МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждения высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра «Математического обеспечения и применения ЭВМ»

|  |  |  |
| --- | --- | --- |
| 1002262_584459121576624_356098510_n |  | logo МОПЭВМ |
|  |  |  |

**Лабораторная работа №2**

по курсу

компьютерная графика

Вариант№ 31

Выполнили:

студент группы КТбо3-9

Митина А.Г.

Ячменев М.И.

Проверил:

Селянкин В.В.

Таганрог, 2016

**Постановка задачи**

Задание заключается в разработке программы развертки прямой и окружности

**Вариант задания**

Необходимо написать программу развертки прямой и окружности в 5 октане.

**Описание алгоритм**

Для развертки прямой в 5 октане необходимо использовать декремент по x и инкремент по y на каждом шаге. При этом дельта x и y вычисляется, как разница между конечной и начальной точкой по x и y соответственно.

Для развертки окружности в 5 октане по алгоритму Брезенхема мы используем декремент по y каждом шаге, при этом пока значение x меньше значения y. Определяем в каком из 3 направлений следует идти.

**Описание программы**

Входные значения: координаты точек отрезка в декартовой системе координат и радиус окружности.

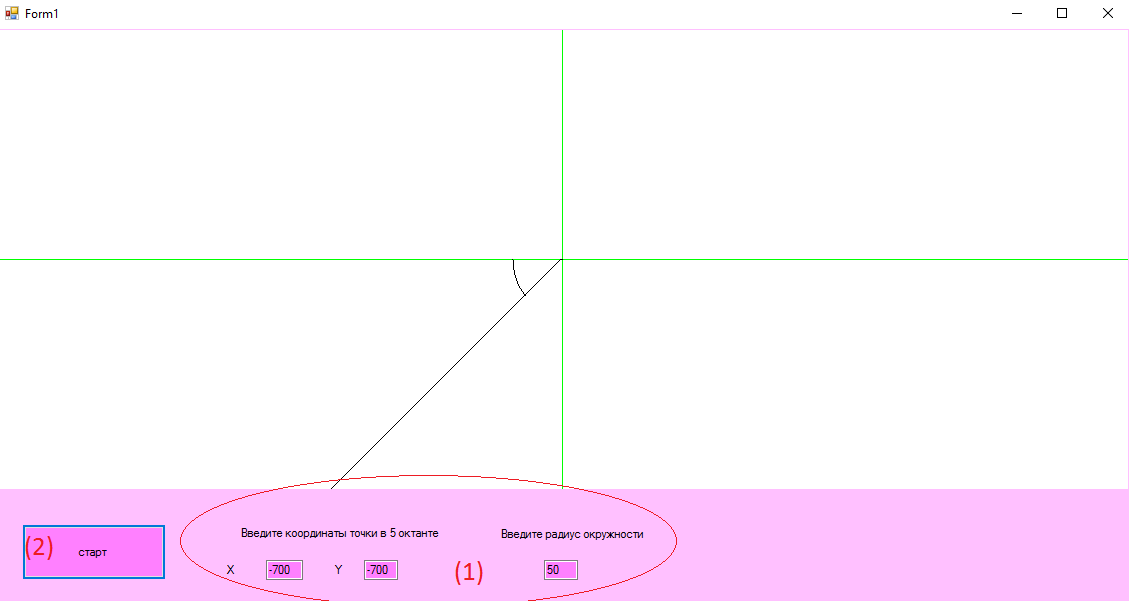
Выходные значения: растровая развертка отрезка и окружности на экране, либо сообщение о не корректных входных данных.

Системные требования: процессор Intel Core I3 2.0Ггц

Видеокарта Nvidia GeForce 940M

Оперативная память 8 Гб

Программы написана на языке С# в среде разработке Visual Studio 2015.

В форме (1) пользователь вводит необходимые значения. Эти значения обрабатываются по нажатию кнопки (2). Если данные введены некорректно, то появляется соответствующее сообщение(3).

****

В методе BackGround() происходит задание фона и системы координат. В мтоде button1\_Click() происходит проверка вводимой информации, а именно проверка на принадлежность точек прямой 5 октану и положительному радиусу окружности. В методе SetArc() и SetLine рисуется окружность и прямая соответственно. Методы GetX и GetY возвращают значения по x и по yв пересчете на новые координаты.

