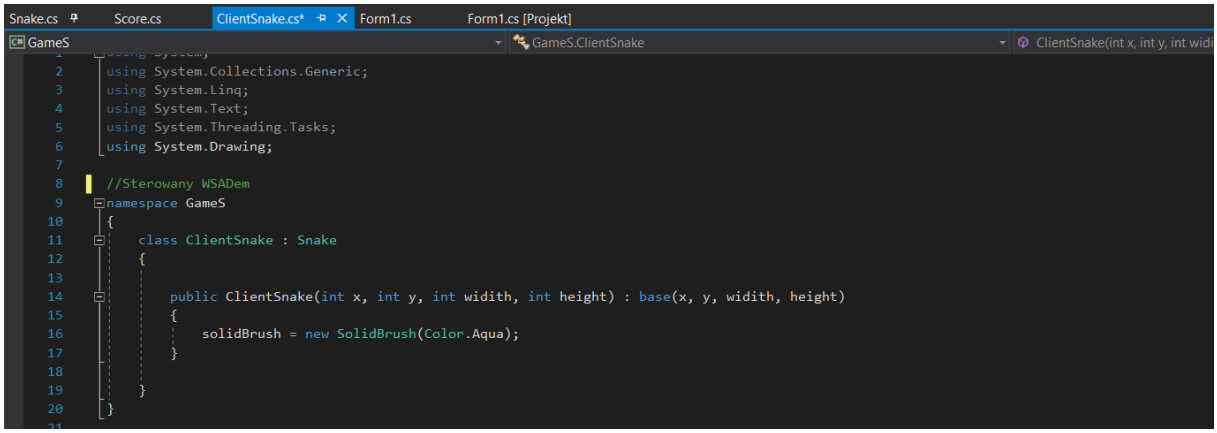
W drugim etapie rozszerzono aplikacje o dodanie nowej klasy ClientSnake która dziedziczy po Snake. W głównym oknie aplikacji dodano sterowanie obiektem clientSnake, obsługę jego „wzrostu” oraz zderzania się z innymi obiektami, również z drugim wężem. Nowy wąż jest sterowany za pomocą klawiszy WSAD.

Klasa ClientSnake:



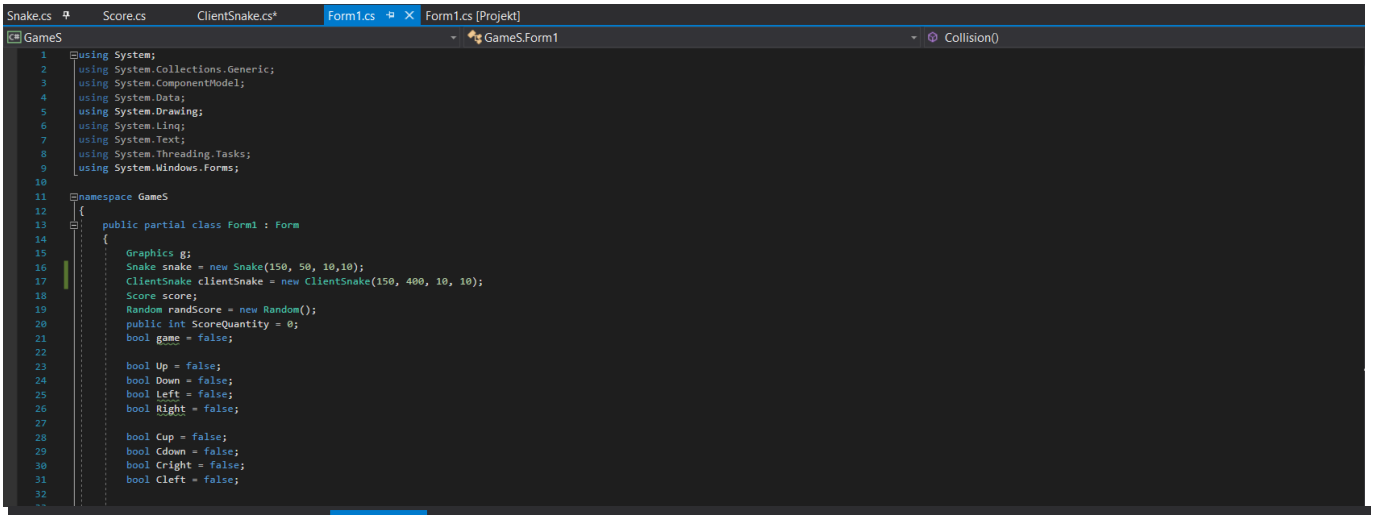
```
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;

//Sterowany WSADem
namespace GameS
{
    class ClientSnake : Snake
    {

        public ClientSnake(int x, int y, int widith, int height) : base(x, y, widith, height)
        {
            solidBrush = new SolidBrush(Color.Aqua);
        }

    }
}
```

