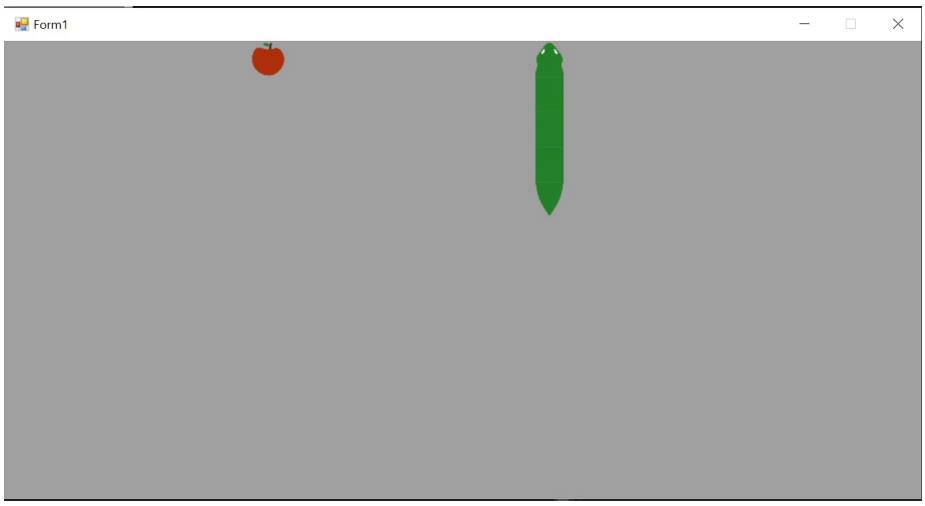
Sprawozdanie do zad. 6. w ramach laboratorium Organizacja Systemów Komputerowych

Paweł Szczepański (188640), Igor Szczyrbak (188661)

19 czerwca 2024

1 Założenia programu

Aplikacja jest implementacją gry zręcznościowej "Snake". Gra jest obsługiwana przy pomocy klawiatury (strzałek). Gracz wybiera kierunek ruchu węża kierując go w stronę owoców i unikając granic oraz ciała węża. Za każdy zebrany owoc gracz zdobywa punkt, a wąż się wydłuża. Gra odbywa się się na siatce 26x13 pól i aktulizuje się co 150 milisekund.



Rysunek 1: Okno główne programu

2 Rozwiązania programowe

2.1 Siatka gry

Stan gry jest zapisywany w obiekcie klasy Board zawierającym dwuwymiarowe tablice zapisujące stan pola oraz kierunek pola. Stan pola decyduje o tym co

znajduje się na danym polu: głowa, ciało, ogon węża, owoc lub nic. Kierunek pola decyduje o tym w jakim kierunku (góra, prawo, dół, lewo) poruszać się będą części węża znajdujące się na danym polu.

```
public class Board
       {
           const int maxX = 26, maxY = 13, cellSize = 35;

           cellState[,] prevCells = new cellState[maxX, maxY];
           cellState[,] cells = new cellState[maxX, maxY];
           direction[,] prevDirs = new direction[maxX, maxY];
           direction[,] dirs = new direction[maxX, maxY];

           int headX = 1, headY = 0;
           int tailX, tailY;
           int growing = 1;
           public int score = 0;
           int bodyIndex = 0;

           Form1 parent;

           public Board(Form1 newParent)
           {
               parent = newParent;

               tailX = headX - 1;
               tailY = headY;

               //Inicjalizacja stanu planszy

               cells[headX, headY] = cellState.head;
               cells[tailX, tailY] = cellState.tail;
               dirs[headX, headY] = direction.right;
               dirs[tailX, tailY] = direction.right;

               SpawnFruit();
           }
```

2.2 Ruch węża

Aktualizacja stanu gry jest wywyływana przez timer o intervale 150ms. Program sprawdza wszystkie pola planszy i jeżeli natrafi na:

Ciało węża komórka wskazywana przez kierunek danego pola jest wypełniania ciałem węża.

Ogon węża komórka wskazywana przez kierunek danego pola jest wypełniania ogonem węża, a komórka poprzednio przez niego zajmowana jest ustawiana jako pusta.

