## WEEK:2-A

**AIM:** Write a Java program that prompts the user for an integer and then prints out all prime numbers up to that integer..

**File Name:** PrimeNumbers.java

```
import java.util.Scanner;
class PrimeNumbers
{
        public static void main(String[] args)
        {
                int n;
                int p;
                Scanner s=new Scanner(System.in);
                System.out.println("Enter a number: ");
                n=s.nextInt();
                for(int i=2;i<n;i++)
                {
                        p=0;
                        for(int j=2;j<i;j++)
                        {
                                if(i%j==0)
                                p=1;
                        }
                        if(p==0)
                                System.out.println(i);
                }
        }
}
```

## WEEK:2-B

**AIM:**write a java program that reads a line of integers and then display each integer and the sum of all integers.

**File Name:** stoken.java

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

class stoken
{
public static void main(String arg[]) throws IOException {
int sm=0,no;
String s;
DataInputStream in=new DataInputStream(System.in);
System.out.print("Enter numbers ?");
s=in.readLine();
StringTokenizer token =new StringTokenizer(s);
while(token.hasMoreTokens())
{
no=Integer.parseInt(token.nextToken());
sm+=no;
System.out.println(no);
}
System.out.println("Sum : "+sm);
}
}
```

## WEEK:3-A

**Aim:** write a java program that checks whether the given string is palindrome or not.

```
import java.io.*;
class pal
{
public static void main(String arg[]) throws IOException
{
int i,j,len;
String s;
DataInputStream in=new DataInputStream(System.in);
System.out.print("Enter a string ?");
s=in.readLine();
len=s.length();
for(i=0,j=len-1;i<len/2;i++,j--)
{
if(s.charAt(i)!=s.charAt(j))
break;
}
if(i==j)
System.out.println("Pal string");
else
System.out.println("Not a Pal string");
}
}
```

## WEEK:3-B

**Aim:** Write a Java program for sorting a given list of names in ascending order.

```
import java.util.*;
class Sorting
{
        void sortStrings( )
        {
                Scanner s = new Scanner(System.in);
                System.out.println("Enter the value of n: ");
                int n = s.nextInt();
                String[] str = new String[n];
                System.out.println("Enter strings: ");
                for(int i = 0; i< n; i++)
                {
                        str[i] = new String(s.next());
                }
                for(inti = 0; i< n; i++)
                {
                        for(int j = i+1; j < n; j++)
                        {
                                if(str[i].compareTo(str[j])>0)
                                {
                                        String temp = str[i];
                                        str[i] = str[j];
                                        str[j] = temp;
                                }
                        }
                }
                System.out.println("Sorted list of strings is:");
                for(int i = 0; i< n ; i++)
                {
                        System.out.println(str[i]);
                }
        }
}
class Driver
{
        public static void main(String[] args)
        {
                Sorting obj = new Sorting();
                obj.sortStrings();
        }
}
```

**WEEK:3-C**

**Aim:** Write a Java program to make frequency count of words in a given text.

```
Import java.io.*;
Class FrequencyCount
{
   Public static void main(String args[]) throwsIOException
   {
      BufferedReader br=newBufferedReader(newInputStreamReader(System.in));
      System.out.println("Enter the String: ");
      String s=br.readLine();
      System.out.println("Enter substring: ");
      String sub=br.readLine();
      Int ind,count=0;
      for(inti=0; i+sub.length()<=s.length(); i++)
                                          //i+sub.length() is used to reduce comparisions
      {
         ind=s.indexOf(sub,i);
         if(ind>=0)
         {
            count++;
            i=ind;
            ind=-1;
         }
      }
      System.out.println("Occurence of '"+sub+"' in String is "+count);

   }
}
```

**WEEK:4-A**

**Aim:** Write a Java program that reads a file name from the user, then displays information about whether the file exists, whether the file is readable, whether the file is writable, the type of file and the length of the file in bytes.

## Algorithm:

1. import java.io.*; package

```
2. create class Filedemo{
3. Create function: public static void p(String str){
    1. Print String: System.out.println(str);
4. Againg create another function: public static void analyze(String s){
    1. File f = new File(s)'
5. if( f.exists()){
    1. p(f.getName()+" is a file");
    2.  p(f.canRead()?" is readable":" is not readable");

    3. p(f.canWrite()?" is writable":" is not writable");

    4. p("Filesize:"+f.length()+" bytes");

    5. p("File last mdified:"+f.lastModified());

6.   if(f.isDirectory()) then

    1. p(f.getName()+" is directory");

    2. p("List of files");

    3. String dir[]=f.list();

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

    5. p(dir[i])

7. Create main function: public static void main(String rr[])throws IOException{
    1. filedemo fd=new filedemo();
    2. BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    3. System.out.println("Enter the file name:");
    4. String s=br.readLine();

    5. fd.analyze(s);
import java.io.*;
class filedemo
{
public static void p(String str)
{
    System.out.println(str);
 }
 public static void analyze(String s)
File f=new File(s);
 if(f.exists())
 {
 p(f.getName()+" is a file");
 p(f.canRead()?" is readable":" is not readable");
 p(f.canWrite()?" is writable":" is not writable");
 p("Filesize:"+f.length()+" bytes");
 p("File last mdified:"+f.lastModified());
 }
 if(f.isDirectory())
 {
 p(f.getName()+" is directory");
 p("List of files");

String dir[]=f.list();
 for(int i=0;i<dir.length;i++)
 p(dir[i]);
 }
 }
}
public class FileDetails
{
 public static void main(String rr[])throws IOException
 {
 filedemo fd=new filedemo();
 BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
 System.out.println("Enter the file name:");
 String s=br.readLine();
 fd.analyze(s);
 }
}
```

## WEEK:4-B

**Aim:** Write a Java program that reads a file and displays the file on the screen, with a line number before each line.

