Program

Multithread(Concurrency)

1) Write a program to create two threads, one thread will print odd number and second thread will print even number between 1 to 20 numbers.

```
public class  ThreadEvenOdd  extends  Thread
{
    public void  run()
        {
                int  oddsum=0,evensum=0;
                Thread t = Thread.currentThread();
                String threadname=t.getName();
                if (threadname.equals("odd"))
                {
                        for (int  i=1;i<=20;i=i+2)
                        {
                                System.out.println(i);
                        }
                }
                else
                {
                        for (int  i=2;i<=20;i=i+2)
                      {
                                System.out.println(i);
                }

                }
        }
}
class test
{
        public static void  main(String[] args)
        {
                 ThreadEvenOdd t1 = new  ThreadEvenOdd();
                 ThreadEvenOdd t2 = new  ThreadEvenOdd();
                 t1.setName("odd");
                 t2.setName("even");
                 t1.start();
                 t2.start();
        }
}
```

2) Write an application that creates and starts three threads. Each thread is instantiated from the same class. It executes a loop with 10 iterations. Each iteration displays string "HELLO", sleeps for 300 milliseconds. The application waits for all the threads to complete & displays the message "Good Bye..."

```
class Thread1 extends Thread
{
	public void run()
	{	try{
			for(int i=0;i<10;i++)
			{
			  System.out.println("hello");
			  Thread.sleep(300);
			}
		  }
		catch(InterruptedException e)
		{
			System.out.println("InterruptedException");
		}
	}
}
class test
{
	public static void main(String args[])
	{
		try{
			Thread1 t1 =new Thread1();
			Thread1 t2 =new Thread1();
			Thread1 t3 =new Thread1();
			t1.start();
			t2.start();
			t3.start();

			t1.join();
			t2.join();
			t3.join();
		}
		catch(Exception e){}
		System.out.println("Good Bye...");
	}
 }
```

3) Write an application that executes two threads. One thread displays "Good Morning" every 1000 milliseconds & another thread displays "Good Afternoon" every 3000 milliseconds. Create the threads by implementing the Runnable interface.

```
class Thread1 implements Runnable
{
	public void run()
	{	try{
		for(int i=0;i<5;i++)
		{	System.out.println("Good Morning");
```

```
                Thread.sleep(1000);
            }
              }
            catch(InterruptedException e)
            {
                System.out.println("InterruptedException");
            }
        }
}
class Thread2 implements Runnable
{
        public void run()
        {
            try{
            for(int i=0;i<5;i++)
            {
                System.out.println("Good afternoon");
                Thread.sleep(3000);
            }
              }
            catch(InterruptedException e)
            {
                System.out.println("InterruptedException");
            }
        }
}
class test
{
        public static void main(String args[])
        {       Thread1 r1 =new Thread1();
            Thread t1 =new Thread(r1);
            Thread2 r2 =new Thread2();
            Thread t2 =new Thread(r2);
            t1.start();
            t2.start();
        }
}
```

4) Write a program to create two threads, one thread will print odd numbers and Second thread will print even numbers between 1 to 20 numbers.

```
class OddThread extends Thread
{
        String name;
        OddThread(String threadName)
        {
            name=threadName;
        }
        public void run()
        {
            for(int i=1;i<=20;i++)
            {
```

```
                    if(i%2!=0)
                    {
                      try
                      {
                            System.out.println(name+": "+i);
                            sleep(500);
                      }
                      catch(InterruptedException e)
                      {
                            System.out.println("The Error is:"+e);
                      }
                    }
              }
        }    }
class EvenThread extends Thread
{
        String name;
        EvenThread(String threadName)
        {
              name=threadName;
        }
        public void run()
        {
              for(int i=1;i<=20;i++)
              {
                    if(i%2==0)
                    {
                          try
                          {
                                sleep(500);
                                System.out.println(name+": "+i);
                          }
                          catch(InterruptedException e)
                          {
                                System.out.println("The Error is:"+e);
                          }
                    }
              }
        }
}
class ThreadOddEven
{
        public static void main(String[] args)
        {
              EvenThread t2=new EvenThread("EvenThread");
              t2.start();
              OddThread t1=new OddThread("OddThread");
              t1.start();
        }
}
```

5) Write a complete multi-threaded program to meet following requirements:

- Read matrix [A] m x n

- Create m number of threads

- Each thread computes summation of elements of one row, i.e. ith row of the matrix is processed by ith thread. Where 0 <= i< m.

