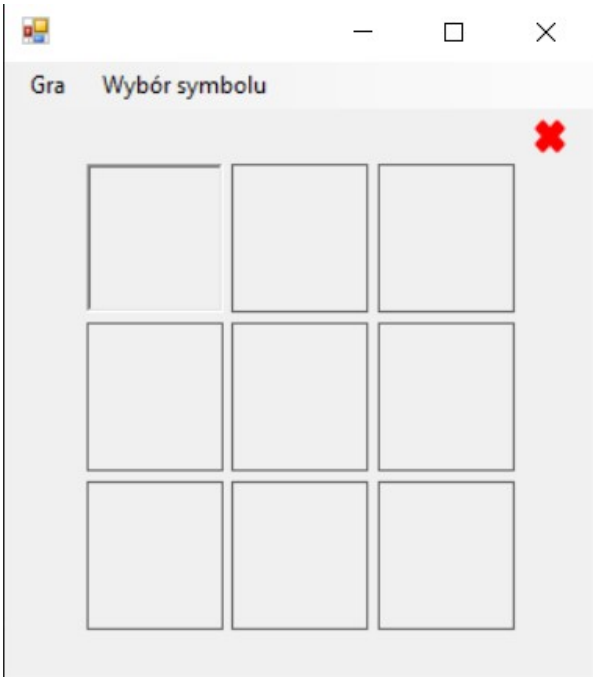
Sprawozdanie do zad. 1 w ramach laboratorium Organizacja Systemów Komputerowych.

Paweł Szczepański (188640), Igor Szczyrbak (188661)

9 kwietnia 2024

1 Założenia programu

Program jest implementacją gry w kółko i krzyżyk dla dwójki graczy. Umożliwia grę przy pomocy myszki i klawiatury. Gracze naprzemiennie wybierają pola klikając je przy pomocy myszki, lub wybierając je poprzez klawisze strzałek i potwierdzając klawiszem Enter. Wybrane pole ma ustawianą inną ramkę. Rozwijane menu pozwala na: reset stanu gry, zamknięcie programu wybór grafiki dla kółek u krzyżyków. O tym, który gracz wykonuje obecnie ruch, informuje obrazek w prawym górnym rogu okna. Źródłem grafik jest strona game-icons.net.



Rysunek 1: Program po uruchomieniu

2 Rozwiązania programowe

2.1 Podstawowa logika programu

Siatce obiektów (0,0 dla górnego lewego i 2,2 dla prawego dolnego) typu PictureBox odpowiada tablica stanów grid przyjmujące wartości empty, cross lub circle.

```
public enum cellState
```

```
{
    empty,
    cross,
    circle
}
public cellState[,] grid;
```

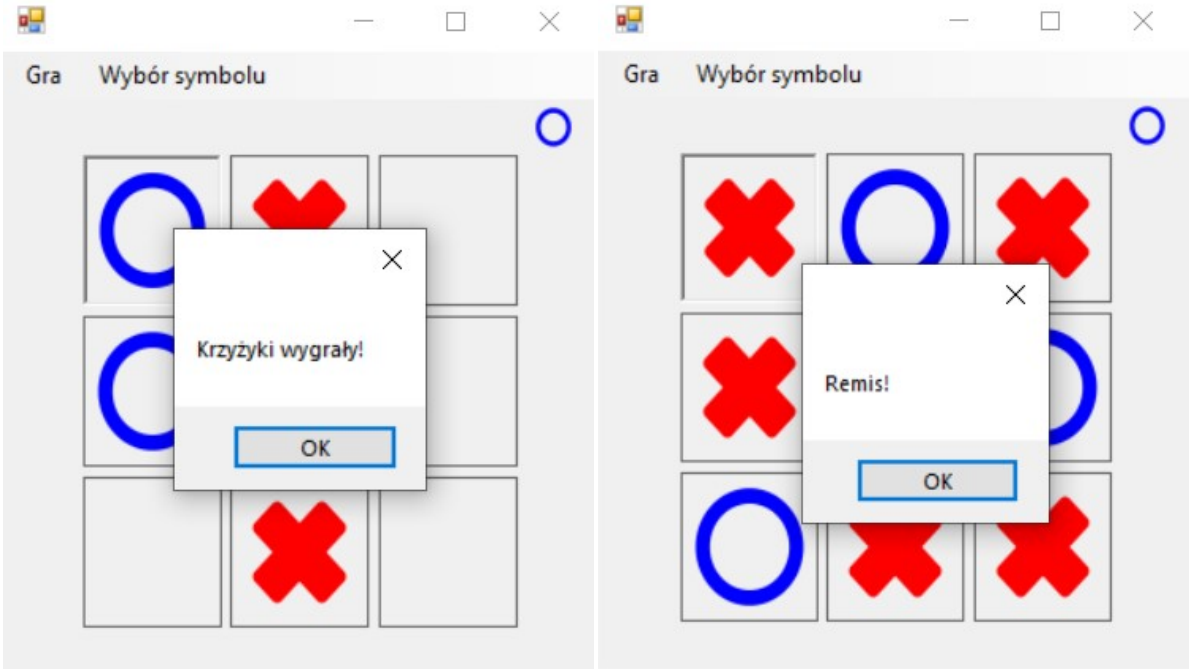
Po kliknięciu na odpowiedniego elementu interfejsu PictureBox, uruchamia się funkcja aktualizująca stan gry. Obiekty typu PictureBox są ułożone wewnątrz elementu interfejsu typu TableLayouPanel, co pozwala na znalezienie indeksu na podstawie referencji do klikniętego obiektu (argument sender). Następnie program sprawdza, czy wartość tablicy grid pod danym indeksem jest inna niż empty. Jeżeli tak, to funkcja jest przerywana. Jeżeli nie, wartość ta jest ustawiana na cross lub circle w zależności od tego, który z graczy wykonał ruch. Na koniec sprawdzany jest warunek zwycięstwa i zmieniany jest aktywny gracz.

```
private void tile_Click(object sender, EventArgs e)
{
    PictureBox img = (PictureBox)sender;
    TableLayoutPanelCellPosition pos = tableLayoutPanel1.GetPositionFromControl(img);

    if (grid[pos.Column, pos.Row] != cellState.empty)
        return;

    turns += 1;
    circleTurn = !circleTurn;
    if (circleTurn)
        infoPicture.BackgroundImage = chosenCircle.Image;
    else
        infoPicture.BackgroundImage = chosenCross.Image;

    if (circleTurn)
    {
        img.BackgroundImage = chosenCross.Image;
        grid[pos.Column, pos.Row] = cellState.cross;
        CheckWin(pos.Column, pos.Row, cellState.cross);
    }
    else
    {

        img.BackgroundImage = chosenCircle.Image;
        grid[pos.Column, pos.Row] = cellState.circle;
        CheckWin(pos.Column, pos.Row, cellState.circle);
    }

    if (turns == 9)
    {
        DialogResult result = MessageBox.Show("Remis!");
        Reset();
    }
}
```

