МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАФІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ФАКУЛЬТЕТ ІНФОРМАТИКИ І ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

Лабораторна робота №1

з дисципліни «Комп’ютерна графіка»

Виконав:

студент гр. ІО-41

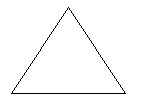
бригада №5

Смішний Дмитро

Перевірив:

Саверченко В. Г.

Київ 2016

**Варіант завдання**: 5

1. Відповідно до варіанту розробити модель елемента зображення М1А1, А2), де А1, А2 - параметри, що визначають розмір і число графічних примітивів. При А1 = А2 = 1 модель елемента зображення відповідає графічному примітиву.

2. Розробити модель орнаменту M2 (М1, B1, B2, B3) де M1 - модель елемента зображення, а B1, B2, B3 - параметри, що визначають конфігурацію (по колу, змійкою і т.п.), кількість і крок тиражування елементів зображення. Змінюючи параметри А1, А2, B1, B2, B3 отримати орнамент, причому елементи зображення не повинні накладатися одна на одну.

3. Домогтися візуального спецефекта (муар, пульсування і т.п.) за рахунок накладення один на одного елементів зображення шляхом зміни параметрів А1, А2, B1, B2, B3.

4. У висновках по роботі вказати значення параметрів по пунктам 2 і 3.

**Код програми**:

**Клас MainFrame**

**public** **class** MainFrame {

**private** JFrame frame;

**private** JPanel panel1;

Triangle triangle = **new** Triangle(70, 70);

Triangle triangle1 = **new** Triangle(300, 300);

Draw drawFigure = **new** Draw(**new** Point (30,100), triangle);

Draw1 drawEffect1 = **new** Draw1 (**new** Point(290-triangle1.getWidth()/2,(**int**)(250+triangle.getHeight()\*1.5)), triangle1);

**private** JTextField width;

**private** JTextField hight;

**private** JTextField numberElements;

**private** JTextField angle;

/\*\*

\* Launch the application.

\*/

**public** **static** **void** main(String[] args) {

EventQueue.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**try** {

MainFrame window = **new** MainFrame();

window.frame.setVisible(**true**);

} **catch** (Exception e) {

e.printStackTrace();

}

}

});

}

/\*\*

\* Create the application.

\*/

**public** MainFrame() {

initialize();

}

/\*\*

\* Initialize the contents of the frame.

\*/

**private** **void** initialize() {

frame = **new** JFrame();

frame.setExtendedState(frame.***MAXIMIZED\_BOTH***);;

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

frame.getContentPane().setLayout(**null**);

triangle.lineAlignment();

drawFigure.setBounds(10, 11, 740, 683);

frame.getContentPane().add(drawFigure);

drawEffect1.setBounds(760, 11, 580, 500);

frame.getContentPane().add(drawEffect1);

width = **new** JTextField();

width.setBounds(857, 522, 95, 25);

width.setText(**new** Integer(triangle1.getWidth()).toString());

frame.getContentPane().add(width);

width.setColumns(10);

hight = **new** JTextField();

hight.setBounds(857, 558, 95, 25);

hight.setText(**new** Integer(triangle1.getHeight()).toString());

frame.getContentPane().add(hight);

hight.setColumns(10);

numberElements = **new** JTextField();

numberElements.setBounds(1151, 522, 103, 25);

numberElements.setText(**new** Integer(drawEffect1.*numberTriangle*).toString());

frame.getContentPane().add(numberElements);

numberElements.setColumns(10);

angle = **new** JTextField();

angle.setBounds(1151, 558, 103, 22);

angle.setText(**new** Integer(drawEffect1.*angle*).toString());

frame.getContentPane().add(angle);

angle.setColumns(10);

JLabel label = **new** JLabel("Ширина :");

label.setBounds(760, 522, 73, 25);

frame.getContentPane().add(label);

JLabel label\_1 = **new** JLabel("Высота :");

label\_1.setBounds(760, 558, 54, 19);

frame.getContentPane().add(label\_1);

JLabel label\_2 = **new** JLabel("Кол-во элементов :");

label\_2.setBounds(1004, 522, 123, 25);

frame.getContentPane().add(label\_2);

