConstruct(String ransomNote, String magazine) {


      if (ransomNote==null || ransomNote.length() == 0) return true;

      if (magazine==null || magazine.length() == 0) return false;
```

```
    boolean flag=false;
    boolean visited[]=new boolean[magazine.length()];
    Arrays.fill(visited,Boolean.FALSE);
    for(int i=0;i<ransomNote.length();i++){
      flag=false;
      for(int j=0;j<magazine.length();j++){
        if(ransomNote.charAt(i)==magazine.charAt(j) && !visited[j] ){
          visited[j]=true;
          flag=true;
          break;
        }
      }
    }
    return flag;
  }
 public static void main(String args[])
 {
  Scanner sc=new Scanner(System.in);
  String s,s1;
  System.out.println("ransomeNote=");
  s=sc.next();
  System.out.println("magazine=");
  s1=sc.next();
  System.out.println(canConstruct(s,s1));


 }
}
```

17. You are given an m x n binary matrix mat of 1's (representing soldiers) and 0's (representing civilians). The soldiers are positioned in front of the civilians. That is, all the 1's will appear to the left of all the 0's in each row.
A row i is weaker than a row j if one of the following is true:
The number of soldiers in row i is less than the number of soldiers in row j.
Both rows have the same number of soldiers and i < j. Return the indices of the k weakest rows in the matrix ordered from weakest to strongest.

Example 1:

Input: mat =
[[1,1,0,0,0],
[1,1,1,1,0],
[1,0,0,0,0],
[1,1,0,0,0],
[1,1,1,1,1]],
k = 3
Output: [2,0,3]

Explanation:

The number of soldiers in each row is:
- Row 0: 2
- Row 1: 4
- Row 2: 1
- Row 3: 2
- Row 4: 5

The rows ordered from weakest to strongest are [2,0,3,1,4].

Example 2:

Input: mat =
[[1, 0, 0, 0],
[1, 1, 1,1],
[1, 0, 0, 0],
[1, 0, 0,0]],
k = 2
Output: [0,2]
Explanation:
The number of soldiers in each row is:
- Row 0: 1
- Row 1: 4
- Row 2: 1
- Row 3: 1

The rows ordered from weakest to strongest are [0, 2, 3, 1].

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

```
  {
   int [][]a;
   int row,col,i,j,K;
   Scanner sc=new Scanner(System.in);
   System.out.println("Enter the no.of.rows:");
   row=sc.nextInt();
   System.out.println("Enter the no.of.coloumns:");
   col=sc.nextInt();
   a=new int[row][col];
   for(i=0;i<row;i++)
   {
    for(j=0;j<col;j++)
    {
      a[i][j]=sc.nextInt();
    }
   }
   System.out.println("k:");
   K=sc.nextInt();
   int A=kWeakestRows(a[][],K);
   System.out.println(A);
  }
   public int[] kWeakestRows(int[][] M, int K) {
    int y = M.length, x = M[0].length, kix = 0;
    int[] vis = new int[y], ans = new int[K];
    for (int j = 0; j <= x; j++)
      for (int i = 0; i < y; i++) {
        if (vis[i] == 0 && (j == x || M[i][j] == 0)) {
          ans[kix++] = i;
          vis[i]++;
        }
        if (kix == K) return ans;
      }
    return ans;
  }
}
```

18. Given an integer num, return the number of steps to reduce it to zero. In one step, if the current number is even, you have to divide it by 2, otherwise, you have to subtract 1 from it.

Example 1:

Input: num = 14
Output: 6
Explanation:
Step 1) 14 is even; divide by 2 and obtain 7.
Step 2) 7 is odd; subtract 1 and obtain 6.
Step 3) 6 is even; divide by 2 and obtain 3.

Step 4) 3 is odd; subtract 1 and obtain 2.
Step 5) 2 is even; divide by 2 and obtain 1.
Step 6) 1 is odd; subtract 1 and obtain 0.

Test Case

Test Case	Inputs
1.	n = 5
2.	n = 10
3.	n = 12
4.	n = 18
5.	n = 20

```
import java.io.*;
import java.util.*;
class reduce
{
 public static void main(String arg[])
 {
  try
  {
    Scanner sc=new Scanner(System.in);
    int n,s=0;
    System.out.println("Enter the number:");
    n=sc.nextInt();
    while(n>0)
    {
      if(n%2==0)
      {
        n=n/2;
      }
      else
      {
         n--;
      }
      s++;
  }
  System.out.println("The no.of.steps:"+s);
  }
 catch(Exception e)
  {
   System.out.println("Due to string exception");
  }
}
}
```

19. Develop a programme that uses Multiple Inheritance concepts to compute a student's grades in six subjects. The total and aggregate are then calculated, and the student's

grade is displayed. If the student achieves an aggregate of more than 75%, the grade is Distinction. If the aggregate is between 60 and 75, the grade is First Division. If the aggregate is between 50 and 60, the grade is Second Division. If the aggregate is between 40 and 50, the grade is Third Division. Otherwise, the grade is FAIL.

Sample Input & Output:
Enter the marks in python: 90
Enter the marks in c programming: 91
Enter the marks in Mathematics: 92
Enter the marks in Physics: 93
Enter the marks in Chemistry: 92
Enter the marks in Professional Ethics: 93
Total= 551
Aggregate = 91.83
Class: DISTINCTION

Test Case

Test Case	Inputs
1.	18, 76,93,65,63,98
2.	73,78,79,75,87,0
3.	98,106,120,95,98,34
4.	96,73, -85,95,84,98
5.	78,59.8,76,79,97,67

20. Write a program to calculate tax given the following conditions:

a. If income is less than or equal to 2,50,000 then no tax
b. If taxable income is 2,50,001 – 5,00,000 the charge 10% tax
c. If taxable income is 5,00,001 – 10,00,000 the charge 20% tax
d. If taxable income is above 10,00,001 then charge 30% tax

Sample Input:
Enter the income: 600000
Sample Output:
Taxable Income: 350000
Tax= 35000

Test Case

Test Case	Inputs
1.	400700
2.	2789239
3.	150000
4.	00000
5.	-125486

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

```
try
 {
  Scanner sc=new Scanner(System.in);
  int i,ti,t=0;
  System.out.println("Enter the income:");
 i=sc.nextInt();
  if(i<=0)
  {
   System.out.println("Enter the valid income:");
  }
  else
  {
    if(i<=250000)
    {

       t=0;
    }
    if(i>250000&&i<=500000)
    {
       ti=i-250000;
       System.out.println("Taxable income:"+ti);
       t=(i-250000)*10/100;
    }
    if(i>500000&&i<=1000000)
    {
      ti=i-250000;
      System.out.println("Taxable income:"+ti);
      if(ti>250000&&ti<=500000)
      {
         t=ti*10/100;
      }
    }
    if(i>1000000)
    {
      ti=i-250000;
      System.out.println("Taxable income:"+ti);
      if(ti>250000&&ti<=500000)
      {
         t=ti*10/100;
      }
      if(ti>500000&&ti<=1000000)
      {
         t=ti*20/100;
      }
      if(ti>1000000)
      {
             t=ti*30/100;
           }
         }
         System.out.println("Tax:"+t);
        }
       }
    catch(Exception e)
    {
     System.out.println("Enter the valid income:");
    }
  }
}
```