2.2 Sprawdzanie warunku zwycięstwa

Warunek zwycięstwa jest sprawdzany przy każdym postawieniu nowej figury. Sprawdzane jest, czy wszystkie pola w zmienionych kolumnie i wierszu mają identyczny stan i jeżeli zmienione pole znajduje się na przekątnej, czy identyczny stan mają pola leżące na przekątnych. Równocześnie, licznik turns zlicza liczbę wykonanych dotychczas ruchów. Gdy minie dziewięć tur i żaden z graczy nie wygra program wykryje remis.

```
private void CheckWin(int x, int y, cellState state)
{
    //Sprawdzamy linię poziomą
    bool win = true;
    for (int i = 0; i < 3; i++)
    {
        if (grid[x, i] != state)
            win = false;
    }
    if (win)
        Win();

    //Sprawdzamy linię pionową
    win = true;
    for (int i = 0; i < 3; i++)
    {
        if (grid[i, y] != state)
            win = false;
    }
    if (win)
        Win();

    //Sprawdzamy po skosie
    if ((x + y) % 2 != 0)
        return;

    win = true;
    for (int i = 0; i < 3; i++)
    {
        if (grid[i, i] != state)
            win = false;
    }
    if (win)
        Win();

    win = true;
    for (int i = 0; i < 3; i++)
    {
        if (grid[2 - i, i] != state)
            win = false;
    }
    if (win)
        Win();
}

private void Win()
{
    string playerName;
    if (circleTurn)
        playerName = "Krzyżyki";
    else
        playerName = "Kółka";

    DialogResult result = MessageBox.Show(playerName + " wygrały!");
    Reset();
}
```



