Практическая работа №7

Разработать программный продукт, реализующий функционал игры «Корзинка». В проекте необходимо присутствие платформы, которая способна совершать движения по горизонтали.

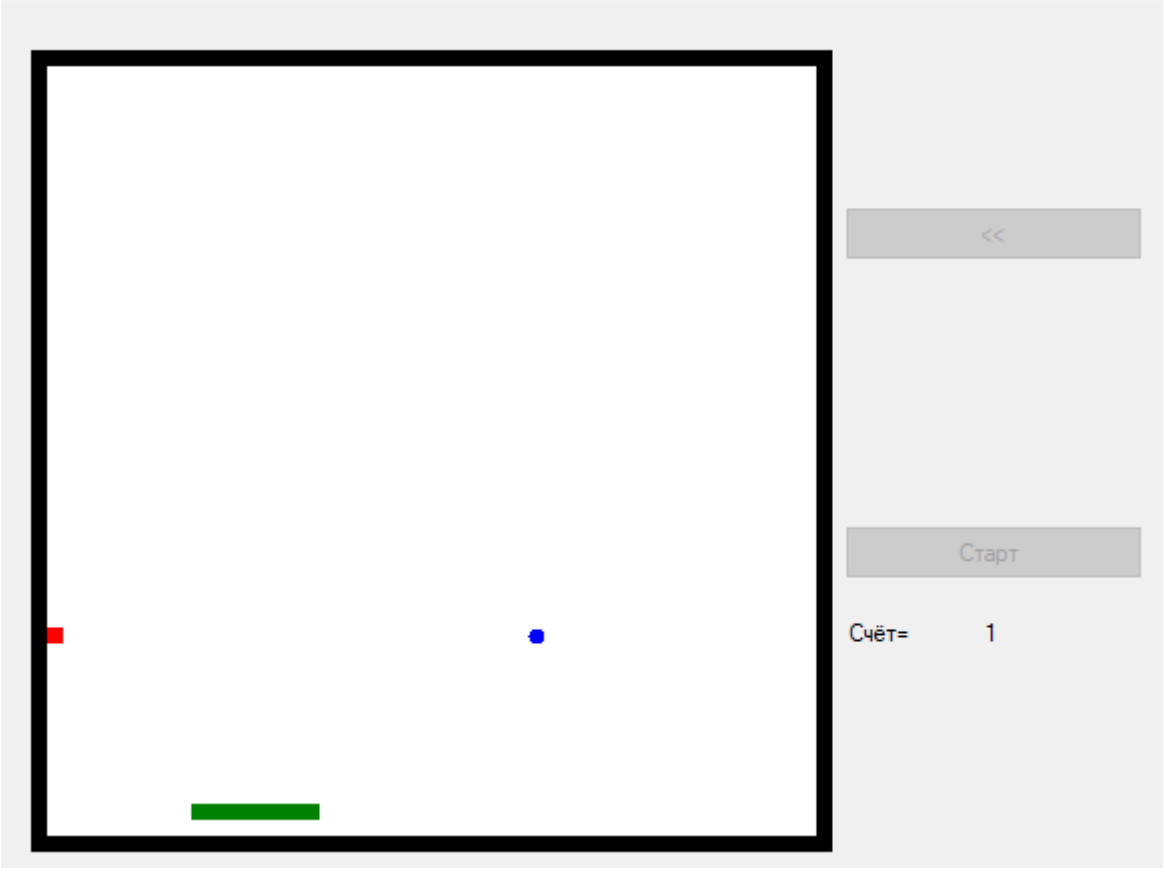
Также необходимо реализовать присутствие объектов, появляющихся вверху игровой области и совершающих движения вертикально. При взаимодействии донных объектов с платформой они исчезают из игровой области, а игрок получает очки.

