

**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ (МИИТ)**

**ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ**

**Лабораторная работа №2**

**по дисциплине** «**Методы программирования**»

«Конструктор ГПИ. Фигуры»

**Выполнил:** ст. гр. ТКИ-341

Соколов И.Д.

Ковров А.И.

**Проверил:** к.т.н. Сафронов А.И.

**Москва – 2024 г**

1. **Цель работы**

«Закрепить навыки разработки визуального пользовательского интерфейса, освоить работу с текстовыми файлами и кодировкой в среде *Microsoft* *Visual Studio*, научиться реализовывать настройку множественных состояний объектов посредством управления компонентами со внутренней индексацией».

# Формулировка задачи

«В интегрированной среде разработки *Microsoft* *Visual Studio* разработать программу в режиме *Windows Forms Application* на языке *Visual C#*, представляющую собой экранную форму, содержащую главное меню, позволяющее:

1. Начать работу с приложением.
2. Прервать работу приложения.
3. Предоставить пользователю справочную информацию о работе с приложением.

Сама программа должна реализовывать вывод в графический элемент управления (например, *PictureBox*) главной экранной формы плоскостную геометрическую фигуру, выбираемую пользователем из списка (вид списка \* задаётся вариантом индивидуального задания). Список должен обязательно содержать следующие пункты:

1. «Квадрат»,
2. «Прямоугольный треугольник»,
3. «Эллипс»,
4. «Равнобедренный треугольник»,
5. «Круг»,
6. «Равносторонний треугольник»,
7. «Окружность»,
8. «Ромб»,
9. «Трапеция»,
10. «Параллелограмм»,
11. «Прямоугольник».

## **2.1 Индивидуальная задача**

Прозрачная форма справки, удаление обычных средств работы с окном и прописывание своих, выбор фигуры реализован с помощью Список и кнопка подтверждения

# Детализированное текстовое описание ситуации

Пользователь запускает программу и может выбрать, что он хочет сделать: начать рисование, получить справку или закрыть окно. При выборе начать рисование пользователь может выбрать фигуру, ее цвет и размер, после чего подтвердить свой выбор и поставить фигуру в желаемое место в picturebox.

1. **Сеть Петри – схема ситуации**

**Изображение выглядит как круг, диаграмма, линия

Автоматически созданное описание4.1 Полная**

Рисунок 1 – Начальная форма

Изображение выглядит как диаграмма, круг, текст, линия

Автоматически созданное описание

Рисунок 2 – Отдельный блок для закрытия, растягивания и скрытия форм

Изображение выглядит как круг, диаграмма, снимок экрана, мультфильм

Автоматически созданное описание

Рисунок 3 – Форма справки

Изображение выглядит как диаграмма, круг, зарисовка, рисунок

Автоматически созданное описание

Рисунок 4 – Полная сеть Петри

**4.2 Краткая**

**Изображение выглядит как круг, диаграмма, дизайн

Автоматически созданное описание**

Рисунок 5 – Начальная форма

Изображение выглядит как круг, диаграмма, линия, дизайн

Автоматически созданное описание

Рисунок 6 – Отдельный блок для закрытия, растягивания и скрытия форм

Изображение выглядит как круг, диаграмма, снимок экрана, дизайн

Автоматически созданное описание

Рисунок 7 – Форма справки

Изображение выглядит как диаграмма, круг, рисунок, линия

Автоматически созданное описание

Рисунок 8 – Форма Paint

1. **Описание:**

*–*состояния (states):

s1 *–* *Начальная форма*

s2 *–* *Отрисована фигура*

s3 *– Форма Paint*

s4 *– форма хелп*

*–* действия (effects):

e1 *–* Пользователь нажимает кнопку \_

e2 *–* Пользователь нажимает кнопку О

e3 *–* Пользователь нажимает кнопку Х

e4 *–* Пользователь кликает мышкой

*–* события (prompts):

p1 *–* кнопка “Начать рисовать”

p2 *–* кнопка “Help”

p3 *–* Кнопка “\_”

p4 *–* Кнопка “О”

p5 *–* Кнопка “Х”

p6 *–* Кнопка “Apply”

p7 *–* Кнопка “тип фигуры”

p6 *–* Кнопка “размер фигуры”

p7 *–* Кнопка “цвет фигуры”

