Java Homework 2

1. Command window input and output practicing.

Write an application program that inputs from the user the radius of a circle as an integer and prints the circle’s diameter, circumference and area.

(Please use the command window input and output)

*Diameter =* 2 r

*Circumference=* 2πr

*Area =* π**

You may use the predefined const Math.PI for the value of π

Note that the values produced by the circumference and area are floating-point numbers.

2. Please execute, observe and compare the following four cases of the java application programs

and write the differences among these four cases (from the syntax and from the result points).

//**Case I: SortFrame.java**

import java.awt.\*;

import javax.swing.\*;

import java.awt.Color;

public class SortFrame extends JFrame

{ int numbers[ ] = new int [100];

public SortFrame( )

{super("SortJFrame");

setSize(500,600)

setVisible(true); }

public void paint(Graphics g)

{ int i,j;

super.paint(g);

g.setColor(Color.magenta);

for(j = 0; j < numbers.length; ++j)

numbers[j] = (int)(Math.random()\*101);

for(j = 0; j < numbers.length; ++j)

g.drawLine(10, 20+3\*j, 10 + numbers[j], 20+3\*j);

g.drawString("The original array", 10, 350);

bubblesort();

for(j = 0; j < numbers.length; ++j)

g.drawLine(225, 20+3\*j, 225 + numbers[j], 20+3\*j);

g.drawString("The sorted array", 225,350); }

public void bubblesort()

{ int size = numbers.length;

int x, y, temp;

for(x = 0; x < size – 1; ++x)

{ for( y = 0; y < size – 1 – x; ++y)

if(numbers[y] > numbers[y +1] )

{ temp = numbers[y];

numbers[y] = numbers[y+1];

numbers[y+1] = temp; } }

}

public static void main(String args[ ])

{ SortFrame f1 =new SortFrame( );

f1.setDefaultCloseOperation( JFrame.EXIT\_ON\_CLOSE ); }

}

**//CaseII SortFrame2.java**

import java.awt.\*;

import javax.swing.\*;

import java.awt.Color;

public class SortFrame2 extends JFrame

{ int numbers[ ] = new int [100];

public SortFrame2( )

{super("SortJFrame");

for(int j = 0; j < numbers.length; ++j)

numbers[j] = (int)(Math.random()\*101);

setSize(500,600);

setVisible(true); }

public void paint(Graphics g)

{ int i,j;

super.paint(g);

g.setColor(Color.magenta);

for(j = 0; j < numbers.length; ++j)

g.drawLine(10, 20+3\*j, 10 + numbers[j], 20+3\*j);

g.drawString("The original array", 10, 350);

bubblesort();

for(j = 0; j < numbers.length; ++j)

g.drawLine(225, 20+3\*j, 225 + numbers[j], 20+3\*j);

g.drawString("The sorted array", 225,350); }

public void bubblesort() //this bubble sort routine sorts integers from low to high

{ int size = numbers.length;

int x, y, temp;

for(x = 0; x < size – 1; ++x)

{ for( y = 0; y < size – 1 – x; ++y)

if(numbers[y] > numbers[y +1] )

{ temp = numbers[y];

numbers[y] = numbers[y+1];

numbers[y+1] = temp; } } }

public static void main(String args[ ])

{ SortFrame2 f1 =new SortFrame2( );

f1.setDefaultCloseOperation( JFrame.EXIT\_ON\_CLOSE ); } }

**//Case III SortPanel101.java**

import java.awt.\*;

import javax.swing.\*;

import java.awt.Color;

public class SortPanel101 extends JPanel

{ int number1[ ] = new int [100];

int number2[ ] = new int [100];

public SortPanel101( )

{for(int j = 0; j < number1.length; ++j)

{ number1[j] = (int)(Math.random()\*101);

number2[j]=number1[j];}

setSize(500,600);

setVisible(true); }

public void paintComponent(Graphics g)

{ int i,j;

super.paintComponent(g);

g.setColor(Color.magenta);

for(j = 0; j < number1.length; ++j)

g.drawLine(10, 10+3\*j, 10 + number1[j], 10+3\*j);

g.drawString("The original array", 10, 325);

System.out.println(" Original");

bubblesort();

for(j = 0; j < number2.length; ++j)

g.drawLine(225, 10+3\*j, 225 + number2[j], 10+3\*j);

g.drawString("The sorted array", 225,325);

System.out.println(" After sort"); }

public void bubblesort()

{ //we can obtain the size of a Java array like this:

int size = number2.length;

int x, y, temp;

for(x = 0; x < size – 1; ++x)

{ for( y = 0; y < size – 1 – x; ++y)

if(number2[y] > number2[y +1] )

{ temp = number2[y];

number2[y] = number2[y+1];

number2[y+1] = temp; }

}

}

public static void main(String args[ ])

{ SortPanel101 p = new SortPanel101( );

JFrame w = new JFrame();

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

w.add(p);

w.setSize(500,600);

w.setVisible(true);