Głowa węża komórka wskazywana przez kierunek danego pola jest wypełniania głową wężą i jej kierunek jest zmieniany na kierunek ruchu głowy węża. Sprawia to, że głowa pozostawia po sobie "ślad"kierunków po którym może poruszać się reszta węża.

```
void MoveCell(int x, int y, direction dir)
{
    int newX = x;
    int newY = y;
    switch (dir)
    {
        case direction.up:
            newY--;
            break;
        case direction.right:
            newX++;
            break;
        case direction.down:
            newY++;
            break;
        case direction.left:
            newX--;
            break;

    }
    if (newY >= 0 && newY < maxY && newX >= 0 && newX < maxX)
    {
        cells[newX, newY] = prevCells[x, y];

        if (prevCells[x, y] == cellState.tail)
        {
            else
            {
                cells[newX, newY] = cellState.body;
                growing--;
                AddBody(newX, newY, prevDirs[x, y],
                    prevDirs[newX, newY]);
            }
        }
        else if (prevCells[x, y] == cellState.head)
        {
            dirs[x, y] = dir;
            dirs[newX, newY] = dir;
            headX = newX;
            headY = newY;

            head.Location = new Point(cellSize * newX,
                cellSize * newY);
        }
```

```
                else if (prevCells[x, y] == cellState.body)
                {

                    body[bodyIndex].Location = new Point(cellSize *
                        newX, cellSize * newY);
                    bodyIndex++;
                }
            }
        }
```

Zmiana kierunku odbywa się poprze zmianę kierunku pola zajmowanego przez głowę i jest wywływane naciśnięciem klawisza. Niemożliwa jest zmiana kierunku ruchu na odwrotny w jednym cyklu.

```
private void Form1_KeyDown(object sender, KeyEventArgs e)
{
    switch(e.KeyCode)
    {
        case Keys.Up:
            board.ChangeDirection(direction.up);
            break;
        case Keys.Right:
            board.ChangeDirection(direction.right);
            break;
        case Keys.Down:
            board.ChangeDirection(direction.down);
                    break;
        case Keys.Left:
            board.ChangeDirection(direction.left);
            break;
    }
}

    public void ChangeDirection(direction dir)
    {
        direction prevDir = dirs[headX, headY];
        if (!(prevDir == direction.up && dir == direction.down) &&
            !(prevDir == direction.right && dir == direction.left) &&
            !(prevDir == direction.down && dir == direction.up) &&
            !(prevDir == direction.left && dir == direction.right))
        {
            prevDirs[headX, headY] = dir;
        }
    }
```

2.3 Kondycje końca gry i zliczanie punktów

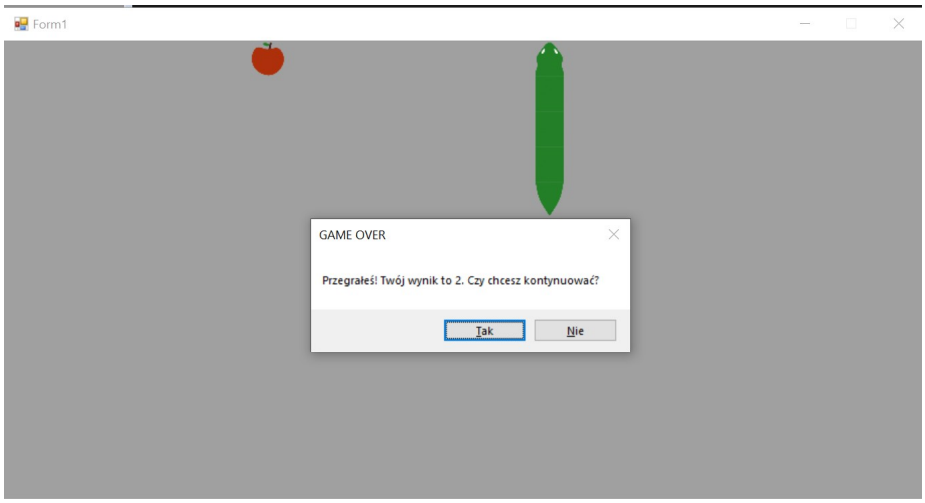
Gra się kończy, gdy głowa węża próbuje się poruszyć poza planszę, lub na pole zajmowane przez ciało lub ogon.

```
if (newY >= 0 && newY < maxY && newX >= 0 && newX < maxX)
{
    .
    .
    .
    else if (prevCells[x, y] == cellState.head)
    {
        if (prevCells[newX, newY] == cellState.body ||
            prevCells[newX, newY] == cellState.tail)
           parent.GameOver();
        .
        .
        .
    }
}
else if (prevCells[x, y] == cellState.head)
    parent.GameOver();
```