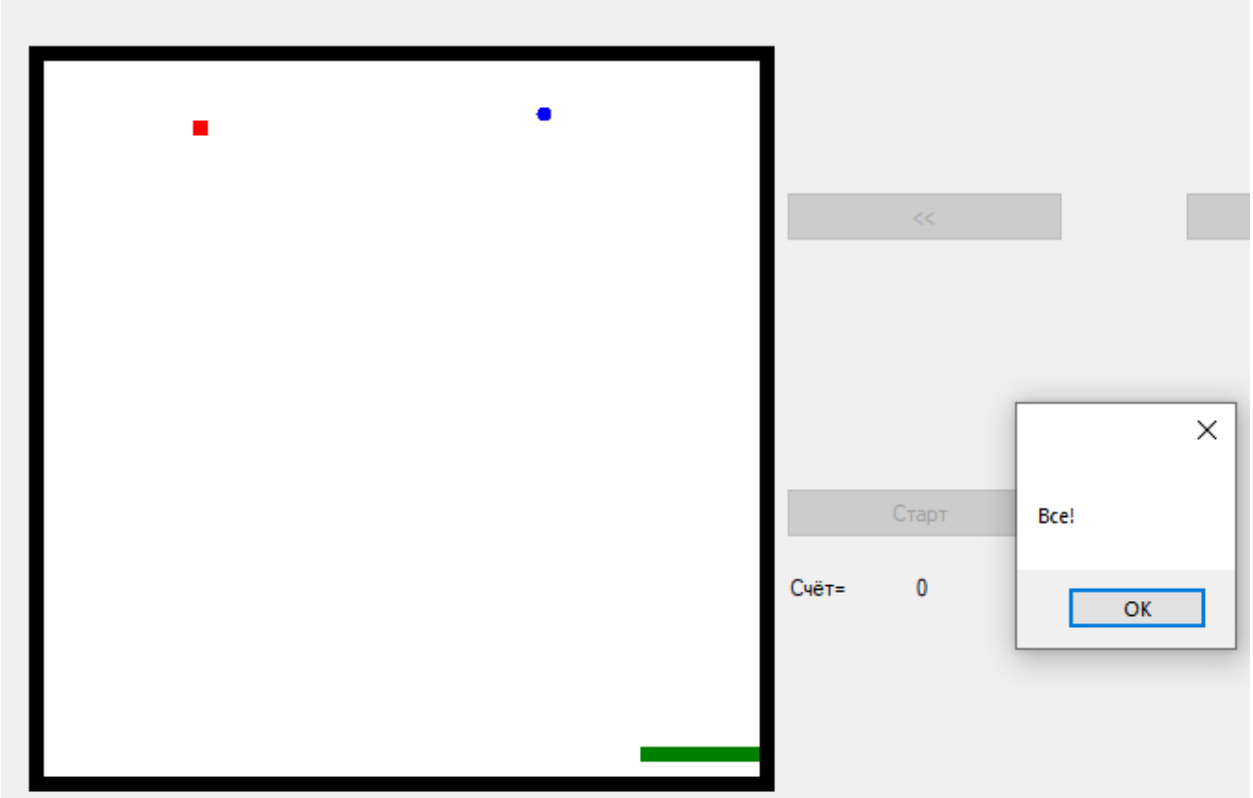
Добавить объекты противоположного назначения- при взаимодействии с платформой они отнимают очки.

Игра заканчивается если очки становятся меньше нуля или игрок не успевает поймать объект, прибавляющий баллы.

Примерный вид программного продукта представлен на рисунке 1.

Управление движением платформы необходимо осуществить с клавиатуры





Form1.cs

```
public partial class Form1 : Form
    {
        Game g;
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            pictureBox1.Image = new Bitmap(pictureBox1.Width, pictureBox1.Height);
            this.Focus();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (g.Move(1) == false)
            {
                MessageBox.Show("Конец игры!");

            }

        }

        private void button4_Click(object sender, EventArgs e)
        {
            if (g.Move(2) == false)
            {
                MessageBox.Show("Конец игры!");

            }
            label2.Text = g.count.ToString();
        }

        private void button5_Click(object sender, EventArgs e)
```

```
{
    g = new Game(pictureBox1);
    button5.Enabled = false;
    timer1.Start();
}

private void button2_Click(object sender, EventArgs e)
{
    g.Move(4);


}

private void button3_Click(object sender, EventArgs e)
{
    g.Move(3);

}

private void timer1_Tick(object sender, EventArgs e)
{
    g.MoveFood2();
    bool f=g.MoveFood();
    label2.Text = g.count.ToString();
    if (f == false)
    {
        timer1.Stop();
        MessageBox.Show("Все!");

    }
}

private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Up)
    {
        if (g.Move(1) == false)
        {
            MessageBox.Show("Все!");
            button5.Enabled = true;
            label2.Text = "0";
        }
        else label2.Text = g.count.ToString();
    }
    else if (e.KeyCode == Keys.Down)
    {
        if (g.Move(2) == false)
        {
            MessageBox.Show("Все!");
            button5.Enabled = true;
            label2.Text = "0";
        }
        else label2.Text = g.count.ToString();
    }
    else if (e.KeyCode == Keys.Right)
    {
        if (g.Move(3) == false)
        {
            MessageBox.Show("Все!");
            button5.Enabled = true;
            label2.Text = "0";
        }
        else label2.Text = g.count.ToString();
    }
    else if (e.KeyCode == Keys.Left)
```

```
            {
                if (g.Move(4) == false)
                {
                    MessageBox.Show("Все!");
                    button5.Enabled = true;
                    label2.Text = "0";
                }
                else label2.Text = g.count.ToString();
            }
        }
    }
```