JLabel label\_3 = **new** JLabel("Угол, градусы :");

label\_3.setBounds(1004, 561, 103, 19);

frame.getContentPane().add(label\_3);

JButton btnNewButton = **new** JButton("Обработать");

btnNewButton.setBounds(1151, 602, 103, 32);

btnNewButton.addActionListener(**new** AbstractAction() {

@Override

**public** **void** actionPerformed(ActionEvent e) {

// **TODO** Auto-generated method stub

triangle1.setWidth(**new** Integer(width.getText()));

triangle1.setHeight(**new** Integer(hight.getText()));

drawEffect1.*numberTriangle*=**new** Integer(numberElements.getText());

drawEffect1.*angle*=**new** Integer(angle.getText());

frame.getContentPane().revalidate();

frame.getContentPane().repaint();

}

});

frame.getContentPane().add(btnNewButton);

}

}

**Клас Triangle**

**public** **class** Triangle {

**private** **int** height;

**private** **int** width;

Triangle(**int** height, **int** width){

**this**.height=height;

**this**.width = width;

}

**public** **int** getHeight(){

**return** height;

}

**public** **int** getWidth(){

**return** width;

}

**public** **void** setHeight(**int** height){

**this**.height=height;

}

**public** **void** setWidth(**int** width){

**this**.width=width;

}

**public** **void** lineAlignment(){

**int** scrollPaneWidth = 740;

**int** initIndent = (7\*20)+(width\*8);

**int** finalIndent = scrollPaneWidth-initIndent;

Draw.*startPoint*.x=finalIndent/2;

}

}

**Клас Draw**

**public** **class** Draw **extends** JPanel{

**public** **static** Point *startPoint*;

**public** Triangle triangle;

Color[] colors = {

Color.***RED***,

Color.***ORANGE***,

Color.***YELLOW***,

Color.***GREEN***,

Color.***BLUE***,

Color.***CYAN***,

Color.***MAGENTA***

};

// Color newColor=colors[(int)(7\*Math.random())];

**int** numberTriangle = 2;

**int** angle=180;

Draw(Point startPoint, Triangle triangle){

**this**.*startPoint* = startPoint;

**this**.triangle = triangle;

}

**public** **void** paintComponent(Graphics gr){

**super**.paintComponent(gr);

Color newColor=colors[(**int**)(7\*Math.*random*())];

gr.setColor(newColor);

Point p2 = **new** Point(*startPoint*.x+triangle.getWidth(), *startPoint*.y);

Point p3 = **new** Point(*startPoint*.x+triangle.getWidth()/2, *startPoint*.y-triangle.getHeight());

gr.drawLine(*startPoint*.x, *startPoint*.y, p2.x, p2.y);

gr.drawLine(p2.x, p2.y, p3.x, p3.y);

gr.drawLine(p3.x, p3.y, *startPoint*.x, *startPoint*.y);

visualEffect(gr, triangle, *startPoint*, numberTriangle, angle);

elementArrHor(gr, p2, 7, 20);

elementArrVer(gr, 180, 5, 740, 95);

Point startPoint = **new** Point (250,630);

visualEffect(gr, triangle, startPoint, numberTriangle, angle);

paintNext(gr, triangle, startPoint);

Point startPoint\_1 = **new** Point(150, 560);

visualEffect(gr, triangle, startPoint\_1, numberTriangle, angle);

paintNext(gr, triangle, startPoint\_1);

}

**public** **int** paintNext(Graphics gr,Triangle triangle, Point startPoint){

Color newColor=colors[(**int**)(7\*Math.*random*())];

gr.setColor(newColor);

Point p2 = **new** Point(startPoint.x+triangle.getWidth(), startPoint.y);

Point p3 = **new** Point(startPoint.x+triangle.getWidth()/2, startPoint.y-triangle.getHeight());

gr.drawLine(startPoint.x, startPoint.y, p2.x, p2.y);

gr.drawLine(p2.x, p2.y, p3.x, p3.y);

gr.drawLine(p3.x, p3.y, startPoint.x, startPoint.y);

