```
package applet;
import java.applet.*;
import java.awt.*;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;
public class MyApplet extends Applet implements MouseListener, MouseMotionListener
{
    String msg = "";
    public void init()
    {
        addMouseListener(this);
        addMouseMotionListener(this);
    }
    public void mouseExited(MouseEvent m)
    {
        msg = "Mouse Exited";
        repaint();
    }
    public void mouseReleased(MouseEvent m)
    {
        msg = "Mouse Released";
        repaint();
    }
    public void mouseEntered(MouseEvent m)
    {
    }
    public void mouseDragged(MouseEvent m)
    {
    }
    public void mousePressed(MouseEvent m)
    {
    }
    public void mouseMoved(MouseEvent m)
    {
        msg = "Mouse Moved";
        repaint();
    }
    public void mouseClicked(MouseEvent m)
    {
        msg = "Mouse Clicked";
        repaint();
    }
```

```
    public void paint(Graphics g)
    {
        Font myfont = new Font("sans-serif",Font.BOLD,18);
        g.setFont(myfont);
        g.setColor(Color.GREEN);
        g.drawString(msg,100,100);
    }
}

package Searching;
public class SelectionSort
{
    static void sort(int[] arr)
    {
        for(int i=0;i<arr.length-1;i++)
        {
            int index = i;
            for(int j = i+1;j<arr.length;j++)
            {
                if(arr[j]<arr[index])
                {
                    index = j;
                }
            }
            int temp = arr[i];
            arr[i] = arr[index];
            arr[index] = temp;
        }
    }
    public static void main(String[] args)
    {
        int[] arr={13,2,1,-1,234,1231231,987};
        sort(arr);
        System.out.print("After sorting : ");
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package windows.Adapter;
import java.awt.*;
public class MyFrame extends Frame
```

```
{
    private String msg = "Welcome";
    private String msg1 = "Mouse Location : ";
    private int x = 0;
    private int y = 0;
    MyFrame()
    {
        addMouseListener(new MyMouseAdapter(this));
        addWindowListener(new MyWindowAdapter(this));
        addMouseMotionListener(new MyMouseMotionAdapter(this));
        setVisible(true);
        setSize(300,300);
    }
    public void setX(int x)
    {
        this.x = x;
    }
    public void setY(int y)
    {
        this.y = y;
    }
    public void setMsg(String msg)
    {
        this.msg = msg;
    }
    public void paint(Graphics g)
    {
        g.drawString(msg,100,100);
        g.drawString(msg1+x+","+y,100,130);
    }
    public static void main(String[] args)
    {
        MyFrame ob = new MyFrame();
    }
}

package windows.Adapter;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
public class MyMouseAdapter extends MouseAdapter
{
    MyFrame myframe;
    public MyMouseAdapter(MyFrame myframe)
    {
```

```
            this.myframe = myframe;
        }
        public void mouseClicked(MouseEvent e)
        {
            myframe.setMsg("Mouse Clicked");
            myframe.repaint();
        }
        public void mouseEntered(MouseEvent e)
        {
            myframe.setMsg("Mouse Entered");
            myframe.repaint();
        }
        public void mouseExited(MouseEvent e)
        {
            myframe.setMsg("Mouse Exited");
            myframe.repaint();
        }
}

package windows.TextInput;
import java.awt.*;
import java.awt.event.*;
public class TextInput extends Frame implements ActionListener
{
        private Label l1,l2;
        private TextField t1,t2;
        Button b1;
        public void actionPerformed(ActionEvent e)
        {
            StringBuffer str = new StringBuffer(t1.getText());
            str = str.reverse();
            String temp = str.toString();
            temp = temp.toUpperCase();
            t2.setText(temp);
        }
       TextInput()
        {
            setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
            l1 = new Label("String :         ");
            l2 = new Label("Reverse :    ");
            t1 = new TextField(35);
            t2 = new TextField(35);
            b1 = new Button("Show");
            b1.addActionListener(this);
```

```
        add(l1);
        add(t1);
        add(l2);
        add(t2);
        add(b1);
        setSize(400,200);
        setVisible(true);
    }
    public static void main(String[] args)
    {
        TextInput a = new TextInput();
    }
}

package Sorting;
public class QuickSort
{

    static int part(int[] arr,int start,int end)
    {
        int i = start+1;
        int j = end;
        int pivot = arr[start];
        while(i<=j)
        {
            while((i<=end)&&(arr[i]<pivot))
            {
                i++;
            }
            while((j>start)&&(arr[j]>pivot))
            {
                j--;
            }
            if(i<j)
            {
                int temp = arr[j];
                arr[j] = arr[i];
                arr[i] = temp;
            }
        }
        arr[start] = arr[j];
        arr[j] = pivot;
        return j;
    }
```

```
    static void sort(int[] arr,int start,int end)
    {
        if(start<end)
        {
            int pivot = part(arr,start,end);
            sort(arr,start,pivot-1);
            sort(arr,pivot+1,end);
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {6,5,4,3,2,1};
        sort(arr,0,arr.length-1);
        for(int i = 0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Sorting;
public class HeapSort
{
    static void heapify(int[] arr,int i,int heapsize)
    {
        int left = i*2+1;
        int right = i*2+2;
        int max = i;
        if((left<heapsize)&&(arr[left]>arr[max]))
        {
            max = left;
        }
        if((right<heapsize)&&(arr[right]>arr[max]))
        {
            max = right;
        }
        if(max!=i)
        {
            int temp = arr[i];
            arr[i] = arr[max];
            arr[max] = temp;
            heapify(arr,max,heapsize);
        }
    }
```

```
    static void buildHeap(int[] arr)
    {
        int top = arr.length/2-1;
        for(int i = top;i>=0;i--)
        {
            heapify(arr,i,arr.length);
        }
    }
    static void sort(int[] arr)
    {
        buildHeap(arr);
        System.out.print("\nAfter building heap : ");
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
        System.out.print("\n\n");
        int heapsize = arr.length;
        for(int i=arr.length-1;i>0;i--)
        {
            int temp = arr[i];
            arr[i] = arr[0];
            arr[0] = temp;
            heapsize--;
            heapify(arr,0,heapsize);
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {12,3,44,-1,3,123,333,123123,-12312};
        sort(arr);
        for(int i = 0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Searching;
public class LinearSearch
{
    static boolean search(int[] arr,int value)
    {
        boolean flag = false;
```

```
            for(int i = 0;i<arr.length;i++)
            {
                if(arr[i]==value)
                {
                    flag = true;
                    break;
                }
            }
            return flag;
        }
        public static void main(String[] args)
        {
            int[] arr = {123,2,34,12,5,6,9};
            boolean var = search(arr,2);
            if(var)
            {
                System.out.print("Value found");
            }
            else
            {
                System.out.print("Value not found");
            }
        }
}

package Sorting;
public class HeapSort
{
        static void heapify(int[] arr,int i,int heapsize)
        {
            int left = i*2+1;
            int right = i*2+2;
            int max = i;
            if((left<heapsize)&&(arr[left]>arr[max]))
            {
                max = left;
            }
            if((right<heapsize)&&(arr[right]>arr[max]))
            {
                max = right;
            }
            if(max!=i)
            {
                int temp = arr[i];
```

```
                arr[i] = arr[max];
                arr[max] = temp;
                heapify(arr,max,heapsize);
            }
        }
        static void buildHeap(int[] arr)
        {
            int top = arr.length/2-1;
            for(int i = top;i>=0;i--)
            {
                heapify(arr,i,arr.length);
            }
        }
        static void sort(int[] arr)
        {
            buildHeap(arr);
            System.out.print("\nAfter building heap : ");
            for(int i =0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
            System.out.print("\n\n");
            int heapsize = arr.length;
            for(int i=arr.length-1;i>0;i--)
            {
                int temp = arr[i];
