**1 Write a program to add and multiply complex no?**

import java.util.Scanner;

class Complex

{

float real;

float img;

Complex()

{

real = 0;

img = 0;

}

Complex(float r, float i )

{

real = r;

img = i;

}

Complex get\_input(Complex a)

{

Scanner input = new Scanner(System.in);

Complex n= new Complex();

System.out.println("Enter the real no. : ");

real = input.nextFloat();

System.out.println("Enter the img no. : ");

img = input.nextFloat();

return (n);

}

void display()

{

System.out.println("\n\tThe number is "+real +"+i"+img);

}

Complex mult( Complex c1,Complex c2)

{

Complex c3 = new Complex();

c3.real = c1.real\*c2.real-c1.img\*c2.img;

c3.img = c1.img\*c2.real + c1.real\*c2.img;

return (c3);

}

Complex add( Complex c1,Complex c2)

{

Complex c3 = new Complex();

c3.real = c1.real+c2.real;

c3.img = c1.img+c2.img;

return (c3);

}

public static void main(String []args)

{ Complex n1 = new Complex();

Complex n2= new Complex();

Complex n3= new Complex();

n1.get\_input(n1);

n2.get\_input(n2);

n1.display();

n2.display();

System.out.println("\n\tAddition ");

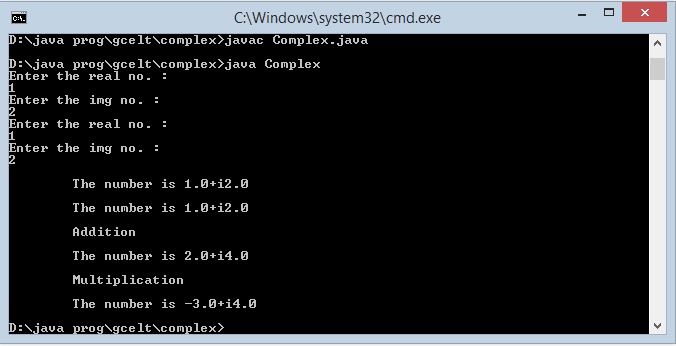
n3 = n3.add(n1, n2);

n3.display();

System.out.println("\n\tMultiplication ");

n3 = n3.mult(n1, n2);

n3.display();}}

o/p=**2 write a program to show current date?**

import java.util.Date;

public class Time {

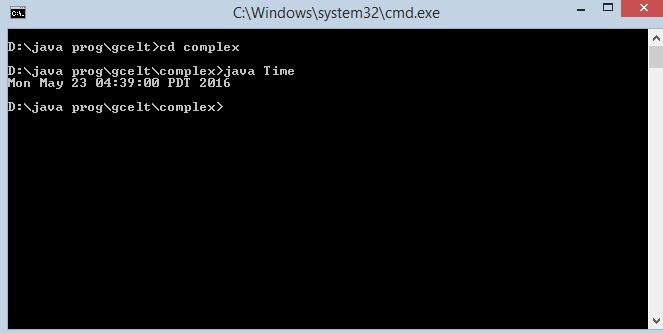
public static void main(String args[]) {

// Instantiate a Date object

Date date = new Date();

// display time and date using toString()

System.out.println(date.toString());}}

o/p=

**3 write a program to bubble sort implementation?**

public class BubbleSort {

public static void main(String[] args) {

int intArray[] = new int[]{5,90,35,45,150,3};

System.out.println("Array Before Bubble Sort");

for(int i=0; i < intArray.length; i++){

System.out.print(intArray[i] + " ");

}

bubbleSort(intArray);

System.out.println("");

System.out.println("Array After Bubble Sort");

for(int i=0; i < intArray.length; i++){

System.out.print(intArray[i] + " ");