Oraz klasa główna sterująca zdarzeniami w oknie, rozszerzona o obsługę obiektu clientSnake

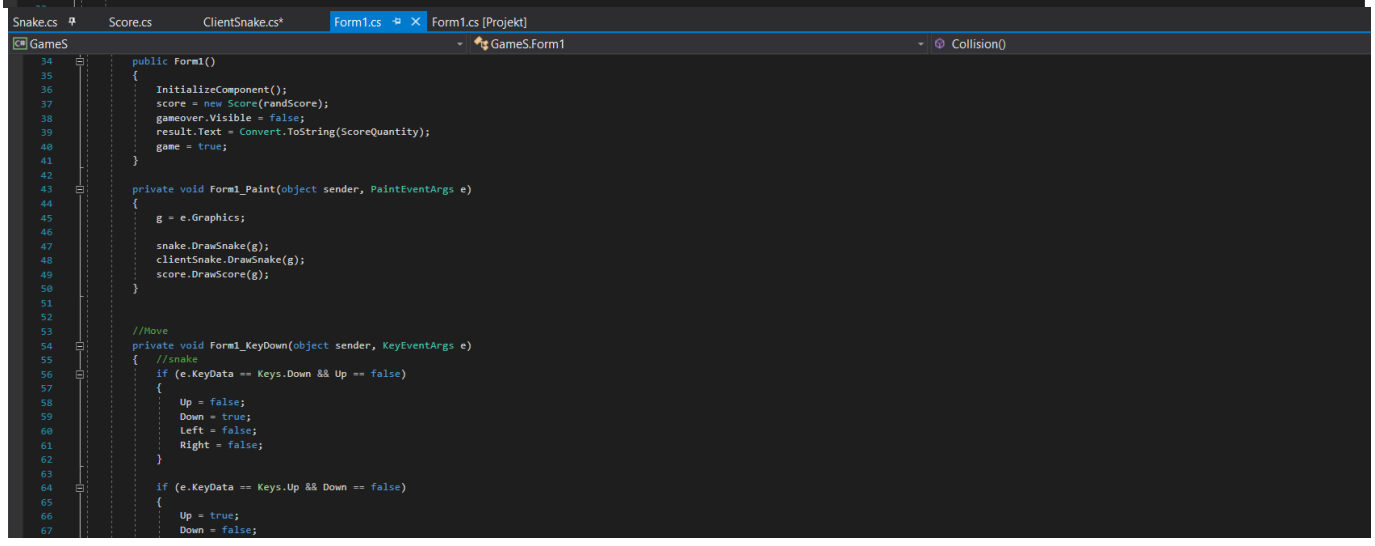


```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace GameS
{
    public partial class Form1 : Form
    {
        Graphics g;
        Snake snake = new Snake(150, 50, 10,10);
        ClientSnake clientSnake = new ClientSnake(150, 400, 10, 10);
        Score score;
        Random randScore = new Random();
        public int ScoreQuantity = 0;
        bool game = false;

        bool Up = false;
        bool Down = false;
        bool Left = false;
        bool Right = false;

        bool Cup = false;
        bool Cdown = false;
        bool Cright = false;
        bool Cleft = false;
```



```
        public Form1()
        {
            InitializeComponent();
            score = new Score(randScore);
            gameover.Visible = false;
            result.Text = Convert.ToString(ScoreQuantity);
            game = true;
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {
            g = e.Graphics;

            snake.DrawSnake(g);
            clientSnake.DrawSnake(g);
            score.DrawScore(g);
        }


        //Move
        private void Form1_KeyDown(object sender, KeyEventArgs e)
        {   //snake
            if (e.KeyData == Keys.Down && Up == false)
            {
                Up = false;
                Down = true;
                Left = false;
                Right = false;
            }

            if (e.KeyData == Keys.Up && Down == false)
            {
                Up = true;
                Down = false;
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs | Form1.cs [Projekt]

GameS | GameS.Form1 | Collision()