                arr[i] = arr[0];
                arr[0] = temp;
                heapsize--;
                heapify(arr,0,heapsize);
            }
        }
        public static void main(String[] args)
        {
            int[] arr = {12,3,44,-1,3,123,333,123123,-12312};
            sort(arr);
            for(int i = 0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
        }
}
```

```
package windows.Adapter;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
public class MyWindowAdapter extends WindowAdapter
{
	MyFrame myframe;
	MyWindowAdapter(MyFrame myframe)
	{
		this.myframe = myframe;
	}
	public void windowClosing(WindowEvent e)
	{
		myframe.setVisible(false);
	}
}

public class Hello
{
public static void main (String args[])
{
System.out.println("hello");
}
}

package windows;
import javafx.scene.control.CheckBox;
import java.awt.*;
import java.awt.event.*;
public class MyClass extends Frame implements ItemListener
{
	String msg1 = "Windows : ";
	String msg2 = "False";
	String msg3 = "Linux : ";
	String msg4 = "False";
	Checkbox c1,c2;
	MyClass()
	{
		setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
		c1 = new Checkbox("Windows");
		c2 = new Checkbox("Linux");
		add(c1);
		add(c2);
		addWindowListener(new WindowAdapter()
		{
```

```
                public void windowClosing(WindowEvent e)
                {
                    setVisible(false);
                }
            });
            c1.addItemListener(this);
            c2.addItemListener(this);
            setSize(300,300);
            setVisible(true);
        }
        public void paint(Graphics g)
        {
            g.drawString(msg1,100,100);
            g.drawString(msg2,180,100);
            g.drawString(msg3,100,140);
            g.drawString(msg4,180,140);
        }
        public void itemStateChanged(ItemEvent e)
        {
            if(c1.getState())
            {
                msg2 = "True";
            }
            else
            {
                msg2 = "False";
            }
            if(c2.getState())
            {
                msg4 = "True";
            }
            else
            {
                msg4 = "False";
            }
            repaint();
        }
        public static void main(String[] args)
        {
            MyClass m = new MyClass();
        }
}
```

```
package AdjacencyList;
import java.util.Scanner;
public class MyClass
{
	public static void main(String[] args)
	{
		int n;
		Scanner in = new Scanner(System.in);
		System.out.print("How many vertices : ");
		n = in.nextInt();
		MyList[] list = new MyList[n];
		for(int i=0;i<n;i++)
		{
			list[i] = new MyList();
		}
		for(int i=0;i<n;i++)
		{
			int TotalVertices;
			System.out.print("\nEnter no of adjacent vertices to "+i+" : ");
			TotalVertices = in.nextInt();
			for(int j=0;j<TotalVertices;j++)
			{
				int value;
				System.out.print("\nEnter adjacent vertix : ");
				value = in.nextInt();
				list[i].insert(value);
			}
		}
		MyQueue result = new MyQueue();
		MyList.dfs_dir(list,0,result);
		result.printQ();
	   // System.out.print("\n\nBFS IS : ");
	   // MyList.bfs(list);
	}
}

package windows.Adapter;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionAdapter;
public class MyMouseMotionAdapter extends MouseMotionAdapter
{
	MyFrame myframe;
	MyMouseMotionAdapter(MyFrame myframe)
	{
```

```
            this.myframe = myframe;
        }
        public void mouseMoved(MouseEvent e)
        {
            myframe.setX(e.getX());
            myframe.setY(e.getY());
            myframe.repaint();
        }
        public void mouseDragged(MouseEvent e)
        {
            myframe.setX(e.getX());
            myframe.setY(e.getY());
            myframe.repaint();
        }
}

package Sorting;
public class RadixSort
{
        static int[] sort(int[] arr)
        {
            int m = 0;
            int e = 1;
            for (int i = 0; i < arr.length; i++)
            {
                if (arr[i] > 0)
                {
                    m = arr[i];
                }
            }
            while (m / e > 0)
            {
                arr = rsort(arr,e);
                e*=10;
            }
            return arr;
        }
        static int[] rsort(int[] arr,int e)
        {
            int[] temp = new int[10];
            int[] c = new int[arr.length];
            for(int i = 0;i<arr.length;i++)
            {
                temp[(arr[i]/e)%10]++;
```

```
            }
            for(int i = 1;i<10;i++)
            {
                temp[i] = temp[i-1]+temp[i];
            }
            for(int i = arr.length-1;i>=0;i--)
            {
                c[temp[(arr[i]/e)%10]-1] = arr[i];
                temp[(arr[i]/e)%10]--;
            }
            return c;
        }
        public static void main(String[] args)
        {
            int[] arr = {123,34,1,2,2,98,765};
            arr = sort(arr);
            for(int i = 0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
        }
}

package windows.Adapter;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
public class MyWindowAdapter extends WindowAdapter
{
        MyFrame myframe;
        MyWindowAdapter(MyFrame myframe)
        {
            this.myframe = myframe;
        }
        public void windowClosing(WindowEvent e)
        {
            myframe.setVisible(false);
        }
}

package Linux;
import java.lang.Runnable;
class MyClass implements Runnable
{
    public MyClass(String name)
```

```
    {
        Thread T = new Thread(this,name);
        T.start();
        System.out.println("New Thread is starting");
    }
     public void run()
     {
          try
          {
               for(int i=0;i<10;i++)
               {
                    System.out.println("New Thread : "+i);
                    Thread.sleep(10000);
               }
          }
          catch (InterruptedException E)
          {
               System.out.println("Exception found");
          }
     }
}
public class MyThread {
     public static void main(String[] args) {
          Thread T = Thread.currentThread();
          MyClass obj=new MyClass("Balraj");
          try {
               for (int i = 0; i < 10; i++) {
                    System.out.println("Main Thread : " + i);
                    T.sleep(2000);
               }
          }
          catch (InterruptedException E)
          {
             System.out.println("Exception found");
          }
     }
}

package Sorting;
public class BubbleSort
{
     static void sort(int[] arr)
     {
          for(int i=0;i<arr.length-1;i++)
```

```
        {
            for(int j=0;j<arr.length-1-i;j++)
            {
                if(arr[j]>arr[j+1])
                {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                }
            }
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {1,98,765,234,-987,2};
        sort(arr);
        for(int i=0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Searching;
public class LineraSerachRecur
{
    static boolean search(int[] arr,int start,int value)
    {
        if(start==arr.length)
        {
            return false;
        }
        else if(arr[start]==value)
        {
            return true;
        }
        else
        {
            return search(arr,start+1,value);
        }
    }
    public static void main(String[] args)
    {
        int[] arr={123,2,34,2,67,8,1};
```

```
            boolean flag = search(arr,0,1);
            if(flag)
            {
                System.out.print("Value found");
            }
            else
            {
                System.out.print("Value not found");
            }
        }
}

package AdjacencyList;
public class Node
{
        static final int WHITE = 0;
        static final int BLACK = 1;
        int value;
        int color;
        Node next;
        Node(int value)
        {
            this.value = value;
            this.color = WHITE;
            this.next = null;
        }
        void setColor(int x)
        {
            this.color = x;
        }
}

package windows.Adapter;
import java.awt.event.MouseEvent;
import java.awt.event.MouseMotionAdapter;
public class MyMouseMotionAdapter extends MouseMotionAdapter
{
        MyFrame myframe;
        MyMouseMotionAdapter(MyFrame myframe)
        {
            this.myframe = myframe;
        }
        public void mouseMoved(MouseEvent e)
        {
```

```
            myframe.setX(e.getX());
            myframe.setY(e.getY());
            myframe.repaint();
        }
        public void mouseDragged(MouseEvent e)
        {
            myframe.setX(e.getX());
            myframe.setY(e.getY());
            myframe.repaint();
        }
}

package windows.TextInput;
