

1. Write java program to display Fibonacci series up to a limit.

Class diagram:

Fib

~n1: int=0

~n2: int=1

~n3: int

~limit: int

~i: int

```
import java.io.*;
import java.util.*;
class Fib
{
    public static void main(String args[])
    {
        Scanner in=new Scanner(System.in);
        int n1=0,n2=1,n3,limit,i;
        System.out.println("ENTER THE
LIMIT:");
        limit=in.nextInt();
        System.out.print(n1+"\t"+n2);
        for(i=2;i<limit;i++)
        {
            n3=n1+n2;
```

```

        n1=n2;
        n2=n3;
        System.out.print("\t"+n3);
    }
}

```

output:

ENTER THE LIMIT:

5

0 1 1 2 3

2. Write java program to display Armstrong numbers within a range.

Class diagram:

Armstrong

~l: int

~h:int

~i: int

~rem: int

~sum: int

~num=int

Import java.util.Scanner;

```

Public class Armstrong {
    Public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter lower limit:");
        Int lower = scanner.nextInt();
        System.out.println("Enter upper limit:");
        Int upper = scanner.nextInt();

        For (int num = lower; num <=upper ; num++) {
            Int j=num;
            Int original=num;
            Int sum=0;

            Int length=Integer.toString(num).length();
            While (j!=0){
                Int remainder=j%10;
                Sum+=Math.pow(remainder,length);
                J=j/10;
            }
        }
    }
}

```

```
    }  
    if(original==sum){  
        System.out.println(original);  
    }  
}  
}
```

output:

ENTER THE LOWER LIMIT AND UPPER LIMIT:

1

500

ARMSTRONG NUMBERS:

153

370

371

407

3. Write a program to perform base conversion

a) Integer to binary

b) Integer to Octal

c) Integer to Hexadecimal

class diagram:

Base

~num: int

~rem:int

~base: int

~str: String

~dig[]: char

```
import java.io.*;
import java.util.*;
class Base
{
    public static void main(String args[])
    {
        Scanner in =new Scanner(System.in);
        int num,rem,base;
        String str="";
        char
dig[]={ '0','1','2','3','4','5','6','7','8','9','A'
,'B','C','D','E','F'};
        System.out.println("ENTER THE NUMBER:");
        num=in.nextInt();
        System.out.println("ENTER THE BASE TO
CONVERT:");
        base=in.nextInt();
        while (num>0)
        {
            rem=num%base;
            str=dig[rem]+str;
            num=num/base;
        }
        System.out.println(str);
    }
}
```

output:

ENTER THE NUMBER:

10

ENTER THE BASE TO CONVERT:

2

1010

4. Write a program to merge two arrays.

Class diagram:

Merge

~m: int

~n: int

~i: int

~i: int

~j: int

~k: int

~a1[]: int

~a2[]: int

~a3[]: int

```
import java.io.*;
import java.util.*;
class Merge
{
    public static void main(String args[])
    {
        Scanner in=new Scanner(System.in);
        int m,n,i,j,k=0;
        int a1[]=new int[10];
        int a2[]=new int[10];
        int a3[]=new int[20];
        System.out.println("enter the size of
array 1:");
        m=in.nextInt();
        System.out.println("enter the elements:");
        for(i=0;i<m;i++)
        {
            a1[i]=in.nextInt();
        }
        System.out.println("enter the size of
array 2:");
        n=in.nextInt();
```

```
System.out.println("enter the elements:");
for(i=0;i<n;i++)
{
    a2[i]=in.nextInt();
}
i=0;
j=0;
k=0;
while(i<m&& j<n)
{
    if(a1[i]<a2[j])
    {
        a3[k]=a1[i];
        i++;
    }
    else
    {
        a3[k]=a2[j];
        j++;
    }
    k++;
}
if(i>=m)
{
    while(j<n)
    {
        a3[k]=a2[j];
        j++;
        k++;
    }
}
if(j>=n)
{
    while(i<m)
    {
        a3[k]=a1[i];
        i++;
        k++;
    }
}
System.out.println("after merging:");
```

```

        for(i=0;i<m+n;i++)
        {
            System.out.println(a3[i]);
        }
    }
}

```

output:

enter the size of array 1:

3

enter the elements:

10

30

50

enter the size of array 2:

2

enter the elements:

20

40

after merging:

10

20

30

40

50

5. Write a program to find the trace and transpose of a matrix.

Class diagram:

Matrix

~n: int

~m: int

~i: int

~j: int

~trace: int=0

~a[][]: int

~b[][]: int

```
import java.io.*;
import java.util.*;
class Matrix
{
    public static void main(String args[])
    {
        Scanner in=new Scanner(System.in);
        int n,m,i,j,trace=0;
        int a[][]=new int[10][10];
        int b[][]=new int[10][10];
        System.out.println("ENTER THE ORDER OF THE
MATRIX:");
        m=in.nextInt();
        n=in.nextInt();
        System.out.println("ENTER THE ELEMENTS:");
        for(i=0;i<m;i++)
        {
```

```

        for(j=0;j<n;j++)
        {
            a[i][j]=in.nextInt();
        }
    }
    for(i=0;i<m;i++)
    {
        for(j=0;j<n;j++)
        {
            b[j][i]=a[i][j];
            if(i==j)
            {
                trace=trace+a[i][j];
            }
        }
    }
    System.out.println("TRACE:"+trace);
    System.out.println("TRANSPOSE");
    for(i=0;i<m;i++)
    {
        for(j=0;j<n;j++)
        {
            System.out.print(b[i][j]);
        }
        System.out.println(" ");
    }
}

```

output:

ENTER THE ORDER OF THE MATRIX:

3

3

ENTER THE ELEMENTS:

1 2 3

4 5 6

7 8 9

TRACE:15

TRANSPOSE

147

258

369

6. Write java program to find the sum of the digits and reverse of a given number using class and objects

class diagram:

Ten

~num: int

~rem: int

~sum: int=0

~rev: int=0

~sumrev()

```
import java.io.*;
import java.util.*;
class Ten
{
    int num,rem,sum=0,rev=0;
    public static void main(String args[])
    {
        Ten obj=new Ten();
        obj.sumrev();
    }
    void sumrev()
    {
        Scanner in=new Scanner(System.in);
        System.out.println("ENTER THE NUMBER:");
        num=in.nextInt();
        while(num!=0)
        {
            rem=num%10;
            sum=sum+rem;
            rev=rem+(rev*10);
        }
    }
}
```

```

        num=num/10;
    }
    System.out.println("SUM:"+sum) ;
    System.out.println("REVERSE:"+rev) ;
}
}

```

output:

ENTER THE NUMBER:

123

SUM:6

REVERSE:321

7. Using class and objects, write a java program to find the sum of two complex numbers (Hint: Use object as parameter to function).

Class diagram:

Complex

~real: int

~image: int

add(complex c1, complex c2):Complex

```

import java.io.*;
import java.util.*;
class Complex
{

```

```

int real,image;
public static void main(String args[])
{
    Scanner in=new Scanner(System.in);
    Complex c1=new Complex();
    Complex c2=new Complex();
    System.out.println("ENTER THE FIRST
COMPLEX NUMBER:");
    c1.real=in.nextInt();
    c1.image=in.nextInt();
    System.out.println("ENTER THE FIRST
COMPLEX NUMBER:");
    c2.real=in.nextInt();
    c2.image=in.nextInt();
    c1.add(c1,c2);
}
void add(Complex c1,Complex c2)
{
    Complex c3=new Complex();
    c3.real=c1.real+c2.real;
    c3.image=c1.image+c2.image;

    System.out.println("SUM:"+c3.real+"+"+c3.image);
}
}

```

output:

ENTER THE FIRST COMPLEX NUMBER:

10

20

ENTER THE FIRST COMPLEX NUMBER:

30

40

SUM:40+60

8. Write a multi thread java program for displaying odd numbers and even numbers up to a limit (Hint: Implement thread using Runnable interface).

Class diagram:

Odd

~i:int

~n:int=10

+run: void

Even

~i:int

~n:int=10

+run: void

OddEvenThread

ob1:Odd

ob2:Even

```
import java.io.*;
import java.util.*;
class Odd implements Runnable
{
    int i,n=10;
    public void run()
    {
        for(i=1;i<n;i=i+2)
        {
            System.out.println("ODD:"+i);
        }
    }
}
class Even implements Runnable
{
    int i,n=10;
    public void run()
    {
        for(i=0;i<n;i=i+2)
        {
            System.out.println("EVEN:"+i);
        }
    }
}
```

```
class OddEvenThread
{
    public static void main(String args[])
    {
        Even ob2=new Even();
        Thread obj2=new Thread(ob2);
        obj2.start();
        Odd ob1=new Odd();
        Thread obj1=new Thread(ob1);
        obj1.start();
    }
}
```

output:

ODD:1

ODD:3

ODD:5

ODD:7

ODD:9

EVEN:0

EVEN:2

EVEN:4

EVEN:6

EVEN:8