1. **Диаграмма классов**

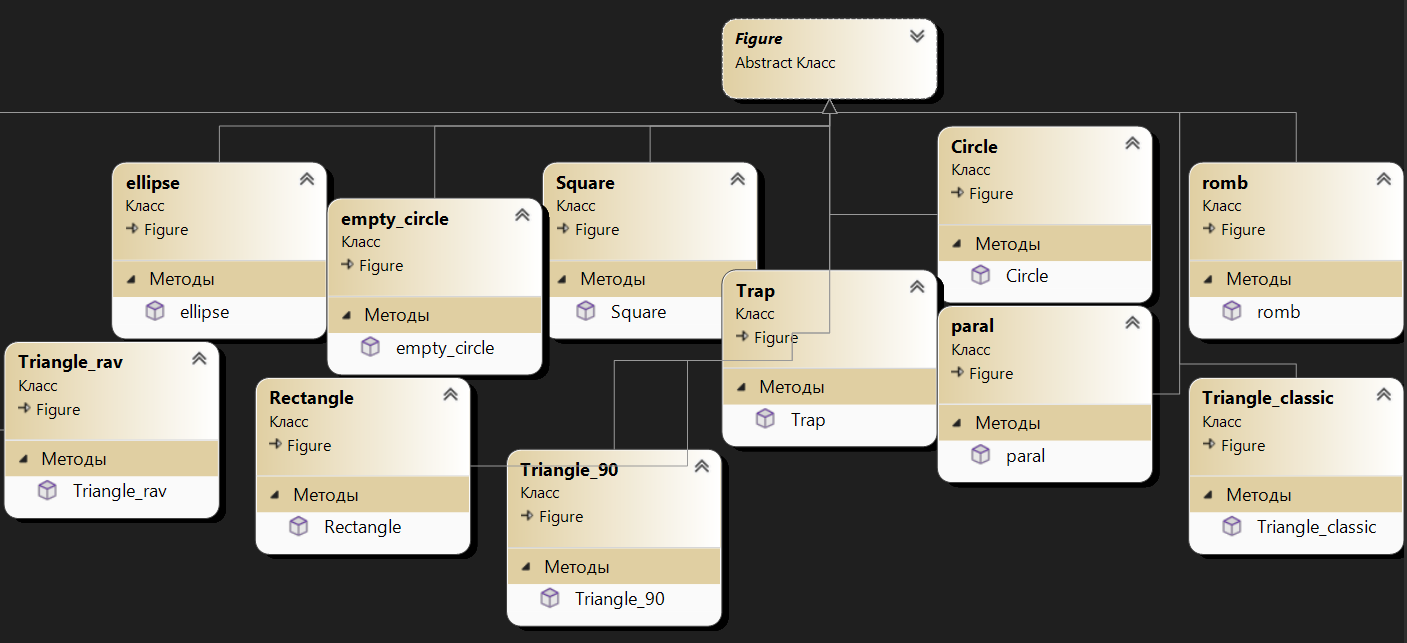


Рисунок 9 – классы фигур

**Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание**

Рисунок 10 – классы форм

1. **Скриншоты работы программы**

**Изображение выглядит как текст, снимок экрана, Шрифт, программное обеспечение

Автоматически созданное описание**

Изображение выглядит как текст, снимок экрана, программное обеспечение, дисплей

Автоматически созданное описаниеРисунок 11 – Начальная форма

Рисунок 12 – Форма Paint

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, компьютер, Шрифт

Автоматически созданное описаниеРисунок 3 – Демонстрация работоспособности программы

Рисунок 3 – Форма справки

Изображение выглядит как текст, снимок экрана, программное обеспечение, компьютер

Автоматически созданное описание

Рисунок 3 – Рисовка квадрата

1. **Вывод**

Была изучена работа с Windows Forms, работа с классами.