import java.awt.*;
import java.awt.event.*;
public class TextInput extends Frame implements ActionListener
{
        private Label l1,l2;
        private TextField t1,t2;
        Button b1;
        public void actionPerformed(ActionEvent e)
        {
            StringBuffer str = new StringBuffer(t1.getText());
            str = str.reverse();
            String temp = str.toString();
            temp = temp.toUpperCase();
            t2.setText(temp);
        }
       TextInput()
        {
            setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
            l1 = new Label("String :         ");
            l2 = new Label("Reverse :    ");
            t1 = new TextField(35);
            t2 = new TextField(35);
            b1 = new Button("Show");
            b1.addActionListener(this);
            add(l1);
            add(t1);
            add(l2);
            add(t2);
            add(b1);
            setSize(400,200);
            setVisible(true);
```

```
    }
    public static void main(String[] args)
    {
        TextInput a = new TextInput();
    }
}

package Searching;
public class LineraSerachRecur
{
    static boolean search(int[] arr,int start,int value)
    {
        if(start==arr.length)
        {
            return false;
        }
        else if(arr[start]==value)
        {
            return true;
        }
        else
        {
            return search(arr,start+1,value);
        }
    }
    public static void main(String[] args)
    {
        int[] arr={123,2,34,2,67,8,1};
        boolean flag = search(arr,0,1);
        if(flag)
        {
            System.out.print("Value found");
        }
        else
        {
            System.out.print("Value not found");
        }
    }
}

package FileHandling;
import java.io.*;
import java.util.Scanner;
public class WriteFile
```

```
{
	public static void main(String[] args) throws IOException
	{
		FileWriter fw = new FileWriter("myfile.txt");
		Scanner in = new Scanner(System.in);
		System.out.print("Enter text : ");
		String str = in.nextLine();
		while(!str.equals("stop"))
		{
			try
			{
				fw.write(str);
				fw.write("\n");
				str = in.nextLine();
			}
			catch(IOException e)
			{
			}
		}
		in.close();
		fw.close();
	}
}

package Searching;
public class InsertionSort
{
	static void sort(int[] arr)
	{
		for(int i = 1;i<arr.length;i++)
		{
			int value = arr[i];
			int j = i-1;
			while((j>=0)&&(arr[j]>value))
			{
				arr[j+1] = arr[j];
				j--;
			}
			arr[j+1]=value;
		}
	}
	public static void main(String[] args)
	{
		int[] arr={123,2,2,456,7,-123,987123,10,0};
```

```
        sort(arr);
        System.out.print("After sorting : ");
        for(int i=0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package DynamicProgramming;
public class MatrixChainMultiplication
{
    static int[][] multiply(int[] arr)
    {
        int n = arr.length-1;
        int[][] temp = new int[n][n];
        for(int i=0;i<n;i++)
        {
            temp[i][i] = 0;
        }
        for(int l =2;l<n+1;l++)
        {
            for(int i=0;i<n-l+1;i++)
            {
                int j = i+l-1;
                temp[i][j] = 9999999;
                for(int k = i;k<j;k++)
                {
                    int q = temp[i][k]+temp[k+1][j]+arr[i]*arr[k+1]*arr[j+1];
                    if(q<temp[i][j])
                    {
                        temp[i][j] = q;
                    }
                }
            }
        }
        return temp;
    }
    public static void main(String[] args)
    {
        int[] arr ={30,35,15,5,10,20,25};
        int[][] temp = multiply(arr);
        System.out.print("Total : "+temp[0][5]);
    }
```

```
}

package Sorting;
public class InsertionSort
{
	static void sort(int[] arr)
	{
		for(int i =1;i<arr.length;i++)
		{
			int value = arr[i];
			int j = i-1;
			while((j>=0)&&(arr[j]>value))
			{
				arr[j+1] = arr[j];
				j--;
			}
			arr[j+1] = value;
		}
	}
	public static void main(String[] args)
	{
		int[] arr = {123,98,32,-12,76,-9998,0};
		sort(arr);
		for(int i=0;i<arr.length;i++)
		{
			System.out.print(" "+arr[i]);
		}
	}
}

package ua.com.pahaoks.hillel.async;
import java.util.ArrayList;
import java.util.concurrent.Phaser;
public class Bus {
	private static final Phaser PHASER = new Phaser(1);//Сразу регистрируем главный
поток
	//Фазы 0 и 6 - это автобусный парк, 1 - 5 остановки
	public static void main(String[] args) throws InterruptedException {
		ArrayList<Passenger> passengers = new ArrayList<>();
		for (int i = 1; i < 5; i++) {			//Сгенерируем пассажиров на
остановках
			if ((int) (Math.random() * 2) > 0)
				passengers.add(new Passenger(i, i + 1));//Этот пассажир выходит на
следующей
```

```
                if ((int) (Math.random() * 2) > 0)
                    passengers.add(new Passenger(i, 5));       //Этот пассажир выходит
на конечной
            }
            for (int i = 0; i < 7; i++) {
                switch (i) {
                    case 0:
                        System.out.println("Автобус выехал из парка.");
                        PHASER.arrive();//В фазе 0 всего 1 участник - автобус
                        break;
                    case 6:
                        System.out.println("Автобус уехал в парк.");
                        PHASER.arriveAndDeregister();//Снимаем главный поток,
ломаем барьер
                        break;
                    default:
                        int currentBusStop = PHASER.getPhase();
                        System.out.println("Остановка № " + currentBusStop);
                        for (Passenger p : passengers)                //Проверяем, есть ли
пассажиры на остановке
                            if (p.departure == currentBusStop) {
                                PHASER.register();//Регистрируем поток, который
будет участвовать в фазах
                                p.start();              // и запускаем
                            }
                        PHASER.arriveAndAwaitAdvance();//Сообщаем о своей
готовности
                }
            }
        }
        public static class Passenger extends Thread {
            private int departure;
            private int destination;
            public Passenger(int departure, int destination) {
                this.departure = departure;
                this.destination = destination;
                System.out.println(this + " ждёт на остановке № " + this.departure);
            }
            @Override
            public void run() {
                try {
                    System.out.println(this + " сел в автобус.");
                    while (PHASER.getPhase() < destination) //Пока автобус не приедет
на нужную остановку(фазу)
```

```
                    PHASER.arriveAndAwaitAdvance();        //заявляем в каждой
фазе о готовности и ждем
                    Thread.sleep(1);
                    System.out.println(this + " покинул автобус.");
                    PHASER.arriveAndDeregister();     //Отменяем регистрацию на
нужной фазе
                } catch (InterruptedException e) {
                }
            }
            @Override
            public String toString() {
                return "Пассажир{" + departure + " -> " + destination + '}';
            }
        }
}

package Linux;
import java.lang.Runnable;
class MyClass implements Runnable
{
    public MyClass(String name)
    {
        Thread T = new Thread(this,name);
        T.start();
        System.out.println("New Thread is starting");
    }
     public void run()
     {
         try
         {
             for(int i=0;i<10;i++)
             {
                 System.out.println("New Thread : "+i);
                 Thread.sleep(10000);
             }
         }
         catch (InterruptedException E)
         {
             System.out.println("Exception found");
         }
     }
}
public class MyThread {
    public static void main(String[] args) {
```

```
            Thread T = Thread.currentThread();
            MyClass obj=new MyClass("Balraj");
            try {
                for (int i = 0; i < 10; i++) {
                    System.out.println("Main Thread : " + i);
                    T.sleep(2000);
                }
            }
            catch (InterruptedException E)
            {
               System.out.println("Exception found");
            }
        }
}

package Sorting;
public class CountSort
{
    static int[] countSort(int[] arr)
    {
        int n = 0;
        for(int i =0;i<arr.length;i++)
        {
            if(arr[i]>n)
            {
                n = arr[i];
            }
        }
        int[] temp = new int[n+1];
        for(int i=0;i<arr.length;i++)
        {
            temp[arr[i]] ++;
        }
        for(int i =1;i<n+1;i++)