- Print the results.

```
import java.util.*;
class demo extends Thread
{
	static int a[][];
	static int m,n;
	int t,i,j,sum=0;
	demo(int x)
	{
		t=x;
	}
	public void run()
	{
		for(i=0;i<m;i++)
		{
			if(t==i)
			{
			  for(i=0;i<n;i++)
			  {
				sum=sum+a[t][i];
			  }
			}
		}
		System.out.println("sum of row "+t+"="+sum);
	}
}
class testdemo
{
	public static void main(String s[])
	{
		int i,j;
		Scanner sc=new Scanner(System.in);
		System.out.println("enter m");
		demo.m=sc.nextInt();
		System.out.println("enter n");
		demo.n=sc.nextInt();
		demo.a=new int[demo.m][demo.n];
		for(i=0;i<demo.m;i++)
		{
			for(j=0;j<demo.n;j++)
			{
				demo.a[i][j]=sc.nextInt();
			}
		}
		demo d[]=new demo[demo.m];
```

```
                for(i=0;i<demo.m;i++)
                {
                        d[i]=new demo(i);
                        d[i].start();
                }
        }
}
```

6) Write a program to create two thread one display alphabet from a to z and other will display numbers from 1 to 100

```
class  ThreadEvenOdd  extends  Thread
{
    public void  run()
        {
                Thread t = Thread.currentThread();
                String threadname=t.getName();
                if(threadname.equals("alphabet"))
                {
                        for(int  i=97;i<=122;i++)
                        {
                                System.out.println((char)i);
                        }
                }
                else
                {
                        for (int i=1;i<=100;i++)
                        {
                                System.out.println(i);
                        }
                }
        }
}
class test
{
        public static void  main(String[] args)
        {
                ThreadEvenOdd t1 = new  ThreadEvenOdd();
                ThreadEvenOdd t2 = new  ThreadEvenOdd();
                t1.setName("alphabet");
                t2.setName("number");
                t1.start();
                t2.start();
        }
}
```

7) Write a multi-threaded program to have two producer threads, each writes (push) total 7 integer items to the same (common) stack. The producers enter into sleep state for 500 ms after writing every item. There is one consumer thread that reads (pop) from the same stack and enters into sleep state then-after for 600 ms. Assume stack size as 10. Incorporate all required conditions so that all valid items are popped only once and there is no stack overflow & underflow.

```
class procon
{
	boolean mark=false;
	int data[]= new int[7];
	int top=0,a;
	synchronized void produce(int n)
	{
		if(mark)
		{
			try
			{
				wait();
			}
			catch(Exception e)
			{
				System.out.println(e);
			}
		}
		data[top]=n;
		a=top;
		top++;
		System.out.println("Put = " + data[a]);
		mark=true;
		notify();
	}
	synchronized void consume()
	{
		if(!mark)
		{
			try
			{
				wait();
			}
			catch(Exception e)
			{
				System.out.println(e);
			}
		}
		System.out.println("Got = " + data[a]);
		top--;
		mark=false;
		notify();
	}
}
class prod extends Thread
{
	procon pc;
	prod(procon pc)
	{
		this.pc=pc;
```

```
                start();
        }
        public void run()
        {
                for(int i=0;i<=7;i++)
                        pc.produce(i);
        }
}
class cons extends Thread
{
        procon pc;
        cons(procon pc)
        {
                this.pc=pc;
                start();
        }
        public void run()
        {
                for(int i=0;i<=7;i++)
                        pc.consume();
        }
}
class th9
{
        public static void main(String args[])
        {
                procon pc = new procon();
                prod p1 = new prod(pc);
                cons c1 = new cons(pc);
        }
}
```