Освоены навыки иллюстрации карты технологического процесса на базе аппарата сетей Петри.

1. **Код программы**
2. using System;
3. using System.Collections.Generic;
4. using System.ComponentModel;
5. using System.Data;
6. using System.Drawing;
7. using System.Linq;
8. using System.Text;
9. using System.Threading.Tasks;
10. using System.Windows.Forms;
11. namespace test
12. {
13. public partial class Form1 : Form
14. {
15. public Form1()
16. {
17. InitializeComponent();
18. this.Text = "Задание №2 выполнили Ковров А.И. и Соколов И.Д.; Номер варианта 11, дата выполнения ХХ.ХХ.ХХ";
19. //this.Opacity = 0.5;
20. this.BackColor = Color.Black;
21. this.ControlBox = false;
22. }
24. private void button1\_Click(object sender, EventArgs e)
25. {
26. this.Close();
27. }
28. bool falg = true;
29. private void button2\_Click(object sender, EventArgs e)
30. {
32. if (falg){
33. this.TopMost = true;
34. //this.FormBorderStyle = FormBorderStyle.None;
35. this.WindowState = FormWindowState.Maximized;
36. falg = false;
37. }
38. else
39. {
40. this.TopMost = true;
41. //this.FormBorderStyle = FormBorderStyle.None;
42. this.WindowState = FormWindowState.Normal;
43. falg = true;
44. }
45. }
46. private void button3\_Click(object sender, EventArgs e)
47. {
48. this.WindowState = FormWindowState.Minimized;
49. }
50. private void button4\_Click(object sender, EventArgs e)
51. {
52. paint form2 = new paint();
53. this.Hide();
54. form2.Show();
55. }
56. private void button5\_Click(object sender, EventArgs e)
57. {
58. help form3 = new help();
60. form3.Show();
61. }
62. }
63. }
64. namespace test
65. {

68. public partial class paint : Form
69. {
70. int r = 11;
71. Graphics g ;
72. Color col = Color.Black;
73. Figure[] mas = new Figure[0];
74. //bool[] elem = new bool[11] { false, false, false, false, false, false, false, false, false, false, false};
75. List<int> figures = new List<int>();
77. int item;
79. public paint()
80. {
81. InitializeComponent();
82. this.ControlBox = false;
83. this.button5.Click += new System.EventHandler(this.button5\_Click);
84. this.comboBox1.SelectedIndexChanged += comboBox1\_SelectedIndexChanged;
85. this.comboBox1.SelectedIndex = r;
86. this.button5.BackColor = col;
87. }
88. private void button1\_Click(object sender, EventArgs e)
89. {
90. Form1 frm1 = new Form1();
91. frm1.Show();
92. this.Close();
93. }
94. bool falg = true;
95. private void button2\_Click(object sender, EventArgs e)
96. {
97. if (falg)
98. {
99. this.TopMost = true;
100. //this.FormBorderStyle = FormBorderStyle.None;
101. this.WindowState = FormWindowState.Maximized;
102. falg = false;
103. }
104. else
105. {
106. this.TopMost = true;
107. //this.FormBorderStyle = FormBorderStyle.None;
108. this.WindowState = FormWindowState.Normal;
109. falg = true;
110. }
111. }
112. private void button3\_Click(object sender, EventArgs e)
113. {
114. this.WindowState = FormWindowState.Minimized;
115. }
116. private void button5\_Click(object sender, EventArgs e)
117. {
118. if (colorDialog1.ShowDialog() == DialogResult.OK)
119. {
120. button5.BackColor = colorDialog1.Color;
121. col = colorDialog1.Color;
122. }
123. }
124. private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)
125. {
126. r = int.Parse(comboBox1.SelectedItem.ToString());
127. }
128. private void button4\_Click(object sender, EventArgs e)
129. {
130. //var item;