        {
            temp[i] = temp[i]+temp[i-1];
        }
        int[] c = new int[arr.length];
        for(int i = arr.length-1;i>=0;i--)
        {
            c[temp[arr[i]]-1] = arr[i];
            temp[arr[i]]--;
        }
        return c;
```

```
    }
    public static void main(String[] args)
    {
        int[] arr = {100,3,2,1,34,23,1,98};
        arr = countSort(arr);
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package AdjacencyList;
public class MyQueue
{
    int start = 0;
    int end = 0;
    int[] arr = new int[100];
    void enqueue(int x)
    {
            arr[end++] = x;
    }
    int dequeue()
    {
        return arr[start++];
    }
    boolean isEmpty()
    {
        return (start==end);
    }
    boolean InQueue(int value)
    {
        boolean flag = false;
        for(int i=start;i<end;i++)
        {
            if(arr[i]==value)
            {
                flag = true;
            }
        }
        return flag;
    }
    void printQ()
    {
```

```
            for(int i=start;i<end;i++)
            {
                System.out.print(" "+this.arr[i]);
            }
        }
}

package Thread;
public class ThreadSync extends Thread
{
    public static  final int ODD = 1;
    public static final int EVEN = 2;
    int type = 0;
    Temp obj = new Temp();
    ThreadSync(int TYPE)
    {
        this.type = TYPE;
        new Thread(this).start();
    }
    public void run()
    {
        synchronized (obj)
        {
            obj.print(type);
        }
    }
    public static void main(String[] args)
    {
        ThreadSync t1 = new ThreadSync(ThreadSync.ODD);
        ThreadSync t2 = new ThreadSync(ThreadSync.EVEN);
    }
}
class Temp
{
    void print(int TYPE)
    {
        int i = 0;
        if(TYPE==ThreadSync.ODD)
        {
            i = 1;
        }
        for(;i<20;i+=2)
        {
            System.out.println(i);
```

```
            }
        }
}

package AdjacencyList;
public class Node
{
    static final int WHITE = 0;
    static final int BLACK = 1;
    int value;
    int color;
    Node next;
    Node(int value)
    {
        this.value = value;
        this.color = WHITE;
        this.next = null;
    }
    void setColor(int x)
    {
        this.color = x;
    }
}

package AdjacencyList;
public class MyList
{
    Node head;
    Node end;
    MyList()
    {
        //System.out.println("Constructor invoked");
        head = new Node(0);
        end = head;
    }
    void insert(int value)
    {
        head.value++;
        end.next = new Node(value);
        end = end.next;
    }
    void print()
    {
        Node run = this.head;
```

```
        while(run!=null)
        {
            System.out.print(" "+run.value);
            run=run.next;
        }
    }
static void bfs(MyList[] arr)
    {
        MyQueue Q = new MyQueue();
        MyQueue result = new MyQueue();
        Q.enqueue(0);
        while(!Q.isEmpty())
        {
            int u = Q.dequeue();
            result.enqueue(u);
            // Visit all the adjacent vertices of u
            for(Node v = arr[u].head.next;v!=null;v = v.next)
            {
                if(v.color==Node.WHITE)
                {
                    v.setColor(Node.BLACK);
                    Q.enqueue(v.value);
                    // Mark 'v' as visited in all the adjacency lists
                    for(int i=0;i<arr.length;i++)
                    {
                        for(Node n = arr[i].head.next;n!=null;n=n.next)
                        {
                            if (n.value == v.value)
                            {
                                n.setColor(Node.BLACK);
                            }
                        }
                    }
                }
            }
            // Mark 'u' visited in all the adjacency lists
            for(int i = 0;i<arr.length;i++)
            {
                for(Node run = arr[i].head.next;run!=null;run=run.next)
                {
                    if(run.value==u)
                    {
                        run.color=Node.BLACK;
                    }
```

```
                    }
                }
            }
            result.printQ();
        }
        static void dfs(MyList[] arr,int i,MyQueue result)
        {
            result.enqueue(i);
            for(Node j = arr[i].head.next;j!=null;j = j.next)
            {
                if(j.color==Node.WHITE)
                {
                    j.setColor(Node.BLACK);
                    dfs(arr,j.value,result);
                }
            }
        }
        static void dfs_dir(MyList[] arr,int i,MyQueue result)
        {
            result.enqueue(i);
            for(int k = 0;k<arr.length;k++)
            {
                for (Node run = arr[k].head.next; run != null; run = run.next)
                {
                    if (run.value == i)
                    {
                        run.setColor(Node.BLACK);
                    }
                }
            }
            for(Node j = arr[i].head.next;j!=null;j = j.next)
            {
                if(j.color==Node.WHITE)
                {
                    dfs_dir(arr,j.value,result);
                }
            }
        }
}

package Sorting;
public class BubbleSort
{
    static void sort(int[] arr)
```

```
    {
        for(int i=0;i<arr.length-1;i++)
        {
            for(int j=0;j<arr.length-1-i;j++)
            {
                if(arr[j]>arr[j+1])
                {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                }
            }
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {1,98,765,234,-987,2};
        sort(arr);
        for(int i=0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Thread;
public class MyThread extends Thread
{
    Thread t;
    int value;
    MyThread(int x)
    {
        value = x;
        t = new Thread(this);
        t.start();
    }
    public void run()
    {
        print(value);
    }
    synchronized void print(int x)
    {
        System.out.print("\nTable of :"+x);
        for(int i=1;i<=10;i++)
```

```
            {
                System.out.print("\n"+(x*i));
            }
        }
        public static void main(String[] args)
        {
            MyThread t1 = new MyThread(5);
            MyThread t2 = new MyThread(10);
            try
            {
                t1.t.join();
                t2.t.join();
                System.out.print("\nThis is main thread");
                for(int i =0;i<=10;i++)
                {
                    System.out.print("\n"+i*1);
                }
            }
            catch(InterruptedException e)
            {
            }
        }
}

package DynamicProgramming;
public class LargestSubsequence
{
        public static void main(String[] args)
        {
        }
}

package windows;
import javafx.scene.control.CheckBox;
import java.awt.*;
import java.awt.event.*;
public class MyClass extends Frame implements ItemListener
{
        String msg1 = "Windows : ";
        String msg2 = "False";
        String msg3 = "Linux : ";
        String msg4 = "False";
        Checkbox c1,c2;
        MyClass()
```

```
{
        setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
        c1 = new Checkbox("Windows");
        c2 = new Checkbox("Linux");
        add(c1);
        add(c2);
        addWindowListener(new WindowAdapter()
        {
                public void windowClosing(WindowEvent e)
                {
                        setVisible(false);
                }
        });
        c1.addItemListener(this);
        c2.addItemListener(this);
        setSize(300,300);
        setVisible(true);
}
public void paint(Graphics g)
{
        g.drawString(msg1,100,100);
        g.drawString(msg2,180,100);
        g.drawString(msg3,100,140);
        g.drawString(msg4,180,140);
}
public void itemStateChanged(ItemEvent e)
{
        if(c1.getState())
        {
                msg2 = "True";
        }
        else
        {
                msg2 = "False";
        }
        if(c2.getState())
        {
                msg4 = "True";
        }
        else
        {
                msg4 = "False";
        }
        repaint();
```

```
    }
    public static void main(String[] args)
    {
        MyClass m = new MyClass();
    }
}

package ua.com.pahaoks.hillel.async;
import java.util.ArrayList;
import java.util.concurrent.Phaser;
public class Bus {
    private static final Phaser PHASER = new Phaser(1);//Сразу регистрируем главный
поток
    //Фазы 0 и 6 - это автобусный парк, 1 - 5 остановки
    public static void main(String[] args) throws InterruptedException {
        ArrayList<Passenger> passengers = new ArrayList<>();