Input/Output

1) Write a program to check that whether the name given from command line is file or not? If it is a file then print the size of file and if it is directory then it should display the name of all files in it.

```
import java.io.*;
class test
{
	public static void main(String s[]) throws IOException
	{
		File f1 = new  File(s[0]);
		if(f1.isFile())
		{
			System.out.println("File size="+f1.length());
		}
		else if(f1.isDirectory())
		{
			System.out.println(f1.getAbsolutePath()+" is a directory");
			String s1[]=f1.list();
			System.out.println("It contains...");
			for(int i=0;i<s1.length;i++)
			{
				File f2=new File(f1.getAbsolutePath()+"/"+s1[i]);
				if(f2.isDirectory())
				{
				  System.out.println(s1[i]+" as a directory");
				}
				else
				{
				  System.out.println(s1[i]+" as a file");
				}
			}
		}
		else
		{
		  System.out.println("it is not present as file or directory");
		}
	}
}
```

2) Write a program that reads file name from user, through command line argument and displays/reads content of the text file on console.

```
import java.io.*;
class test
{
	public static void main(String args[]) throws IOException
	{
		FileInputStream f =new FileInputStream(args[0]);
		int i=0;
		while((i = f.read())!=-1)
		{
			System.out.print((char)i);
		}
		f.close();
	 }
```

```
}
```
3) Write a program to create directories (/home/abc/bcd/def/ghi/jkl) in the home directory /home/abc and list the files and directories showing file/directory,file size. Read-write-execute permissions. Write destructor to destroy the data of a class.
```
import java.io.*;
class test
{
	public static void main(String[] args) throws Exception
	{
		File f = new File("D:/home/abc/bcd/def/ghi/jkl");
		f.mkdirs();
		File f1 = new File("D:/home/abc/s1.txt");
		File f2 = new File("D:/home/abc/s2.txt");
		File f3 = new File("D:/home/abc/s3.txt");
		f1.createNewFile();
		f2.createNewFile();
		f3.createNewFile();

		File obj = new File("D:/home/abc");
		File arr[] = obj.listFiles();

		int i;
		for(i=0;i<arr.length;i++)
		{
			System.out.println("Name="+arr[i].getName());
			System.out.println("Length="+arr[i].length());
			System.out.println("Directory??="+arr[i].isDirectory());
			System.out.println("File??="+arr[i].isFile());
			System.out.println("CanRead??="+arr[i].canRead());
			System.out.println("CanWrite??="+arr[i].canWrite());
			System.out.println("\n");
		}
	}
}
```
4) Write a program using FileInputStream, BufferedInputStream, FileOutputStream, BufferedOutputStream to copy Content of one file File1.txt into another file File2.txt.
```
import java.io.*;
class FileCopy
{
	public static void main(String[] args) throws IOException
	{
		int x;
		FileInputStream fis=null;
		FileOutputStream fos=null;
		fis=new FileInputStream("src/Programs/SourceFile.txt");
		fos=new FileOutputStream("src/Programs/DestinationFile.txt");
		BufferedInputStream bis=new BufferedInputStream(fis);
		BufferedOutputStream bos=new BufferedOutputStream(fos);
		do
		{
```

```
                x = bis.read();
                if(x != -1)
                bos.write((char) x);
            } while(x != -1);
            System.out.println("File Successfully Copied!!!");
            bis.close();
            bos.close();
            fis.close();
            fos.close();
        }
}
```

5) Write a program to replace all "word1" by "word2" from a file1, and output is written to file2 file and display the no. of replacement.

```
import java.io.*;
class test4
{
        public static void main(String arg[])throws IOException
        {
            FileReader fr=new FileReader(arg[0]);
            BufferedReader br=new BufferedReader(fr);
            String s2,s4="",s3[];
            int word=0;
            while((s2=br.readLine())!=null)
            {
                s3=s2.split(" ");
                s4=s4+s2+System.lineSeparator();
                for(int i=0;i<s3.length;i++)
                {
                    if(s3[i].equals(arg[1]))
                    {
                      word++;
                    }
                }
            }
            System.out.println("Total no. of  Replace word="+word);
            String s5;
            s5=s4.replaceAll(arg[1],arg[2]);
            FileWriter fw=new FileWriter(arg[3]);
            PrintWriter pw=new PrintWriter(fw);
            pw.println(s5);
            fw.close();
            pw.close();
        }
}
```