}}

**// Case IV: contains SortPanel.java and DrawSortTest.java--- TWO files**

//File: SortPanel.java

import java.awt.\*;

import javax.swing.\*;

import java.awt.Color;

public class SortPanel extends JPanel

{ int numbers[ ] = new int [100];

public void paintComponent(Graphics g)

{ int i,j;

super.paintComponent(g);

g.setColor(Color.magenta);

for(j = 0; j < numbers.length; ++j)

numbers[j] = (int)(Math.random()\*101);

for(j = 0; j < numbers.length; ++j)

g.drawLine(10, 20+3\*j, 10 + numbers[j], 20+3\*j);

g.drawString("The original array", 10, 350);

System.out.println(" Original");

bubblesort();

for(j = 0; j < numbers.length; ++j)

g.drawLine(225, 20+3\*j, 225 + numbers[j], 20+3\*j);

g.drawString("The sorted array", 225,350);

System.out.println(" After sort"); }

public void bubblesort()

{ int size = numbers.length;

int x, y, temp;

for(x = 0; x < size – 1; ++x)

{ for( y = 0; y < size – 1 – x; ++y)

if(numbers[y] > numbers[y +1] )

{ temp = numbers[y];

numbers[y] = numbers[y+1];

numbers[y+1] = temp; }

} } }

//File DrawSortTest.java

import javax.swing.JFrame;

public class DrawSortTest

{

public static void main(String args[ ])

{ **SortPanel p = new SortPanel( );**

**JFrame w = new JFrame();**

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

**w.add(p);**

w.setSize(500,600);

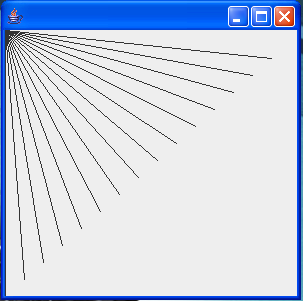
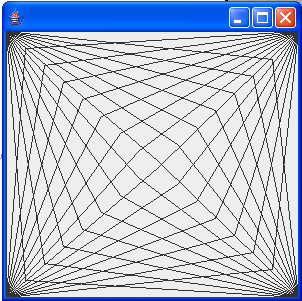
w.setVisible(true); }}

3. Perform the same operation as Fig2.15 in page 106 of the textbook, with the following

modifications:

1. Input data from the input dialog
2. Put the results on the textArea with scroller so the user can scroll results.
3. Put and show the result of the step 2 on the message dialog.

4.

  p fig-1 fig-2

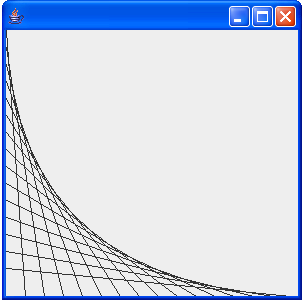
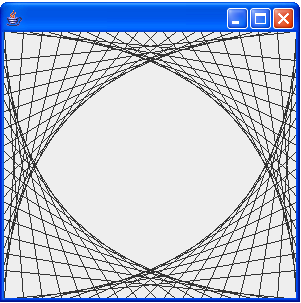
 

fig-3 fig-4

(a).Create the design in the left screen capture of fig-1. The design draws lines from the

top-left corner, fanning out the lines until they cover the upper-left half of the panel.

(b).Modify your program in part (a) to have lines fan out from all four corners, as shown

in fig-2. Lines from opposite corners should intersect along the middle.

(c).Create the design in the left screen capture of fig-3. Begin by dividing each edge into

an equal number of increments. The first line starts in the top-left corner and ends one

step right on the bottom edge. For each successive line, move down one increment on the

left edge and right one increment on the bottom edge. The figure should scale as you

resize the window so that the endpoints always touch the edges.

(d).Modify your program in part (c) to mirror the design in all four corners, as shown in fig-4.

(e). You can extend these problems into more advanced. For example: The use can provide the

number of the lines fan out from four corners, ….colored lines.

5. A scalar quantity can be represented by a single number. For example, the

temperature at a given location is a scalar.

In contrast, a vector quantity has both a magnitude and a direction associated

with it. For example, the velocity of an automobile is a vector, since it has both

a magnitude and a direction.

Vectors can be defined either by a magnitude and a direction, o by the components

of the vector projected along the axes of a rectangular coordinate system.

The two representations are equivalent.

For two-dimensional vectors, write a Java program, input option and corresponding data:

Option 1: Convert from rectangular from into polar form

Option II: Convert from polar form into rectangular form

Case I: Use input dialog and output dialog for input and output.

Case II: Use command line input and output.

1. Z = i X + j Y (rectangular form) where i and j are the unit vectors in the x and y directions, respectively.

2. Z = A∠θ (polar form)

where A is the magnitude, θ is the phase angle

3. X = A cosθ Y = A sinθ ( polar form --> rectangular form )

A =  θ= ( Y/X )

(rectangular form--> polar form )