}

}

private static void bubbleSort(int[] intArray) {

int n = intArray.length;

int temp = 0;

for(int i=0; i < n; i++){

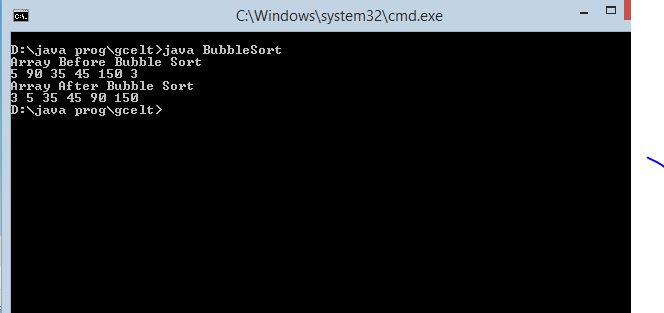
for(int j=1; j < (n-i); j++){

if(intArray[j-1] > intArray[j]){

temp = intArray[j-1];

intArray[j-1] = intArray[j];

intArray[j] = temp;}}}}}}

op/=

**4 write a program to implement 3 thread?**

class RunnableDemo implements Runnable {

private Thread t;

private String threadName;

RunnableDemo( String name){

threadName = name;

System.out.println("Creating " + threadName );

}

public void run() {

System.out.println("Running " + threadName );

try {

for(int i = 3; i > 0; i--) {

System.out.println("Thread: " + threadName + ", " + i);

// Let the thread sleep for a while.

Thread.sleep(50);

}

} catch (InterruptedException e) {

System.out.println("Thread " + threadName + " interrupted.");

}

System.out.println("Thread " + threadName + " exiting.");

}

public void start ()

{

System.out.println("Starting " + threadName );

if (t == null)

{

t = new Thread (this, threadName);

t.start ();

}

}

}

public class TestThread {

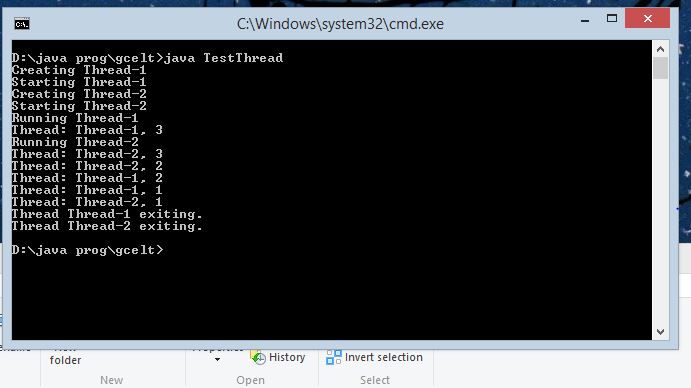
public static void main(String args[]) {

RunnableDemo R1 = new RunnableDemo( "Thread-1");

R1.start();

RunnableDemo R2 = new RunnableDemo( "Thread-2");

R2.start();}}

o/p=

**5 write a program to multiply two matrix?**

import java.util.Scanner;

class MatrixMultiplication

{

public static void main(String args[])

{

int m, n, p, q, sum = 0, c, d, k;

Scanner in = new Scanner(System.in);

System.out.println("Enter the number of rows and columns of first matrix");

m = in.nextInt();

n = in.nextInt();

int first[][] = new int[m][n];

System.out.println("Enter the elements of first matrix");

for ( c = 0 ; c < m ; c++ )

for ( d = 0 ; d < n ; d++ )

first[c][d] = in.nextInt();

System.out.println("Enter the number of rows and columns of second matrix");

p = in.nextInt();

q = in.nextInt();

if ( n != p )

System.out.println("Matrices with entered orders can't be multiplied with each other.");

else

{

int second[][] = new int[p][q];

int multiply[][] = new int[m][q];

System.out.println("Enter the elements of second matrix");

for ( c = 0 ; c < p ; c++ )

for ( d = 0 ; d < q ; d++ )

second[c][d] = in.nextInt();

for ( c = 0 ; c < m ; c++ )

{

for ( d = 0 ; d < q ; d++ )

{

for ( k = 0 ; k < p ; k++ )

{

sum = sum + first[c][k]\*second[k][d];