6) Write a program that takes input for filename and search word from command line arguments and checks whether that file exists or not. If exists, the program will display those lines from a file that contains given search word.

```
import java.io.*;
class test4
{
```

```
	public static void main(String arg[])throws IOException
	{
	   File f1=new File(arg[0]);
	   if(f1.exists())
	   {
		FileReader fr=new FileReader(f1);
		BufferedReader br=new BufferedReader(fr);
		String s1=arg[1];
		String s2,s3[];
		while((s2=br.readLine())!=null)
		{
			s3=s2.split(" ");
			for(int i=0;i<s3.length;i++)
			{
				if(s3[i].equals(s1))
				{
					System.out.println(s2);
					break;
				}
			}
		}
	   }
	   else
	   {
		System.out.println("File not exists");
	   }
	}
}
```

7) Write a program to sort the one file numbers to another.for example one file contain the unsorted number separated by line and write the another filr with sorted number.

```
import java.io.*;
class test5
{
	public static void main(String arg[])throws IOException
	{
		FileReader fr=new FileReader(arg[0]);
		BufferedReader br=new BufferedReader(fr);
		String s1=arg[1];
		String s2;
		int i=0;
		int x[]=new int[5];
		while((s2=br.readLine())!=null)
		{
			x[i]=Integer.parseInt(s2);
			i++;
		}
		Arrays.sort(x);
		FileWriter fw=new FileWriter(s1);
		PrintWriter pw=new PrintWriter(fw);
		for(i=0;i<x.length;i++)
```

```
                {
                        pw.println(x[i]);
                }
                fw.close();
                pw.close();
        }
}
```

8) Write an application that reads a file and counts the number of occurrences of digit 5. Supply the file name as a command-line argument.

```
import java.io.*;
class test
{
        public static void main(String arg[])throws IOException
        {
                FileReader fr=new FileReader(arg[0]);
                BufferedReader br=new BufferedReader(fr);
                String s1;
                int count=0;
                while((s1=br.readLine())!=null)
                {
                        for(int i=0;i<s1.length();i++)
                        {
                                if(s1.charAt(i)=='5')
                                {
                                        count++;
                                }
                        }
                }
                System.out.println("Total no. of occurrences of 5="+count);
        }
}
```

9)Write a program to display the bytes of a file in reverse sequence. Provide the name of the file as a command line argument. (Use RandomAccessFile).

```
import java.io.*;
class test
{
        public static void main(String args[])throws IOException
        {
                RandomAccessFile raf =new RandomAccessFile(args[0],"rw");
                long pos =raf.length();
                for(int i=pos-1;i>=0;i--)
                {
                        raf.seek(i);
                        byte b= raf.readByte();
                        System.out.print((char)b);
                }
        }     }
```

10) Write a java program which read numbers from number.txt file and store even number to even.txt and odd number into odd.txt file.

```
import java.io.*;
class test
{
	public static void main(String s[]) throws IOException
	{
		FileReader fr=new FileReader("number.txt");
		BufferedReader br=new BufferedReader(fr);

		FileWriter fw1=new FileWriter("even.txt");
		PrintWriter pw1=new PrintWriter(fw1);
		FileWriter fw2=new FileWriter("odd.txt");
		PrintWriter pw2=new PrintWriter(fw2);
		String s1;
		int n;
		while((s1=br.readLine())!=null)
		{
			n=Integer.parseInt(s1);
			if(n%2==0)
			{
			  pw1.println(n);
			}
			else
			{
			  pw2.println(n);
			}
		}
		pw1.close();
		pw2.close();
	}
}
```