```
64          if (e.KeyData == Keys.Up && Down == false)
65          {
66              Up = true;
67              Down = false;
68              Left = false;
69              Right = false;
70          }
71
72          if (e.KeyData == Keys.Right && Left == false)
73          {
74              Up = false;
75              Down = false;
76              Left = false;
77              Right = true;
78          }
79
80          if (e.KeyData == Keys.Left && Right == false)
81          {
82              Up = false;
83              Down = false;
84              Left = true;
85              Right = false;
86          }
87
88          //Client Snake
89          if(e.KeyData == Keys.W && Cdown == false)
90          {
91              Cup = true;
92              Cdown = false;
93              Cright = false;
94              Cleft = false;
95          }
96
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs | Form1.cs [Projekt]

GameS | GameS.Form1 | Collision()

```
97          if (e.KeyData == Keys.S && Cup == false)
98          {
99              Cup = false;
100             Cdown = true;
101             Cright = false;
102             Cleft = false;
103         }
104
105         if (e.KeyData == Keys.A && Cright == false)
106         {
107             Cup = false;
108             Cdown = false;
109             Cright = false;
110             Cleft = true;
111         }
112
113         if (e.KeyData == Keys.D && Cleft == false)
114         {
115             Cup = false;
116             Cdown = false;
117             Cright = true;
118             Cleft = false;
119         }
120     }
121
122
123     private void timer1_Tick(object sender, EventArgs e)
124     {
125         //Snake
126         if (Up)
127         {
128             snake.MoveUp();
129         }
130         if (Down)
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs | Form1.cs [Projekt]

GameS | GameS.Form1 | Collision()

```
130         if (Down)
131         {
132             snake.MoveDown();
133         }
134         if (Right)
135         {
136             snake.MoveRight();
137         }
138         if (Left)
139         {
140             snake.MoveLeft();
141         }
142
143         //ClientSnake
144         if (Cup)
145         {
146             clientSnake.MoveUp();
147         }
148         if (Cdown)
149         {
150             clientSnake.MoveDown();
151         }
152         if (Cright)
153         {
154             clientSnake.MoveRight();
155         }
156         if (Cleft)
157         {
158             clientSnake.MoveLeft();
159         }
160
161         //speed
162         if (ScoreQuantity == 15)
163         {
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs | Form1.cs [Projekt]