134. if (checkedListBox1.CheckedItems.Count > 1)
135. {
136. MessageBox.Show("Выбрано больше одной фигуры");
137. }
138. else
139. {
140. foreach (var item1 in checkedListBox1.CheckedItems)
141. {
143. item = int.Parse(item1.ToString()[0].ToString()+ item1.ToString()[1].ToString()) -1 ;
145. }
146. }
147. }
148. private void pictureBox1\_MouseDown(object sender, MouseEventArgs e)
149. {
150. g = this.pictureBox1.CreateGraphics();
151. // if (e.Button == MouseButtons.Left)
152. //{
153. Figure s;
154. Pen brush = new Pen(col, 4);
156. switch (item)
157. {
158. case 0: //квадрат
159. //MessageBox.Show("work");
160. s = new Square(g, r, e.X, e.Y, col);
161. break;
162. case 1: //окружность
163. s = new empty\_circle(g, r, e.X, e.Y, brush);
164. break;
165. case 2: //эллипс
166. s = new ellipse(g, r, e.X, e.Y, col);
167. break;
168. case 3: //трапеция
169. s = new Trap(g, r, e.X, e.Y, col);
170. break;
171. case 4: //круг
172. s = new Circle(g, r, e.X, e.Y, col);
173. break;
174. case 5: //Прямоугольник
175. s = new Rectangle(g, r, e.X, e.Y, col);
176. break;
177. case 6: //Параллелограмм
178. s = new paral(g, r, e.X, e.Y, col);
179. break;
180. case 7: //Равнобедренный треугольник
181. s = new Triangle\_rav(g, r, e.X, e.Y, col);
182. break;
183. case 8: //Прямоугольный треуг
184. s = new Triangle\_90(g, r, e.X, e.Y, col);
185. break;
186. case 9: //Равносторонний реуг
187. //MessageBox.Show("work");
188. s = new Triangle\_classic(g, r, e.X, e.Y, col);
189. break;
190. case 10: // ромб
191. s = new romb(g, r, e.X, e.Y, col);
192. break;
193. }
194. //pictureBox1.Refresh();
195. }
196. }
197. public abstract class Figure
198. {
199. //public abstract void draw(Graphics g, int r, int x, int y, Pen brush);
200. }
201. public class Square : Figure
202. {
203. public Square(Graphics g, int r, int x, int y, Color col)
204. {
205. Brush brush = new SolidBrush(col);
206. Point p1 = new Point(x - r / 2, y + r / 2);
207. Point p2 = new Point(x + r / 2, y + r / 2);
208. Point p3 = new Point(x + r / 2, y - r / 2);
209. Point p4 = new Point(x - r / 2, y - r / 2);
210. Point[] massiv = { p1, p2, p3, p4 };
211. g.FillPolygon(brush, massiv);
212. }
213. }
214. public class ellipse : Figure
215. {
216. public ellipse(Graphics g, int r, int x, int y, Color col)
217. {
218. Brush brush = new SolidBrush(col);
219. //Point p1 = new Point(x - r / 2, y + r / 2);
220. //Point p2 = new Point(x + r / 2, y + r / 2);
222. //Point[] massiv = { p1, p2};
223. g.FillEllipse(brush, x,y,2\*r,r/2);
224. }
225. }
226. public class empty\_circle : Figure
227. {
228. public empty\_circle(Graphics g, int r, int x, int y, Pen brush)
229. {
230. g.DrawEllipse(brush, x - r, y - r, 2 \* r, 2 \* r);
231. }
232. }
233. public class Rectangle : Figure //прямоугольник
234. {
235. public Rectangle(Graphics g, int r, int x, int y, Color col)
236. {
237. Brush brush = new SolidBrush(col);
238. Point p1 = new Point(x - r , y + r / 2);
239. Point p2 = new Point(x + r , y + r / 2);
240. Point p3 = new Point(x + r , y - r / 2);
241. Point p4 = new Point(x - r , y - r / 2);