**return** p2.x;

}

**public** **int** spinningX(**int** numberTriangle, Point fatherStartPoint, Point motherStartPoint, **int** angle){

**int** alpha = angle\*numberTriangle;

**double** radian = (Math.***PI***\*alpha)/180;

**int** x0=fatherStartPoint.x+triangle.getWidth()/2;

**int** y0= (**int**) ((fatherStartPoint.y-triangle.getHeight())+triangle.getHeight()/1.5);

**int** turnedX=(**int**) (x0+(motherStartPoint.x-x0)\*Math.*cos*(radian)-(motherStartPoint.y-y0)\*Math.*sin*(radian));

**return** turnedX;

}

**public** **int** spinningY(**int** numberTriangle, Point fatherStartPoint, Point motherStartPoint, **int** angle){

**int** alpha = angle\*numberTriangle;

**double** radian = (Math.***PI***\*alpha)/180;

**int** x0=fatherStartPoint.x+triangle.getWidth()/2;

**int** y0= (**int**) ((fatherStartPoint.y-triangle.getHeight())+triangle.getHeight()/1.5);

**int** turnedY=(**int**) (y0+(motherStartPoint.y-y0)\*Math.*cos*(radian)+(motherStartPoint.x-x0)\*Math.*sin*(radian));

**return** turnedY;

}

**public** **void** spinningTriangle(Graphics gr,Triangle triangle, Point startPoint, **int** numberTriangle, **int** angle){

Color newColor=colors[(**int**)(7\*Math.*random*())];

gr.setColor(newColor);

Point finP1 = **new** Point (spinningX(numberTriangle, startPoint, startPoint, angle), spinningY(numberTriangle, startPoint, startPoint, angle));

Point p2 = **new** Point(startPoint.x+triangle.getWidth(), startPoint.y);

Point finP2= **new** Point(spinningX(numberTriangle, startPoint,p2, angle), spinningY(numberTriangle, startPoint, p2, angle));

Point p3 = **new** Point(startPoint.x+triangle.getWidth()/2, startPoint.y-triangle.getHeight());

Point finP3 = **new** Point (spinningX(numberTriangle, startPoint, p3, angle), spinningY(numberTriangle, startPoint, p3, angle));

gr.drawLine(finP1.x, finP1.y, finP2.x, finP2.y);

gr.drawLine(finP2.x, finP2.y, finP3.x, finP3.y);

gr.drawLine(finP3.x, finP3.y, finP1.x, finP1.y);

}

**public** **void** visualEffect(Graphics g, Triangle triangle, Point startPoint, **int** numberTriangle, **int** angle){

**for** (**int** i=1; i<numberTriangle; i++)

spinningTriangle(g, triangle, startPoint, i, angle);

}

**public** **void** elementArrHor(Graphics gr, Point p2, **int** sizeArr, **int** widthBetweenElements){

**int** newX=p2.x+widthBetweenElements;

**int** newY= p2.y;

**for** (**int** i=0; i<sizeArr; i++){

Point startPoint = **new** Point(newX, newY);

visualEffect(gr, triangle, startPoint, numberTriangle, angle);

newX = paintNext(gr, triangle, startPoint)+widthBetweenElements;

}

}

**public** **void** elementArrVer(Graphics gr, **int** beginDown, **int** sizeArr, **int** scrollPaneWidth, **int** widthBetweenElements){

**int** downY = beginDown;

**for** (**int** j=0; j<sizeArr; j++){

Point startPoint = **new** Point (scrollPaneWidth/2-(triangle.getWidth()/2), downY);

visualEffect(gr, triangle, startPoint, numberTriangle, angle);

paintNext(gr, triangle, startPoint);

downY+=widthBetweenElements;

}

}

}

**Клас Draw1**

**public** **class** Draw1 **extends** JPanel{

**public** **static** Point *startPoint*;

**public** Triangle triangle;

Color[] colors = {

Color.***RED***,

Color.*decode*("#BA55D3"),

Color.*decode*("#008B8B"),

Color.*decode*("#08000"),

Color.*decode*("#FF7F50"),

Color.*decode*("#8B4513"),

Color.*decode*("#A52A2A")

};

Color newColor=colors[(**int**)(7\*Math.*random*())];