MyPoint.cs

```
public class MyPoint
    {
        public int x, y;

        public MyPoint(int n_x, int n_y)
        {
            x = n_x;
            y = n_y;
        }
    }
```

Snake.cs

```
public class Snake : ParentClass
    {
        public int speed = 1;

        public Snake(Color n_color) : base(n_color)
        {
            //color=n_color;
        }
        public void AddHvost() //добаввление хвоста
        {
            int n = body.Count - 1;
            if ((body[n].x - body[n - 1].x) == 0)
            {
                body.Add(new MyPoint(body[n].x, body[n].y + 1));
            }
            else
            {
                body.Add(new MyPoint(body[n].x + 1, body[n].y));
            }
        }
        public void Move(int dx, int dy)  // движение змейки
        {
            //if (!(body[0].x - 1 < 0 || body[body.Count - 1].x + 1 > 50))
           // {
                if (dx == -1)
                {
                    if (body[0].x - 1 >= 0)
                    {

                        for (int i = body.Count - 1; i > 0; i--)
                        {
                            body[i].x = body[i - 1].x;
                            body[i].y = body[i - 1].y;
                        }
                        body[0].x = body[0].x + dx;
                        body[0].y = body[0].y + dy;
                    }
                }
```

```
                else
                {
                    if (body[body.Count - 1].x + 1 < 49)
                    {
                        for (int i = 0; i < body.Count - 1; i++)
                        {
                            body[i].x = body[i + 1].x;
                            body[i].y = body[i + 1].y;
                        }
                        body[body.Count - 1].x = body[body.Count - 1].x + dx;
                        body[body.Count - 1].y = body[body.Count - 1].y + dy;
                    }
                }
            }
       // }

    }
```

ParentClass.cs

```
public class ParentClass
    {
        public Color color;
        public List<MyPoint> body = new List<MyPoint>();

        public ParentClass(Color n_color)
        {
            color = n_color;
        }
        public void Add(MyPoint p)
        {
            body.Add(p);
        }
    }
```

Food.cs

```
public class Food:ParentClass
    {

            public Food(Color n_color) : base(n_color)
            {

            }
            public void Delete(int t)//удаление еды с экрана
            {
                body.RemoveAt(t);
            }
            public MyPoint RandomFood(int sx, int sy)//случайное появление еды
            {
                Random r = new Random();
                MyPoint a = new MyPoint(r.Next(0, sx), 0);
                return a;
            }


       public void Move()
        {
            for (int i = 0; i < body.Count; i++)
            {

                body[i].y = body[i].y+1;
            }
       }
    }
```