        for (int i = 1; i < 5; i++) {                   //Сгенерируем пассажиров на
остановках
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, i + 1));//Этот пассажир выходит на
следующей
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, 5));       //Этот пассажир выходит
на конечной
        }
        for (int i = 0; i < 7; i++) {
            switch (i) {
                case 0:
                    System.out.println("Автобус выехал из парка.");
                    PHASER.arrive();//В фазе 0 всего 1 участник - автобус
                    break;
                case 6:
                    System.out.println("Автобус уехал в парк.");
                    PHASER.arriveAndDeregister();//Снимаем главный поток,
ломаем барьер
                    break;
                default:
                    int currentBusStop = PHASER.getPhase();
                    System.out.println("Остановка № " + currentBusStop);
                    for (Passenger p : passengers)                //Проверяем, есть ли
пассажиры на остановке
                        if (p.departure == currentBusStop) {
                            PHASER.register();//Регистрируем поток, который
будет участвовать в фазах
```

```
                                p.start();              // и запускаем
                            }
                        PHASER.arriveAndAwaitAdvance();//Сообщаем о своей
готовности
                }
            }
        }
        public static class Passenger extends Thread {
            private int departure;
            private int destination;
            public Passenger(int departure, int destination) {
                this.departure = departure;
                this.destination = destination;
                System.out.println(this + '' ждёт на остановке № '' + this.departure);
            }
            @Override
            public void run() {
                try {
                    System.out.println(this + '' сел в автобус.'');
                    while (PHASER.getPhase() < destination) //Пока автобус не приедет
на нужную остановку(фазу)
                        PHASER.arriveAndAwaitAdvance();        //заявляем в каждой
фазе о готовности и ждем
                    Thread.sleep(1);
                    System.out.println(this + '' покинул автобус.'');
                    PHASER.arriveAndDeregister();     //Отменяем регистрацию на
нужной фазе
                } catch (InterruptedException e) {
                }
            }
            @Override
            public String toString() {
                return ''Пассажир{'' + departure + '' -> '' + destination + '}';
            }
        }
}

package ua.com.pahaoks.hillel.async;
import java.util.ArrayList;
import java.util.concurrent.Phaser;
public class Bus {
    private static final Phaser PHASER = new Phaser(1);//Сразу регистрируем главный
поток
    //Фазы 0 и 6 - это автобусный парк, 1 - 5 остановки
```

```
    public static void main(String[] args) throws InterruptedException {
        ArrayList<Passenger> passengers = new ArrayList<>();
        for (int i = 1; i < 5; i++) {                    //Сгенерируем пассажиров на
остановках
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, i + 1));//Этот пассажир выходит на
следующей
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, 5));       //Этот пассажир выходит
на конечной
        }
        for (int i = 0; i < 7; i++) {
            switch (i) {
                case 0:
                    System.out.println("Автобус выехал из парка.");
                    PHASER.arrive();//В фазе 0 всего 1 участник - автобус
                    break;
                case 6:
                    System.out.println("Автобус уехал в парк.");
                    PHASER.arriveAndDeregister();//Снимаем главный поток,
ломаем барьер
                    break;
                default:
                    int currentBusStop = PHASER.getPhase();
                    System.out.println("Остановка № " + currentBusStop);
                    for (Passenger p : passengers)                //Проверяем, есть ли
пассажиры на остановке
                        if (p.departure == currentBusStop) {
                            PHASER.register();//Регистрируем поток, который
будет участвовать в фазах
                            p.start();              // и запускаем
                        }
                    PHASER.arriveAndAwaitAdvance();//Сообщаем о своей
готовности
            }
        }
    }
    public static class Passenger extends Thread {
        private int departure;
        private int destination;
        public Passenger(int departure, int destination) {
            this.departure = departure;
            this.destination = destination;
            System.out.println(this + " ждёт на остановке № " + this.departure);
```

```
            }
            @Override
            public void run() {
                try {
                    System.out.println(this + " сел в автобус.");
                    while (PHASER.getPhase() < destination) //Пока автобус не приедет
на нужную остановку(фазу)
                        PHASER.arriveAndAwaitAdvance();        //заявляем в каждой
фазе о готовности и ждем
                    Thread.sleep(1);
                    System.out.println(this + " покинул автобус.");
                    PHASER.arriveAndDeregister();     //Отменяем регистрацию на
нужной фазе
                } catch (InterruptedException e) {
                }
            }
            @Override
            public String toString() {
                return "Пассажир{" + departure + " -> " + destination + '}';
            }
        }
}

package windows;
import java.awt.*;
import java.awt.event.*;
public class Lists extends Frame implements ItemListener
{
    String msg1 = "Selected item : ";
    String msg2 = "";
    Choice mylist;
    Lists()
    {
        setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
        mylist = new Choice();
        mylist.add("Windows");
        mylist.add("Mac");
        mylist.add("Linux");
        mylist.add("BSD");
        mylist.addItemListener(this);
        add(mylist);
        setSize(300,300);
        setVisible(true);
        addWindowListener(new WindowAdapter()
```

```
            {
                public void windowClosing(WindowEvent w)
                {
                    setVisible(false);
                }
            });
        }
        public void itemStateChanged(ItemEvent e)
        {
            msg2 = mylist.getSelectedItem();
            repaint();
        }
        public void paint(Graphics g)
        {
            g.drawString(msg1+msg2,100,100);
        }
        public static void main(String[] args)
        {
            Lists l = new Lists();
        }
}

package Sorting;
public class CountSort
{
        static int[] countSort(int[] arr)
        {
            int n = 0;
            for(int i =0;i<arr.length;i++)
            {
                if(arr[i]>n)
                {
                    n = arr[i];
                }
            }
            int[] temp = new int[n+1];
            for(int i=0;i<arr.length;i++)
            {
                temp[arr[i]] ++;
            }
            for(int i =1;i<n+1;i++)
            {
                temp[i] = temp[i]+temp[i-1];
            }
```

```
        int[] c = new int[arr.length];
        for(int i = arr.length-1;i>=0;i--)
        {
            c[temp[arr[i]]-1] = arr[i];
            temp[arr[i]]--;
        }
        return c;
    }
    public static void main(String[] args)
    {
        int[] arr = {100,3,2,1,34,23,1,98};
        arr = countSort(arr);
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package AdjacencyList;
public class MyList
{
    Node head;
    Node end;
    MyList()
    {
        //System.out.println("Constructor invoked");
        head = new Node(0);
        end = head;
    }
    void insert(int value)
    {
        head.value++;
        end.next = new Node(value);
        end = end.next;
    }
    void print()
    {
        Node run = this.head;
        while(run!=null)
        {
            System.out.print(" "+run.value);
            run=run.next;
        }
```

```
 }
static void bfs(MyList[] arr)
 {
        MyQueue Q = new MyQueue();
        MyQueue result = new MyQueue();
        Q.enqueue(0);
        while(!Q.isEmpty())
        {
              int u = Q.dequeue();
              result.enqueue(u);
              // Visit all the adjacent vertices of u
              for(Node v = arr[u].head.next;v!=null;v = v.next)
              {
                    if(v.color==Node.WHITE)
                    {
                          v.setColor(Node.BLACK);
                          Q.enqueue(v.value);
                          // Mark 'v' as visited in all the adjacency lists