```
import java.util.*;
import java.io.*;
class Rfile
{
	public static void main(String args[])throws IOException
	{
		int j=1;
		char ch;
		Scanner scr=new Scanner(System.in);
		System.out.print("\nEnter File name: ");
		String str=scr.next();
		FileInputStream f=new FileInputStream(str);
		System.out.println("\nContents of the file are");
		int n=f.available();
		System.out.print(j+": ");
		for(int i=0;i<n;i++)
		{
			ch=(char)f.read();
			System.out.print(ch);
			if(ch=='\n')
			{
				System.out.print(++j+": ");

			}

		}
	}
}
```

## WEEK:4-C

**Aim:** Write a Java program that displays the number of characters, lines and words in a text file

```
//Program to count number of lines,characters,and words in a text file
import java.util.*;
import java.io.*;
class Cfile
{
        public static void main(String args[])throws IOException
        {
                int nl=1,nw=0;
                char ch;
                Scanner scr=new Scanner(System.in);
                System.out.print("\nEnter File name: ");
                String str=scr.nextLine();
                FileInputStream f=new FileInputStream(str);
                int n=f.available();
                for(int i=0;i<n;i++)
                {
                        ch=(char)f.read();
                        if(ch=='\n')
                        nl++;
                        else if(ch==' ')
                                nw++;

                }
                System.out.println("\nNumber of lines : "+nl);
                System.out.println("\nNumber of words : "+(nl+nw));
                System.out.println("\nNumber of characters : "+n);


        }
}
```

## Week-5- A

**Aim**: Write a Java program that creates three threads. First thread displays ―Good Morning‖ every one second, the second thread displays ―Hello‖ every two seconds and the third thread displays ―Welcome every three seconds.

```
import java.io.*;
import java.util.*;
class ChildThread implements Runnable
{
        Thread t;
        ChildThread(String name)
        {
                t = new Thread(this, name);
                t.start();
        }
        public void run()
        {
                for(int i=1;i<=5;i++)
                {
                try
        {
            if(t.getName().equals("First Thread"))
            {
                Thread.sleep(1000);
                System.out.println(t.getName()+": Good Morning");
            }
            else if(t.getName().equals("Second Thread"))
            {
                Thread.sleep(2000);
                System.out.println(t.getName()+": Hello");
            }
            else
            {
                Thread.sleep(3000);
                System.out.println(t.getName()+": Welcome");
            }
        }
        catch(InterruptedException e)
        {
            System.out.println(t.getName()+" is interrupted");
        }
        }
    }
}
class ThreeThreads
{
    public static void main(String args[])
    {
        ChildThread one = new ChildThread("First Thread");
        ChildThread two = new ChildThread("Second Thread");
        ChildThread three = new ChildThread("Third Thread");
    }
}
```

## Week-5- B

**Aim:** Write a Java program that correctly implements producer consumer problem using the concept of inter thread communication.

```
// Java program to implement solution of producer
// consumer problem.

import java.util.LinkedList;

public class Threadexample {
    public static void main(String[] args)
        throws InterruptedException
    {
```

```
    // Object of a class that has both produce()
    // and consume() methods
    final PC pc = new PC();

    // Create producer thread
    Thread t1 = new Thread(new Runnable() {
        @Override
        public void run()
        {
            try {
                pc.produce();
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    });

    // Create consumer thread
    Thread t2 = new Thread(new Runnable() {
        @Override
        public void run()
        {
            try {
                pc.consume();
            }
            catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    });

    // Start both threads
    t1.start();
    t2.start();

    // t1 finishes before t2
    t1.join();
    t2.join();
}

// This class has a list, producer (adds items to list
// and consumer (removes items).
public static class PC {

    // Create a list shared by producer and consumer
    // Size of list is 2.
    LinkedList<Integer> list = new LinkedList<>();
    int capacity = 2;
```

```
// Function called by producer thread
public void produce() throws InterruptedException
{
    int value = 0;
    while (true) {
        synchronized (this)
        {
            // producer thread waits while list
            // is full
            while (list.size() == capacity)
                wait();

            System.out.println("Producer produced-"
                                + value);

            // to insert the jobs in the list
            list.add(value++);

            // notifies the consumer thread that
            // now it can start consuming
            notify();

            // makes the working of program easier
            // to understand
            Thread.sleep(1000);
        }
    }
}

// Function called by consumer thread
public void consume() throws InterruptedException
{
    while (true) {
        synchronized (this)
        {
            // consumer thread waits while list
            // is empty
            while (list.size() == 0)
                wait();

            // to retrieve the ifrst job in the list
            int val = list.removeFirst();

            System.out.println("Consumer consumed-"
                                + val);

            // Wake up producer thread
            notify();

            // and sleep
```

```
                    Thread.sleep(1000);
                }
            }
        }
    }
}
```