}

multiply[c][d] = sum;

sum = 0;

}

}

System.out.println("Product of entered matrices:-");

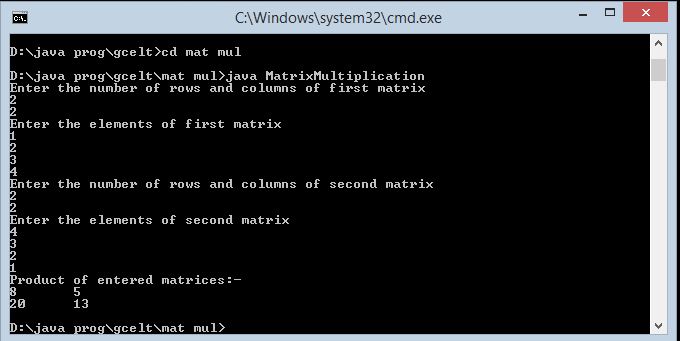
for ( c = 0 ; c < m ; c++ )

{

for ( d = 0 ; d < q ; d++ )

System.out.print(multiply[c][d]+"\t");

System.out.print("\n"); } } } }

o/p=

6 write a program to implement color band?

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.EventQueue;

import java.awt.GridBagConstraints;

import java.awt.GridBagLayout;

import java.awt.Insets;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.JScrollPane;

import javax.swing.JSlider;

import javax.swing.Timer;

import javax.swing.UIManager;

import javax.swing.UnsupportedLookAndFeelException;

import javax.swing.event.ChangeEvent;

import javax.swing.event.ChangeListener;

public class ColorBands {

public static void main(String[] args) {

new ColorBands();

}

public ColorBands() {

EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

} catch (ClassNotFoundException | InstantiationException | IllegalAccessException | UnsupportedLookAndFeelException ex) {

}

JFrame frame = new JFrame("Testing");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setLayout(new BorderLayout());

frame.add(new TestPane());

frame.pack();

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

});

}

public class TestPane extends JPanel {

private JPanel bandsPane;

private JSlider slider;

private Timer changeTimer;

public TestPane() {

bandsPane = new JPanel(new GridBagLayout());

slider = new JSlider(1, 100);

setLayout(new BorderLayout());

add(new JScrollPane(bandsPane));

add(slider, BorderLayout.SOUTH);

slider.addChangeListener(new ChangeListener() {

@Override

public void stateChanged(ChangeEvent e) {

changeTimer.restart();

}

});

changeTimer = new Timer(250, new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

int bands = slider.getValue();

List<Color> bandsList = getColorBands(Color.RED, bands);

bandsPane.removeAll();

GridBagConstraints gbc = new GridBagConstraints();

gbc.gridwidth = GridBagConstraints.REMAINDER;

gbc.insets = new Insets(1, 1, 1, 1);

for (Color color : bandsList) {

bandsPane.add(new ColorBand(color), gbc);

}

gbc.weighty = 1;

bandsPane.add(new JPanel(), gbc);

revalidate();

repaint();

}

});

changeTimer.setRepeats(false);

slider.setValue(1);

}

@Override

public Dimension getPreferredSize() {

return new Dimension(200, 200);

}

}

public List<Color> getColorBands(Color color, int bands) {

List<Color> colorBands = new ArrayList<>(bands);

for (int index = 0; index < bands; index++) {

colorBands.add(darken(color, (double) index / (double) bands));

}

return colorBands;

}

public static Color darken(Color color, double fraction) {

int red = (int) Math.round(Math.max(0, color.getRed() - 255 \* fraction));

int green = (int) Math.round(Math.max(0, color.getGreen() - 255 \* fraction));

int blue = (int) Math.round(Math.max(0, color.getBlue() - 255 \* fraction));

int alpha = color.getAlpha();

return new Color(red, green, blue, alpha);

}

public class ColorBand extends JPanel {

public ColorBand(Color color) {

setBackground(color);

}

@Override

public Dimension getPreferredSize() {

return new Dimension(100, 20);}}}

o/p=