                          for(int i=0;i<arr.length;i++)
                          {
                                for(Node n = arr[i].head.next;n!=null;n=n.next)
                                {
                                      if (n.value == v.value)
                                      {
                                            n.setColor(Node.BLACK);
                                      }
                                }
                          }
                    }
              }
              // Mark 'u' visited in all the adjacency lists
              for(int i = 0;i<arr.length;i++)
              {
                    for(Node run = arr[i].head.next;run!=null;run=run.next)
                    {
                          if(run.value==u)
                          {
                                run.color=Node.BLACK;
                          }
                    }
              }
        }
        result.printQ();
 }
```

```
    static void dfs(MyList[] arr,int i,MyQueue result)
    {
        result.enqueue(i);
        for(Node j = arr[i].head.next;j!=null;j = j.next)
        {
            if(j.color==Node.WHITE)
            {
                j.setColor(Node.BLACK);
                dfs(arr,j.value,result);
            }
        }
    }
    static void dfs_dir(MyList[] arr,int i,MyQueue result)
    {
        result.enqueue(i);
        for(int k = 0;k<arr.length;k++)
        {
            for (Node run = arr[k].head.next; run != null; run = run.next)
            {
                if (run.value == i)
                {
                    run.setColor(Node.BLACK);
                }
            }
        }
        for(Node j = arr[i].head.next;j!=null;j = j.next)
        {
            if(j.color==Node.WHITE)
            {
                dfs_dir(arr,j.value,result);
            }
        }
    }
}

package Sorting;
public class MergeSort
{
    static void merge(int[] arr,int start,int mid,int end)
    {
        int n = mid-start+1;
        int m = end-mid;
        int index = start;
        int[] arr1 = new int[n];
```

```
        int[] arr2 = new int[m];
        for(int i=0;i<n;i++)
        {
            arr1[i] = arr[index++];
        }
        for(int i=0;i<m;i++)
        {
            arr2[i] = arr[index++];
        }
        int i = 0;
        int j = 0;
        index = start;
        while((i<n)&&(j<m))
        {
            if(arr1[i]<arr2[j])
            {
                arr[index++] = arr1[i++];
            }
            else
            {
                arr[index++] = arr2[j++];
            }
        }
        while(i<n)
        {
            arr[index++] = arr1[i++];
        }
        while(j<m)
        {
            arr[index++] = arr2[j++];
        }
    }
    static void sort(int[] arr,int start,int end)
    {
        if(start<end)
        {
            int mid =(start+end)/2;
            sort(arr,start,mid);
            sort(arr,mid+1,end);
            merge(arr,start,mid,end);
        }
    }
    public static void main(String[] args)
    {
```

```
        int[] arr = {123,2,-123,-65,334,1,2,1};
        sort(arr,0,arr.length-1);
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Searching;
public class LinearSearch
{
    static boolean search(int[] arr,int value)
    {
        boolean flag = false;
        for(int i = 0;i<arr.length;i++)
        {
            if(arr[i]==value)
            {
                flag = true;
                break;
            }
        }
        return flag;
    }
    public static void main(String[] args)
    {
        int[] arr = {123,2,34,12,5,6,9};
        boolean var = search(arr,2);
        if(var)
        {
            System.out.print("Value found");
        }
        else
        {
            System.out.print("Value not found");
        }
    }
}

package FileHandling;
import java.io.*;
public class ReadFile
{
```

```
    public static void main(String[] args) throws IOException
    {
        FileReader fr = new FileReader("myfile.txt");
        BufferedReader br = new BufferedReader(fr);
        FileWriter fw = new FileWriter("myfile1.txt");
        BufferedWriter bw = new BufferedWriter(fw);
        String str = br.readLine();
        while(str!=null)
        {
            bw.write(str);
            bw.write("\n");
            str = br.readLine();
        }
        bw.close();
        br.close();
    }
}

package AdjacencyList;
import java.util.Scanner;
public class MyClass
{
    public static void main(String[] args)
    {
        int n;
        Scanner in = new Scanner(System.in);
        System.out.print("How many vertices : ");
        n = in.nextInt();
        MyList[] list = new MyList[n];
        for(int i=0;i<n;i++)
        {
            list[i] = new MyList();
        }
        for(int i=0;i<n;i++)
        {
            int TotalVertices;
            System.out.print("\nEnter no of adjacent vertices to "+i+" : ");
            TotalVertices = in.nextInt();
            for(int j=0;j<TotalVertices;j++)
            {
                int value;
                System.out.print("\nEnter adjacent vertix : ");
                value = in.nextInt();
                list[i].insert(value);
```

```
                }
            }
            MyQueue result = new MyQueue();
            MyList.dfs_dir(list,0,result);
            result.printQ();
           // System.out.print("\n\nBFS IS : ");
           // MyList.bfs(list);
    }
}

package Searching;
public class InsertionSort
{
    static void sort(int[] arr)
    {
        for(int i = 1;i<arr.length;i++)
        {
            int value = arr[i];
            int j = i-1;
            while((j>=0)&&(arr[j]>value))
            {
                arr[j+1] = arr[j];
                j--;
            }
            arr[j+1]=value;
        }
    }
    public static void main(String[] args)
    {
        int[] arr={123,2,2,456,7,-123,987123,10,0};
        sort(arr);
        System.out.print("After sorting : ");
        for(int i=0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package windows.Adapter;
import java.awt.*;
public class MyFrame extends Frame
{
    private String msg = "Welcome";
```

```
    private String msg1 = "Mouse Location : ";
    private int x = 0;
    private int y = 0;
    MyFrame()
    {
        addMouseListener(new MyMouseAdapter(this));
        addWindowListener(new MyWindowAdapter(this));
        addMouseMotionListener(new MyMouseMotionAdapter(this));
        setVisible(true);
        setSize(300,300);
    }
    public void setX(int x)
    {
        this.x = x;
    }
    public void setY(int y)
    {
        this.y = y;
    }
    public void setMsg(String msg)
    {
        this.msg = msg;
    }
    public void paint(Graphics g)
    {
        g.drawString(msg,100,100);
        g.drawString(msg1+x+","+y,100,130);
    }
    public static void main(String[] args)
    {
        MyFrame ob = new MyFrame();
    }
}

package windows;
import java.awt.*;
import java.awt.event.*;
public class Lists extends Frame implements ItemListener
{
    String msg1 = "Selected item : ";
    String msg2 = "";
    Choice mylist;
    Lists()
    {
```

```
        setLayout(new FlowLayout(FlowLayout.CENTER,10,10));
        mylist = new Choice();
        mylist.add("Windows");
        mylist.add("Mac");
        mylist.add("Linux");
        mylist.add("BSD");
        mylist.addItemListener(this);
        add(mylist);
        setSize(300,300);
        setVisible(true);
        addWindowListener(new WindowAdapter()
        {
            public void windowClosing(WindowEvent w)
            {
                setVisible(false);
            }
        });
    }
    public void itemStateChanged(ItemEvent e)
    {
        msg2 = mylist.getSelectedItem();
        repaint();
    }
    public void paint(Graphics g)
    {
        g.drawString(msg1+msg2,100,100);
    }
    public static void main(String[] args)
    {
        Lists l = new Lists();
    }
}

package FileHandling;
import java.io.*;
import java.util.Scanner;
public class WriteFile
{
    public static void main(String[] args) throws IOException
    {
        FileWriter fw = new FileWriter("myfile.txt");
        Scanner in = new Scanner(System.in);
        System.out.print("Enter text : ");
        String str = in.nextLine();
```

```
            while(!str.equals("stop"))
            {
                try
                {
                    fw.write(str);
                    fw.write("\n");
                    str = in.nextLine();
                }
                catch(IOException e)
                {
                }
            }
            in.close();
            fw.close();
        }
}

package DynamicProgramming;
public class MatrixChainMultiplication
{
