Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа № 3

По дисциплине ГИИС

Тема: «Алгоритмы растеризации»

Выполнил:

Студент 4 курса

Группы ИИ-16 (2)

Пешко А.С.

Проверил:

Михно Е.В.

Брест, 2021

**Цель работы:**  Изучить алгоритмы растеризации.

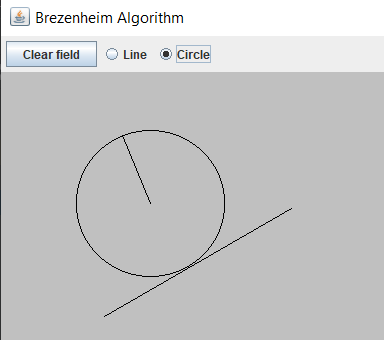
**Ход работы:**

**Задание:** написать программу, которая из отдельных пикселов рисует линию и окружность, используя алгоритм Брезенхейма.

public class BrezenheimAlgorithm extends JFrame {  
  
 String mode;  
 HashSet<Point> points = new HashSet<>();  
   
 public BrezenheimAlgorithm() {  
 setTitle("Brezenheim Algorithm");  
 setSize(1000, 1000);  
 setLocationRelativeTo(null);  
 setLayout(new BorderLayout());  
 JPanel submenu = createSubMenu();  
 add(submenu, BorderLayout.NORTH);  
 JPanel drawField = new JPanel();  
  
 drawField.setBackground(Color.lightGray);  
 add(drawField, BorderLayout.CENTER);  
 ArrayList<Point> list = new ArrayList<>();  
 drawField.addMouseListener(new MouseListener() {  
  
 @Override  
 public void mouseClicked(MouseEvent e) {  
  
  
 }  
  
 @Override  
 public void mousePressed(MouseEvent e) {  
  
 }  
  
 @Override  
 public void mouseReleased(MouseEvent e) {  
 if (list.size() == 2) {  
 if (mode.equals("line"))  
 drawBrasenhamLine(list.get(0), list.get(1));  
 else  
 drawCircle(list.get(0), list.get(1));  
 list.clear();  
 }  
 }  
  
 @Override  
 public void mouseEntered(MouseEvent e) {  
  
 }  
  
 @Override  
 public void mouseExited(MouseEvent e) {  
  
 }  
  
 });  
 setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);  
 setResizable(true);  
 setVisible(true);  
 }  
  
 @Override  
 public void paint(Graphics g) {  
 super.paint(g);  
 Graphics2D graphics2D = (Graphics2D) g;  
 graphics2D.setColor(Color.black);  
 for (Point p : points)  
 graphics2D.drawLine(p.x, p.y, p.x, p.y);  
 }  
  
 private JPanel createSubMenu() {  
 JPanel panel = new JPanel();  
  
 panel.setLayout(new FlowLayout(FlowLayout.LEFT));  
  
 JButton clear = new JButton("Clear field");  
 clear.addActionListener(e -> {  
 points.clear();  
 repaint();  
 });  
 JRadioButton line = new JRadioButton("Line");  
 mode = "line";  
 line.setSelected(true);  
 JRadioButton circle = new JRadioButton("Circle");  
 line.addActionListener(e -> {  
 mode = "line";  
 circle.setSelected(false);  
 line.setSelected(true);  
 });  
 circle.addActionListener(e -> {  
 mode = "circle";  
 line.setSelected(false);  
 circle.setSelected(true);  
 });  
  
 panel.add(clear);  
 panel.add(line);  
 panel.add(circle);  
  
 return panel;  
 }  
  
 public void drawBrasenhamLine(Point start, Point end) {  
 int x, y, deltaX, deltaY, incX, incY, pdx, pdy, deltaErr, el, err;  
  
 deltaX = start.x - end.x;  
 deltaY = start.y - end.y;  
  
 incX = Integer.compare(deltaX, 0);  
 incY = Integer.compare(deltaY, 0);  
  
 deltaX = Math.abs(deltaX);  
  
 deltaY = Math.abs(deltaY);  
  
 if (deltaX > deltaY) {  
 pdx = incX;  
 pdy = 0;  
 deltaErr = deltaY;  
 el = deltaX;  
 } else {  
 pdx = 0;  
 pdy = incY;  
 deltaErr = deltaX;  
 el = deltaY;  
 }  
 x = start.x;  
 y = start.y;  
 err = el / 2;  
 points.add(new Point(x, y));  
  
 for (int t = 0; t < el; t++) {  
 err -= deltaErr;  
 if (err < 0) {  
 err += el;  
 x -= incX;

y -= incY;   
 } else {  
 x -= pdx;   
 y -= pdy;   
 }  
  
 points.add(new Point(x, y));  
 }  
 repaint();  
 }  
  
  
 private void drawCircle(Point center, Point pointRadius) {  
 int radius = Math.round(new Float(Math.sqrt(Math.pow(center.x - pointRadius.x, 2) + Math.pow(center.y - pointRadius.y, 2))));  
 int x = 0;  
 int y = radius;  
 int delta = 1 - 2 \* radius;  
 int error;  
 while (y >= 0) {  
 points.add(new Point(center.x + x, center.y + y));  
 points.add(new Point(center.x + x, center.y - y));  
 points.add(new Point(center.x - x, center.y + y));  
 points.add(new Point(center.x - x, center.y - y));  
  
 error = 2 \* (delta + y) - 1;  
 if (delta < 0 && error <= 0) {  
 ++x;  
 delta += 2 \* x + 1;  
 continue;  
 }  
 error = 2 \* (delta - x) - 1;  
 if (delta > 0 && error > 0) {  
 --y;  
 delta += 1 - 2 \* y;  
 continue;  
 }  
 ++x;  
 delta += 2 \* (x - y);  
 --y;  
 }  
 repaint();  
 }  
 public static void main(String[] args) {  
 new BrezenheimAlgorithm();  
 }  
}

**Результат работы:**



**Вывод:** изучили алгоритмы растеризации.