GameS | GameS.Form1 | Collision()

```
160
161             //speed
162             if (ScoreQuantity == 15)
163             {
164                 timer1.Interval = 200;
165             }
166             if (ScoreQuantity == 30)
167             {
168                 timer1.Interval = 150;
169             }
170             if (ScoreQuantity == 45)
171             {
172                 timer1.Interval = 100;
173             }
174             if (ScoreQuantity == 60)
175             {
176                 timer1.Interval = 50;
177             }
178
179             this.Invalidate();
180
181             Collision();
182             Eat();
183         }
184
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs* | Form1.cs [Projekt]*

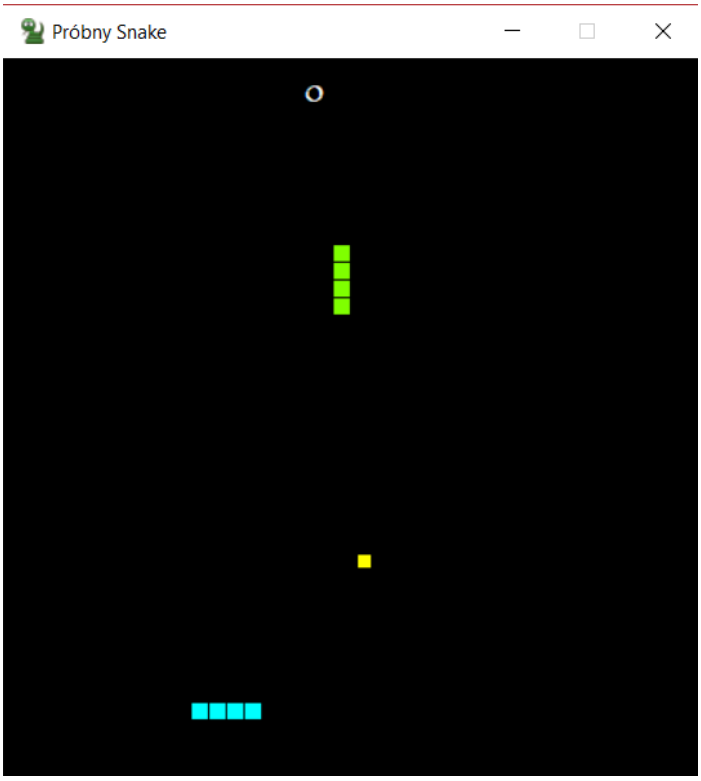
GameS | GameS.Form1 | Collision()

```
184
185         public void Collision()
186         {
187             //5 przypadków kolizji
188             //kolizja ze ścianą
189             if (snake.SnakePart[0].X <= 1 || snake.SnakePart[0].X >= 425 || clientSnake.SnakePart[0].X <= 1 || clientSnake.SnakePart[0].X >= 425)
190             {
191                 gameover.Visible = true;
192                 timer1.Stop();
193             }
194             if (snake.SnakePart[0].Y <= 1 || snake.SnakePart[0].Y >= 430 || clientSnake.SnakePart[0].Y <= 1 || clientSnake.SnakePart[0].Y >= 430)
195             {
196                 gameover.Visible = true;
197                 timer1.Stop();
198             }
199
200             //kolizja z samym sobą
201             for (int i = 1; i < snake.SnakePart.Length; i++)
202             {
203                 if (snake.SnakePart[0].IntersectsWith(snake.SnakePart[i]))
204                 {
205                     gameover.Visible = true;
206                     timer1.Enabled = false;
207                 }
208             }
209
210             for (int i = 1; i < clientSnake.SnakePart.Length; i++)
211             {
212                 if (clientSnake.SnakePart[0].IntersectsWith(clientSnake.SnakePart[i]))
213                 {
214                     gameover.Visible = true;
215                     timer1.Enabled = false;
216                 }
217             }
```

Snake.cs | Score.cs | ClientSnake.cs* | Form1.cs* | Form1.cs [Projekt]*

GameS | GameS.Form1 | Collision()

```
220
221             for (int i = 1; i < snake.SnakePart.Length; i++)
222             {
223                 if (snake.SnakePart[0].IntersectsWith(clientSnake.SnakePart[i]))
224                 {
225                     gameover.Visible = true;
226                     timer1.Enabled = false;
227                 }
228             }
229
230
231         }
232
233         public void Eat()
234         {
235             if (snake.SnakePart[0].IntersectsWith(score.ScoreRectangle))
236             {
237                 ScoreQuantity += 1;
238                 result.Text = Convert.ToString(ScoreQuantity);
239                 snake.GrowUp(ScoreQuantity);
240
241                 score.ScoreLocation(randScore);
242
243             }
244
245             if (clientSnake.SnakePart[0].IntersectsWith(score.ScoreRectangle))
246             {
247                 ScoreQuantity += 1;
248                 result.Text = Convert.ToString(ScoreQuantity);
249                 clientSnake.GrowUp(ScoreQuantity);
250
251                 score.ScoreLocation(randScore);
252
253             }
```


Próbny Snake


Próbny Snake
GAME OVER