    static int[][] multiply(int[] arr)
    {
        int n = arr.length-1;
        int[][] temp = new int[n][n];
        for(int i=0;i<n;i++)
        {
            temp[i][i] = 0;
        }
        for(int l =2;l<n+1;l++)
        {
            for(int i=0;i<n-l+1;i++)
            {
                int j = i+l-1;
                temp[i][j] = 9999999;
                for(int k = i;k<j;k++)
                {
                    int q = temp[i][k]+temp[k+1][j]+arr[i]*arr[k+1]*arr[j+1];
                    if(q<temp[i][j])
                    {
                        temp[i][j] = q;
                    }
                }
            }
        }
```

```
            return temp;
        }
        public static void main(String[] args)
        {
            int[] arr ={30,35,15,5,10,20,25};
            int[][] temp = multiply(arr);
            System.out.print("Total : "+temp[0][5]);
        }
}

package Thread;
public class ThreadSync extends Thread
{
        public static    final int ODD = 1;
        public static final int EVEN = 2;
        int type = 0;
        Temp obj = new Temp();
        ThreadSync(int TYPE)
        {
            this.type = TYPE;
            new Thread(this).start();
        }
        public void run()
        {
            synchronized (obj)
            {
                obj.print(type);
            }
        }
        public static void main(String[] args)
        {
            ThreadSync t1 = new ThreadSync(ThreadSync.ODD);
            ThreadSync t2 = new ThreadSync(ThreadSync.EVEN);
        }
}
class Temp
{
        void print(int TYPE)
        {
            int i = 0;
            if(TYPE==ThreadSync.ODD)
            {
                i = 1;
            }
```

```
        for(;i<20;i+=2)
        {
            System.out.println(i);
        }
    }
}

package Sorting;
public class SelectionSort
{
    static void sort(int[] arr)
    {
        for(int i=0;i<arr.length-1;i++)
        {
            int index = i;
            for(int j=i+1;j<arr.length;j++)
            {
                if(arr[j]<arr[index])
                {
                    index = j;
                }
            }
            int temp = arr[i];
            arr[i] = arr[index];
            arr[index] = temp;
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {123,-1,987,-23423,123,3,4,6,34};
        sort(arr);
        for(int i = 0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Sorting;
public class MergeSort
{
    static void merge(int[] arr,int start,int mid,int end)
    {
        int n = mid-start+1;
```

```
        int m = end-mid;
        int index = start;
        int[] arr1 = new int[n];
        int[] arr2 = new int[m];
        for(int i=0;i<n;i++)
        {
            arr1[i] = arr[index++];
        }
        for(int i=0;i<m;i++)
        {
            arr2[i] = arr[index++];
        }
        int i = 0;
        int j = 0;
        index = start;
        while((i<n)&&(j<m))
        {
            if(arr1[i]<arr2[j])
            {
                arr[index++] = arr1[i++];
            }
            else
            {
                arr[index++] = arr2[j++];
            }
        }
        while(i<n)
        {
            arr[index++] = arr1[i++];
        }
        while(j<m)
        {
            arr[index++] = arr2[j++];
        }
    }
    static void sort(int[] arr,int start,int end)
    {
        if(start<end)
        {
            int mid =(start+end)/2;
            sort(arr,start,mid);
            sort(arr,mid+1,end);
            merge(arr,start,mid,end);
        }
```

```
    }
    public static void main(String[] args)
    {
        int[] arr = {123,2,-123,-65,334,1,2,1};
        sort(arr,0,arr.length-1);
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package Sorting;
public class RadixSort
{
    static int[] sort(int[] arr)
    {
        int m = 0;
        int e = 1;
        for (int i = 0; i < arr.length; i++)
        {
            if (arr[i] > 0)
            {
                m = arr[i];
            }
        }
        while (m / e > 0)
        {
            arr = rsort(arr,e);
            e*=10;
        }
        return arr;
    }
    static int[] rsort(int[] arr,int e)
    {
        int[] temp = new int[10];
        int[] c = new int[arr.length];
        for(int i = 0;i<arr.length;i++)
        {
            temp[(arr[i]/e)%10]++;
        }
        for(int i = 1;i<10;i++)
        {
            temp[i] = temp[i-1]+temp[i];
```

```
            }
            for(int i = arr.length-1;i>=0;i--)
            {
                c[temp[(arr[i]/e)%10]-1] = arr[i];
                temp[(arr[i]/e)%10]--;
            }
            return c;
        }
        public static void main(String[] args)
        {
            int[] arr = {123,34,1,2,2,98,765};
            arr = sort(arr);
            for(int i = 0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
        }
}

package Searching;
public class BubbleSort
{
        static void sort(int[] arr)
        {
            for(int i =0;i<arr.length-1;i++)
            {
                for(int j = 0;j<arr.length-1-i;j++)
                {
                    if(arr[j]>arr[j+1])
                    {
                        int temp = arr[j];
                        arr[j] = arr[j+1];
                        arr[j+1] = temp;
                    }
                }
            }
        }
        public static void main(String[] args)
        {
            int[] arr={123,2,-12,234,1,56464,0,5};
            sort(arr);
            System.out.print("After sorting : ");
            for(int i = 0;i<arr.length;i++)
            {
```

```
                System.out.print(" "+arr[i]);
            }
        }
}

package Searching;
public class BubbleSort
{
    static void sort(int[] arr)
    {
        for(int i =0;i<arr.length-1;i++)
        {
            for(int j = 0;j<arr.length-1-i;j++)
            {
                if(arr[j]>arr[j+1])
                {
                    int temp = arr[j];
                    arr[j] = arr[j+1];
                    arr[j+1] = temp;
                }
            }
        }
    }
    public static void main(String[] args)
    {
        int[] arr={123,2,-12,234,1,56464,0,5};
        sort(arr);
        System.out.print("After sorting : ");
        for(int i = 0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package AdjacencyList;
import java.util.Scanner;
public class MyClass
{
    public static void main(String[] args)
    {
        int n;
        Scanner in = new Scanner(System.in);
        System.out.print("How many vertices : ");
```

```
            n = in.nextInt();
            MyList[] list = new MyList[n];
            for(int i=0;i<n;i++)
            {
                  list[i] = new MyList();
            }
            for(int i=0;i<n;i++)
            {
                  int TotalVertices;
                  System.out.print("\nEnter no of adjacent vertices to "+i+" : ");
                  TotalVertices = in.nextInt();
                  for(int j=0;j<TotalVertices;j++)
                  {
                        int value;
                        System.out.print("\nEnter adjacent vertix : ");
                        value = in.nextInt();
                        list[i].insert(value);
                  }
            }
            MyQueue result = new MyQueue();
            MyList.dfs_dir(list,0,result);
            result.printQ();
           // System.out.print("\n\nBFS IS : ");
           // MyList.bfs(list);
      }
}

package applet;
import java.applet.*;
import java.awt.*;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.event.MouseMotionListener;
public class MyApplet extends Applet implements MouseListener, MouseMotionListener
{
      String msg = "";
      public void init()
      {
            addMouseListener(this);
            addMouseMotionListener(this);
      }
      public void mouseExited(MouseEvent m)
      {
            msg = "Mouse Exited";
```

```
            repaint();
        }
        public void mouseReleased(MouseEvent m)
        {
            msg = "Mouse Released";
            repaint();
        }
        public void mouseEntered(MouseEvent m)
        {
        }
        public void mouseDragged(MouseEvent m)
        {
        }
        public void mousePressed(MouseEvent m)
        {
        }
        public void mouseMoved(MouseEvent m)
        {
            msg = "Mouse Moved";
            repaint();
        }
        public void mouseClicked(MouseEvent m)
        {
            msg = "Mouse Clicked";
            repaint();
        }
    public void paint(Graphics g)
    {
        Font myfont = new Font("sans-serif",Font.BOLD,18);
        g.setFont(myfont);
        g.setColor(Color.GREEN);
        g.drawString(msg,100,100);
    }
}