Punkty są naliczne, gdy głowa porusza się na pole zajmowane przez owoc. Powoduje to również utworzenie nowego owocu na losowym wpustym polu i inkrementację *growing*. Dopóki *growing* jest niezerowe ogon zamiast się przesuwać będzie tworzył nowe segmenty ciała i dekrementował *growing*.

```
if (prevCells[x, y] == cellState.tail)
{
    if(growing <= 0)
    {
        cells[x, y] = cellState.empty;
        tail.Location = new Point(cellSize * newX, cellSize * newY);
    }
    else
    {
        cells[newX, newY] = cellState.body;
        growing--;
        AddBody(newX, newY, prevDirs[x, y], prevDirs[newX, newY]);
    }
}
else if (prevCells[x, y] == cellState.head)
{
    if (prevCells[newX, newY] == cellState.body || prevCells[newX,
        newY] == cellState.tail)
       parent.GameOver();
    else if (prevCells[newX, newY] == cellState.fruit)
    {
        score++;
```

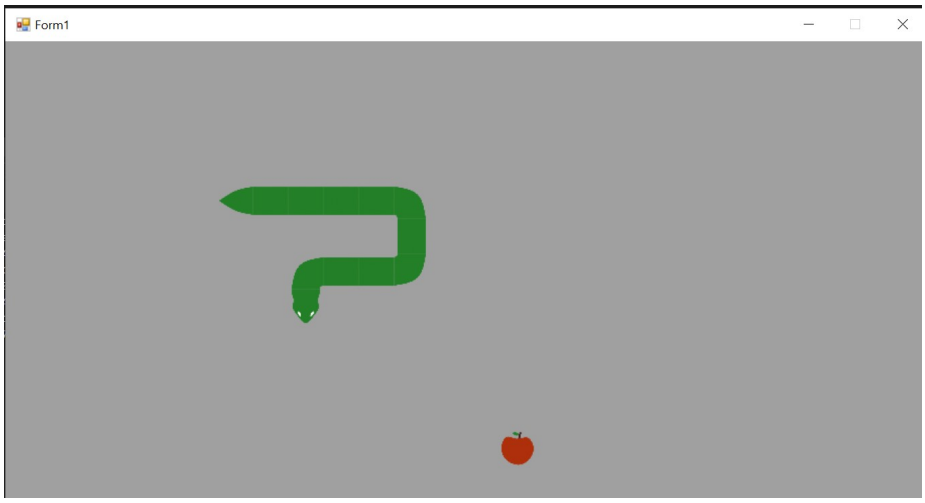
```
            growing++;
            SpawnFruit();
        }

        dirs[newX, newY] = dir;
        headX = newX;
        headY = newY;

        head.Location = new Point(cellSize * newX, cellSize * newY);
    }
```



Rysunek 2: Ekran końca gry

2.4 Grafika

Każdy z segmentów węża jest oddzielnym obiektem PictureBox. Nowe kopie segmentu ciała są tworzone według potrzeby. Po wyznaczeniu nowego stanu planszy każdy z PictureBoxów jest ustawiany w pozycji $(x \cdot 35, y \cdot 35)$, gdzie x i y to koordynaty pola odpowiadającego danemu obiektowi, a 35 to rozmiar pola w pikselach. W zależności od poprzedniego i obecneg kierunku pola wybieru jest odpowiedni obrazek.



Rysunek 3: Prezentacja różnych obrazków w zależności od kierunku ruchu segmentu

3 Dyskusja osiągniętych wyników

Program spełnia założenia projektowe. Umożliwia on jednemu graczowi grę w "Snake'a". Program również zawiera logikę umożliwiającą kontekstowe dobieranie obrazów.