Game.cs

```
public class Game
    {
        Snake my_snake = new Snake(Color.Green);
        Food my_food = new Food(Color.Red);
        Food my_food2 = new Food(Color.Blue);
        Block my_block = new Block(Color.Black);

        public int count = 0;
        int l = 50;// Масштаб одного блока
        Color pole = Color.White;
        PictureBox myPb;

        public Game(PictureBox n_myPb)
        {
            myPb = n_myPb;
            LoadLevel(1);
            RandomFood();

            Show();
        }

        public void LoadLevel(int n)
        {
            StreamReader sr = new StreamReader(n.ToString() + ".txt");
            int str = 0;
            while (!sr.EndOfStream)
            {
                string s = sr.ReadLine();
                for (int i = 0; i < s.Length; i++)
                {
                    if (s[i] == '*')
                    {
                        my_block.Add(new MyPoint(i, str));

                    }
                    else if (s[i] == '$')
                    {
                        my_snake.Add(new MyPoint(i, str));

                    }
                }
                str++;
            }
            sr.Close();
        }
        public void EdaColision()
        {
            for (int i = 0; i < my_food2.body.Count; i++)
            {
                for (int j = 0; j < my_snake.body.Count; j++)
                {

                    if (my_food2.body[i].x == my_snake.body[j].x && my_food2.body[i].y ==
my_snake.body[j].y)
                    {
                        my_food2.Delete(i);
                        count--;
                        //my_snake.AddHvost();
                        RandomFood();

                    }
                }
            }
```

```
            for (int i = 0; i < my_food.body.Count; i++)
            {
                for (int j = 0; j < my_snake.body.Count; j++)
                {

                    if (my_food.body[i].x == my_snake.body[j].x && my_food.body[i].y ==
my_snake.body[j].y)
                    {
                        my_food.Delete(i);
                        count++;
                        //my_snake.AddHvost();
                        RandomFood();

                    }
                }
            }
        }
        public void RandomFood()
        {

            MyPoint p = my_food.RandomFood(l, l);
            my_food.Add(p);
            Thread.Sleep(100);
            MyPoint r = my_food2.RandomFood(l, l);
            my_food2.Add(r);
            Thread.Sleep(100);
        }
        public void Show()
        {
            int k = myPb.Width / l;
            Graphics gr = Graphics.FromImage(myPb.Image);
            SolidBrush cl = new SolidBrush(pole);
            gr.FillRectangle(cl, 0, 0, myPb.Width, myPb.Height);
            SolidBrush color_snake1 = new SolidBrush(Color.Purple);
            SolidBrush color_snake = new SolidBrush(my_snake.color);
            for (int i = 0; i < my_snake.body.Count; i++)
            {
                /*if (i == 0)
                {
                    gr.FillRectangle(color_snake1, my_snake.body[i].x * k,
my_snake.body[i].y * k, k, k);
                }
                else*/
                {
                    gr.FillRectangle(color_snake, my_snake.body[i].x * k,
my_snake.body[i].y * k, k, k);
                }

            }
            SolidBrush color_block = new SolidBrush(my_block.color);
            for (int i = 0; i < my_block.body.Count; i++)
            {
                gr.FillRectangle(color_block, my_block.body[i].x * k, my_block.body[i].y
* k, k, k);
            }
            SolidBrush color_food = new SolidBrush(my_food.color);
            for (int i = 0; i < my_food.body.Count; i++)
            {
                gr.FillRectangle(color_food, my_food.body[i].x * k, my_food.body[i].y *
k, k, k);
            }
            SolidBrush color_food2 = new SolidBrush(my_food2.color);
            for (int i = 0; i < my_food2.body.Count; i++)
            {
```

```
                gr.FillEllipse(color_food2, my_food2.body[i].x * k, my_food2.body[i].y *
k, k, k);
            }

            myPb.Refresh();

        }
        public bool Move(int k)
        {
            bool a = true;
            if (k == 1)
            {
                my_snake.Move(0, -1);
            }
            else if (k == 2)
            {
                my_snake.Move(0, 1);

            }
            else if (k == 3)
            {
                my_snake.Move(1, 0);
            }
            else if (k == 4)
            {
                my_snake.Move(-1, 0);
            }
            EdaColision();
            a = Crush();
            if (a == true)
            {
                Show();
            }

            return a;

        }
        public bool MoveFood2()
        {
            my_food2.Move();
            EdaColision();
            bool f = true;
            for (int i = 0; i < my_food2.body.Count; i++)
            {
                if (my_food2.body[i].y > l)
                {
                    f = false;
                }
            }

            Show();
            return f;

        }

        public bool MoveFood()
        {
            my_food.Move();
            EdaColision();
            bool f = true;
            for (int i=0; i< my_food.body.Count; i++)
            {
                if (my_food.body[i].y > l)
                {
                    f = false;
```

```
                }
            }

            Show();
            return f;
        }
        public bool Crush()
        {
            bool t = true;
            for (int i = 0; i < my_block.body.Count; i++)
            {
                if (my_snake.body[0].x == my_block.body[i].x && my_snake.body[0].y ==
my_block.body[i].y)
                {
                    t = false;
                    break;
                }
            }
            for (int i = 1; i < my_snake.body.Count; i++)
            {
                if (my_snake.body[0].x == my_snake.body[i].x && my_snake.body[0].y ==
my_snake.body[i].y)
                {
                    t = false;
                    break;
                }
            }
            return t;
        }
    }
```

Block.cs

```
public class Block : ParentClass//класс для препядствия
    {
        public Block(Color n_color) : base(n_color)
        {

        }
    }
```

Food2.cs

```
public class Food2:ParentClass
    {
        public Food2(Color n_color) : base(n_color)
        {

        }
        public void Delete() //удаление еды с экрана
        {

        }
        public MyPoint RandomFood(int sx, int sy)// Случайное появление еды
        {
            Random r = new Random();
            MyPoint a = new MyPoint(r.Next(0, sx), 0);
            return a;
        }
        public void Move()
        {
            for (int i = 0; i < body.Count; i++)
            {

                body[i].y = body[i].y + 1;
```

```
            }
        }
    }
```

Ссылка на гитхаб:

https://github.com/Alexandrov911/Practical7.2022.git