package ua.com.pahaoks.hillel.async;
import java.util.ArrayList;
import java.util.concurrent.Phaser;
public class Bus {
    private static final Phaser PHASER = new Phaser(1);//Сразу регистрируем главный
поток
    //Фазы 0 и 6 - это автобусный парк, 1 - 5 остановки
    public static void main(String[] args) throws InterruptedException {
        ArrayList<Passenger> passengers = new ArrayList<>();
```

```
        for (int i = 1; i < 5; i++) {                    //Сгенерируем пассажиров на
остановках
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, i + 1));//Этот пассажир выходит на
следующей
            if ((int) (Math.random() * 2) > 0)
                passengers.add(new Passenger(i, 5));       //Этот пассажир выходит
на конечной
        }
        for (int i = 0; i < 7; i++) {
            switch (i) {
                case 0:
                    System.out.println("Автобус выехал из парка.");
                    PHASER.arrive();//В фазе 0 всего 1 участник - автобус
                    break;
                case 6:
                    System.out.println("Автобус уехал в парк.");
                    PHASER.arriveAndDeregister();//Снимаем главный поток,
ломаем барьер
                    break;
                default:
                    int currentBusStop = PHASER.getPhase();
                    System.out.println("Остановка № " + currentBusStop);
                    for (Passenger p : passengers)                //Проверяем, есть ли
пассажиры на остановке
                        if (p.departure == currentBusStop) {
                            PHASER.register();//Регистрируем поток, который
будет участвовать в фазах
                            p.start();              // и запускаем
                        }
                    PHASER.arriveAndAwaitAdvance();//Сообщаем о своей
готовности
            }
        }
    }
    public static class Passenger extends Thread {
        private int departure;
        private int destination;
        public Passenger(int departure, int destination) {
            this.departure = departure;
            this.destination = destination;
            System.out.println(this + " ждёт на остановке № " + this.departure);
        }
        @Override
```

```
        public void run() {
            try {
                System.out.println(this + " сел в автобус.");
                while (PHASER.getPhase() < destination) //Пока автобус не приедет
на нужную остановку(фазу)
                    PHASER.arriveAndAwaitAdvance();        //заявляем в каждой
фазе о готовности и ждем
                Thread.sleep(1);
                System.out.println(this + " покинул автобус.");
                PHASER.arriveAndDeregister();     //Отменяем регистрацию на
нужной фазе
            } catch (InterruptedException e) {
            }
        }
        @Override
        public String toString() {
            return "Пассажир{" + departure + " -> " + destination + '}';
        }
    }
}

package Searching;
public class SelectionSort
{
    static void sort(int[] arr)
    {
        for(int i=0;i<arr.length-1;i++)
        {
            int index = i;
            for(int j = i+1;j<arr.length;j++)
            {
                if(arr[j]<arr[index])
                {
                    index = j;
                }
            }
            int temp = arr[i];
            arr[i] = arr[index];
            arr[index] = temp;
        }
    }
    public static void main(String[] args)
    {
        int[] arr={13,2,1,-1,234,1231231,987};
```

```
        sort(arr);
        System.out.print("After sorting : ");
        for(int i =0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package FileHandling;
import java.io.*;
public class ReadFile
{
    public static void main(String[] args) throws IOException
    {
        FileReader fr = new FileReader("myfile.txt");
        BufferedReader br = new BufferedReader(fr);
        FileWriter fw = new FileWriter("myfile1.txt");
        BufferedWriter bw = new BufferedWriter(fw);
        String str = br.readLine();
        while(str!=null)
        {
            bw.write(str);
            bw.write("\n");
            str = br.readLine();
        }
        bw.close();
        br.close();
    }
}

package Sorting;
public class SelectionSort
{
    static void sort(int[] arr)
    {
        for(int i=0;i<arr.length-1;i++)
        {
            int index = i;
            for(int j=i+1;j<arr.length;j++)
            {
                if(arr[j]<arr[index])
                {
                    index = j;
```

```
                    }
                }
                int temp = arr[i];
                arr[i] = arr[index];
                arr[index] = temp;
            }
        }
        public static void main(String[] args)
        {
            int[] arr = {123,-1,987,-23423,123,3,4,6,34};
            sort(arr);
            for(int i = 0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
        }
}

package Sorting;
public class InsertionSort
{
        static void sort(int[] arr)
        {
            for(int i =1;i<arr.length;i++)
            {
                int value = arr[i];
                int j = i-1;
                while((j>=0)&&(arr[j]>value))
                {
                    arr[j+1] = arr[j];
                    j--;
                }
                arr[j+1] = value;
            }
        }
        public static void main(String[] args)
        {
            int[] arr = {123,98,32,-12,76,-9998,0};
            sort(arr);
            for(int i=0;i<arr.length;i++)
            {
                System.out.print(" "+arr[i]);
            }
        }
```

```
}

package windows.Adapter;
import java.awt.event.MouseAdapter;
import java.awt.event.MouseEvent;
public class MyMouseAdapter extends MouseAdapter
{
	MyFrame myframe;
	public MyMouseAdapter(MyFrame myframe)
	{
		this.myframe = myframe;
	}
	public void mouseClicked(MouseEvent e)
	{
		myframe.setMsg("Mouse Clicked");
		myframe.repaint();
	}
	public void mouseEntered(MouseEvent e)
	{
		myframe.setMsg("Mouse Entered");
		myframe.repaint();
	}
	public void mouseExited(MouseEvent e)
	{
		myframe.setMsg("Mouse Exited");
		myframe.repaint();
	}
}

package DynamicProgramming;
public class LargestSubsequence
{
	public static void main(String[] args)
	{
	}
}

package Thread;
public class MyThread extends Thread
{
	Thread t;
	int value;
	MyThread(int x)
	{
```

```
        value = x;
        t = new Thread(this);
        t.start();
    }
    public void run()
    {
        print(value);
    }
    synchronized void print(int x)
    {
        System.out.print("\nTable of :"+x);
        for(int i=1;i<=10;i++)
        {
            System.out.print("\n"+(x*i));
        }
    }
    public static void main(String[] args)
    {
        MyThread t1 = new MyThread(5);
        MyThread t2 = new MyThread(10);
        try
        {
            t1.t.join();
            t2.t.join();
            System.out.print("\nThis is main thread");
            for(int i =0;i<=10;i++)
            {
                System.out.print("\n"+i*1);
            }
        }
        catch(InterruptedException e)
        {
        }
    }
}

package Sorting;
public class QuickSort
{

    static int part(int[] arr,int start,int end)
    {
        int i = start+1;
        int j = end;
```

```
        int pivot = arr[start];
        while(i<=j)
        {
            while((i<=end)&&(arr[i]<pivot))
            {
                i++;
            }
            while((j>start)&&(arr[j]>pivot))
            {
                j--;
            }
            if(i<j)
            {
                int temp = arr[j];
                arr[j] = arr[i];
                arr[i] = temp;
            }
        }
        arr[start] = arr[j];
        arr[j] = pivot;
        return j;
    }
    static void sort(int[] arr,int start,int end)
    {
        if(start<end)
        {
            int pivot = part(arr,start,end);
            sort(arr,start,pivot-1);
            sort(arr,pivot+1,end);
        }
    }
    public static void main(String[] args)
    {
        int[] arr = {6,5,4,3,2,1};
        sort(arr,0,arr.length-1);
        for(int i = 0;i<arr.length;i++)
        {
            System.out.print(" "+arr[i]);
        }
    }
}

package AdjacencyList;
public class MyQueue
```

```
{
	int start = 0;
	int end = 0;
	int[] arr = new int[100];
	void enqueue(int x)
	{
		arr[end++] = x;
	}
	int dequeue()
	{
	    return arr[start++];
	}
	boolean isEmpty()
	{
		return (start==end);
	}
	boolean lnQueue(int value)
	{
		boolean flag = false;
		for(int i=start;i<end;i++)
		{
			if(arr[i]==value)
			{
				flag = true;
			}
		}
		return flag;
	}
	void printQ()
	{
		for(int i=start;i<end;i++)
		{
			System.out.print(" "+this.arr[i]);
		}
	}
}
```