**Листинг**

1. using System;
2. using System.Collections.Generic;
3. using System.ComponentModel;
4. using System.Data;
5. using System.Drawing;
6. using System.Linq;
7. using System.Text;
8. using System.Threading.Tasks;
9. using System.Windows.Forms;
10. using System.Windows;
11. namespace CG\_2
12. {
13. public partial class Form1 : Form
14. {
15. Graphics gr;
16. Pen p = new Pen(Color.Lime);
17. SolidBrush fon;
19. public Form1()
20. {
21. InitializeComponent();
22. }
23. *// рисуем оси координат*
24. private void BackGround()
25. {
26. Color pix = Color.Chocolate;
27. gr = pictureBox1.CreateGraphics();
28. fon = new SolidBrush(Color.Black);
30. gr.FillRectangle(fon, 0, 0, pictureBox1.Width, pictureBox1.Height);
31. gr.DrawLine(p, 0, pictureBox1.Height / 2, pictureBox1.Width, pictureBox1.Height / 2);
32. gr.DrawLine(p, pictureBox1.Width / 2, 0, pictureBox1.Width / 2, pictureBox1.Height);
33. }
34. *// обрабатываем входные данные*
35. private void button1\_Click(object sender, EventArgs e)
36. {
37. BackGround();
38. Point Start\_Coor = new Point(0, 0);
39. Point Point\_Coor = new Point(Int32.Parse(X\_Box.Text), Int32.Parse(Y\_Box.Text));
40. int r = Int32.Parse(RBox.Text);
41. if ((Point\_Coor.X < 0) && (Point\_Coor.Y >= Point\_Coor.X) && (Point\_Coor.Y < 0) && r >= 0)*// проверка на корректность*
42. {
43. label4.Text = "";
44. SetLine(Start\_Coor, Point\_Coor);
45. int x = Int32.Parse(sentreX\_Box.Text);
46. int y = Int32.Parse(sentreY\_Box.Text);
47. SetArc(r, x, y);
48. }
49. else
50. {
51. label4.Text = "Некорректные данные!";
52. }
53. }
55. *// равертка окружности по алгоритму Брезенхема*
56. private void SetArc(int radius, int \_x, int \_y)
57. {
58. int x = -radius, y = 0, gap = 0, delta = (2 - 2 \* radius);
59. while (x <= 0 && x <= y)
60. {
61. PutPixel(GetX(\_x + x), GetY(\_y + y));
62. PutPixel(GetX(\_x + x), GetY(\_y - y));
63. PutPixel(GetX(\_x - x), GetY(\_y - y));
64. PutPixel(GetX(\_x - x), GetY(\_y + y));
65. PutPixel(GetX(\_y + y), GetY(\_x + x));
66. PutPixel(GetX(\_y - y), GetY(\_x + x));
67. PutPixel(GetX(\_y - y), GetY(\_x - x));
68. PutPixel(GetX(\_y + y), GetY(\_x - x));
70. gap = 2 \* (delta + y) - 1;
71. if (delta < 0 && gap <= 0)
72. {
73. y--;
74. delta -= 2 \* y + 1;
75. continue;
76. }
77. if (delta > 0 && gap > 0)
78. {
79. x++;
80. delta += 2 \* x + 1;
81. continue;
82. }
83. x++;
84. delta += 2 \* (x - y);
85. y--;
86. }
88. }
89. *// развертка прямой*
90. private void SetLine( Point start, Point finish)
91. {
92. int x = start.X;
93. int y = start.Y;
94. int dx = finish.X - start.X;
95. int dy = finish.Y - start.Y;
96. int D = -dx;
97. int DX = dx >> 1;
98. int DY = dy >> 1;
99. while (x > finish.X)
100. {
101. PutPixel(GetX(x), GetY(y));
102. --x;
103. D += DY;
104. if (D < 0)
105. {
106. --y;
107. D -= DX;
108. }
109. }
110. }
111. private void PutPixel(int x,int y)
112. {
113. gr.FillRectangle(Brushes.Aqua, x, y, 1, 1);
114. }
115. *// вернуть координаты по x в пересчете на декартовую систему*
116. private int GetX(int X)
117. {
118. return X + pictureBox1.Width/2;
119. }
120. *// вернуть координаты по y*
121. private int GetY(int Y)
122. {
123. return pictureBox1.Height/2-Y;
124. }
126. private void label3\_Click(object sender, EventArgs e)
127. {
129. }
131. private void pictureBox1\_Click(object sender, EventArgs e)
132. {
134. }
136. private void label5\_Click(object sender, EventArgs e)
137. {
139. }
140. }
141. }