**public** **static** **int** *numberTriangle* = 10000;

**public** **static** **int** *angle*=3;

Draw1(Point startPoint, Triangle triangle){

**this**.*startPoint* = startPoint;

**this**.triangle = triangle;

}

**public** **void** paintComponent(Graphics gr){

**super**.paintComponent(gr);

gr.setColor(newColor);

Point p2 = **new** Point(*startPoint*.x+triangle.getWidth(), *startPoint*.y);

Point p3 = **new** Point(*startPoint*.x+triangle.getWidth()/2, *startPoint*.y-triangle.getHeight());

gr.drawLine(*startPoint*.x, *startPoint*.y, p2.x, p2.y);

gr.drawLine(p2.x, p2.y, p3.x, p3.y);

gr.drawLine(p3.x, p3.y, *startPoint*.x, *startPoint*.y);

visualEffect(gr, triangle, *startPoint*, *numberTriangle*, *angle*);

}

**public** **int** spinningX(**int** numberTriangle, Point fatherStartPoint, Point motherStartPoint, **int** angle){

**int** alpha = angle\*numberTriangle;

**double** radian = (Math.***PI***\*alpha)/180;

**int** x0=fatherStartPoint.x+triangle.getWidth()/2;

**int** y0= (**int**) ((fatherStartPoint.y-triangle.getHeight())+triangle.getHeight()/1.5);

**int** turnedX=(**int**) (x0+(motherStartPoint.x-x0)\*Math.*cos*(radian)-(motherStartPoint.y-y0)\*Math.*sin*(radian));

**return** turnedX;

}

**public** **int** spinningY(**int** numberTriangle, Point fatherStartPoint, Point motherStartPoint, **int** angle){

**int** alpha = angle\*numberTriangle;

**double** radian = (Math.***PI***\*alpha)/180;

**int** x0=fatherStartPoint.x+triangle.getWidth()/2;

**int** y0= (**int**) ((fatherStartPoint.y-triangle.getHeight())+triangle.getHeight()/1.5);

**int** turnedY=(**int**) (y0+(motherStartPoint.y-y0)\*Math.*cos*(radian)+(motherStartPoint.x-x0)\*Math.*sin*(radian));

**return** turnedY;

}

**public** **void** spinningTriangle(Graphics gr,Triangle triangle, Point startPoint, **int** numberTriangle, **int** angle){

Point finP1 = **new** Point (spinningX(numberTriangle, startPoint, startPoint, angle), spinningY(numberTriangle, startPoint, startPoint, angle));

Point p2 = **new** Point(startPoint.x+triangle.getWidth(), startPoint.y);

Point finP2= **new** Point(spinningX(numberTriangle, startPoint,p2, angle), spinningY(numberTriangle, startPoint, p2, angle));

Point p3 = **new** Point(startPoint.x+triangle.getWidth()/2, startPoint.y-triangle.getHeight());

Point finP3 = **new** Point (spinningX(numberTriangle, startPoint, p3, angle), spinningY(numberTriangle, startPoint, p3, angle));

gr.drawLine(finP1.x, finP1.y, finP2.x, finP2.y);

gr.drawLine(finP2.x, finP2.y, finP3.x, finP3.y);

gr.drawLine(finP3.x, finP3.y, finP1.x, finP1.y);

}

**public** **void** visualEffect(Graphics gr, Triangle triangle, Point startPoint, **int** numberTriangle, **int** angle){

**for** (**int** i=1; i<numberTriangle; i++){

gr.setColor(newColor);

spinningTriangle(gr, triangle, startPoint, i, angle);

}

}

}

**Результати роботи програми:**

**Висновок:** Було розроблено програму, що забезпечує малювання літери J за допомогою моделі, створеної з супротивно-повернутих трикутних примітивів.

Таким чином:

-- А1=300, А2=2,

-- В1=форма літери J, B2=15, B3=20.

Створення самої моделі відбувалося за наступною формулою:

X = x0 + (x - x0) \* cos(a) - (y - y0) \* sin(a);  
Y = y0 + (y - y0) \* cos(a) + (x - x0) \* sin(a);

Де, х0, у0 – координати центру.

Для отримання наступних зображень змінювалися такі параметри: ширина і висота початкового трикутника, зміна кута обертання, кількість елементів в примітиві.