242. Point[] massiv = { p1, p2, p3, p4 };
243. g.FillPolygon(brush, massiv);
244. }
245. }
246. public class Circle : Figure
247. {
248. public Circle(Graphics g, int r, int x, int y, Color col)
249. {
250. Brush brush = new SolidBrush(col);
251. g.FillEllipse(brush, x-r,y-r,2\*r,2\*r);
252. }
253. }
254. public class Triangle\_classic : Figure
255. {
256. public Triangle\_classic(Graphics g, int r, int x, int y, Color col)
257. {
258. Brush brush = new SolidBrush(col);
259. Point p1 = new Point(x, y- Convert.ToInt32(0.866 \* ((2\*r)/3)));
260. Point p2 = new Point(x + r/2, y + Convert.ToInt32(0.866\*(r/3)));
261. Point p3 = new Point(x - r/2, y + Convert.ToInt32(0.866 \* (r/3)));
262. Point[] massiv1 = { p1, p2, p3 };
263. g.FillPolygon(brush, massiv1);
264. }
265. }
266. public class Triangle\_rav : Figure
267. {
268. public Triangle\_rav(Graphics g, int r, int x, int y, Color col)
269. {
270. Brush brush = new SolidBrush(col);
271. Point p1 = new Point(x, y - Convert.ToInt32(0.866 \* (2 \* r)));
272. Point p2 = new Point(x + r / 2, y + Convert.ToInt32(0.866 \* (r / 3)));
273. Point p3 = new Point(x - r / 2, y + Convert.ToInt32(0.866 \* (r / 3)));
274. Point[] massiv1 = { p1, p2, p3 };
275. g.FillPolygon(brush, massiv1);
276. }
277. }
278. public class Triangle\_90 : Figure
279. {
280. public Triangle\_90(Graphics g, int r, int x, int y, Color col)
281. {
282. Brush brush = new SolidBrush(col);
283. Point p1 = new Point(x - r / 2, y -r / 2);
284. Point p2 = new Point(x - r / 2, y +r / 2);
285. Point p3 = new Point(x + r / 2, y + +r / 2);
286. Point[] massiv1 = { p1, p2, p3 };
287. g.FillPolygon(brush, massiv1);
288. }
289. }
290. public class Trap : Figure
291. {
292. public Trap(Graphics g, int r, int x, int y, Color col)
293. {
294. Brush brush = new SolidBrush(col);
295. Point p1 = new Point(x - r / 2, y - r / 2);
296. Point p2 = new Point(x - r, y + r / 2);
297. Point p3 = new Point(x + r, y + r / 2);
298. Point p4 = new Point(x + r / 2, y - r / 2);
299. Point[] massiv1 = { p1, p2, p3, p4 };
300. g.FillPolygon(brush, massiv1);
301. }
302. }
303. public class paral : Figure
304. {
305. public paral(Graphics g, int r, int x, int y, Color col)
306. {
307. Brush brush = new SolidBrush(col);
308. Point p1 = new Point(x - r , y - r / 2);
309. Point p2 = new Point(x - r / 2, y + r / 2);
310. Point p3 = new Point(x + r , y + r / 2);
311. Point p4 = new Point(x + r / 2, y - r / 2);
312. Point[] massiv1 = { p1, p2, p3, p4 };
313. g.FillPolygon(brush, massiv1);
314. }
315. }
316. public class romb : Figure
317. {
318. public romb(Graphics g, int r, int x, int y, Color col)
319. {
320. Brush brush = new SolidBrush(col);
321. Point p1 = new Point(x , y - r );
322. Point p2 = new Point(x - r / 2, y );
323. Point p3 = new Point(x , y + r );
324. Point p4 = new Point(x + r / 2, y );
325. Point[] massiv1 = { p1, p2, p3, p4 };
326. g.FillPolygon(brush, massiv1);
327. }
328. }
329. }

namespace test

{

public partial class help : Form

{

public help()

{

InitializeComponent();

this.Opacity = 0.5;

}

}

}