(a) Ekran zwycięstwa

(b) Ekran remisu

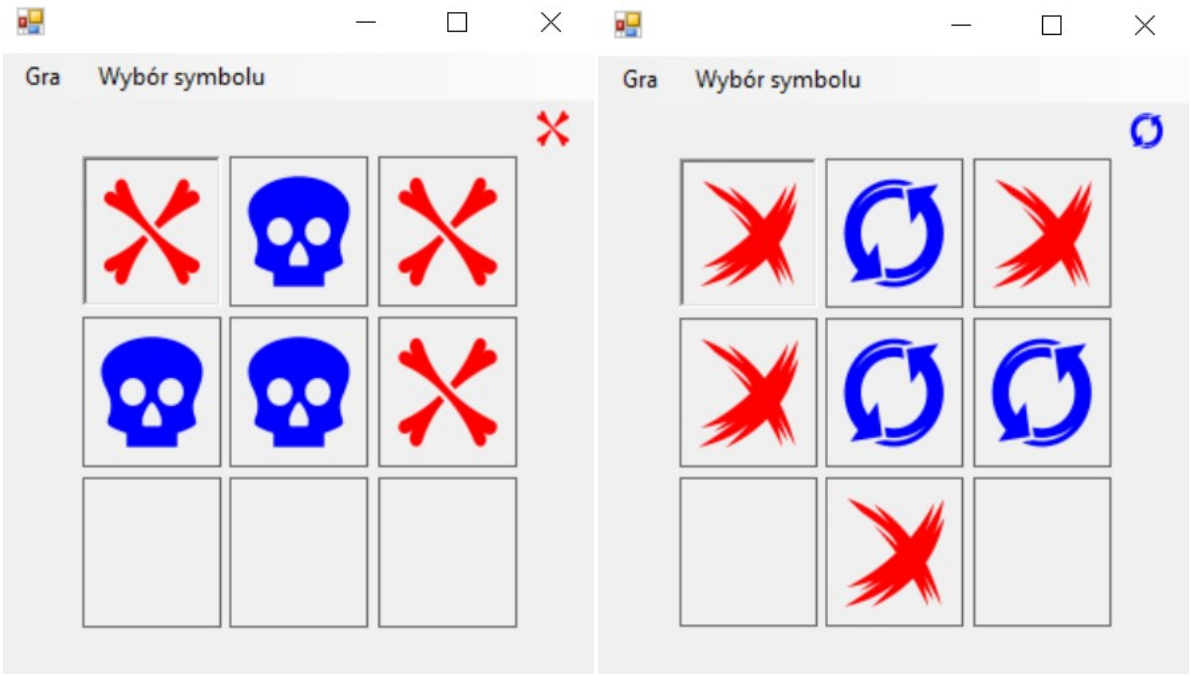
Rysunek 2: Działanie omówionego programu

2.3 Zmiana skórek

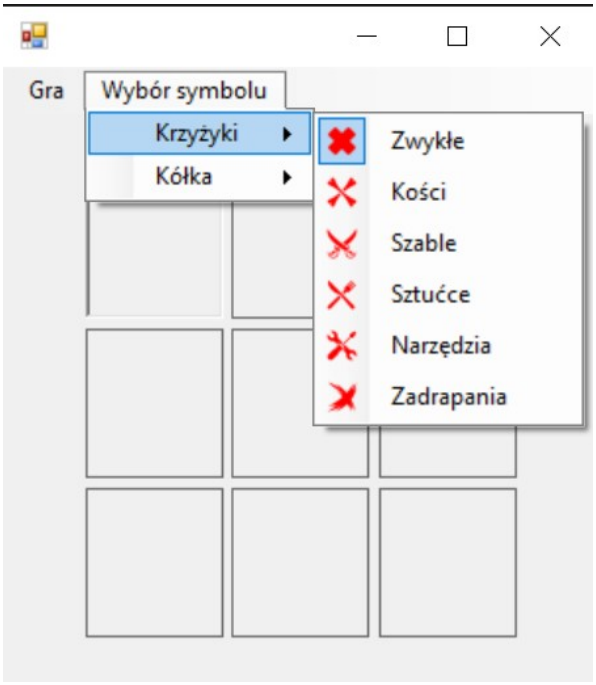
Zmiana wykorzystywanych obrazków działa przy pomocy zmiennych chosenCross i chosenCircle zapisujących referencje do obiektów typu MenuItem wybieranych z listy. Zmienne te są używane przy zmianie wartości BackgroundImage obiektów typu PictureBox – ustawiany jest obraz przypisany obiektowi MenuItem.

```
        private void cross_Click(object sender, EventArgs e)
    {
        chosenCross.Checked = false;
        chosenCross = (ToolStripMenuItem)sender;
        chosenCross.Checked = true;

        if (!circleTurn)
            infoPicture.BackgroundImage = chosenCross.Image;

        for (int x = 0; x < 3; x++)
        {
            for (int y = 0; y < 3; y++)
            {
                if (grid[x, y] == cellState.cross)
                {
                    PictureBox img =
                        (PictureBox)tableLayoutPanel1.GetControlFromPosition(x, y);
                    img.BackgroundImage = chosenCross.Image;
                }
            }
        }
    }

    private void circle_Click(object sender, EventArgs e)
    {
        chosenCircle.Checked = false;
```

```
        chosenCircle = (ToolStripMenuItem)sender;
        chosenCircle.Checked = true;

        if (circleTurn)
            infoPicture.BackgroundImage = chosenCircle.Image;

        for (int x = 0; x < 3; x++)
        {
            for (int y = 0; y < 3; y++)
            {
                if (grid[x, y] == cellState.circle)
                {
                    PictureBox img =
                        (PictureBox)tableLayoutPanel1.GetControlFromPosition(x, y);
                    img.BackgroundImage = chosenCircle.Image;
                }
            }
        }
    }
```



Rysunek 3: Przykłady opcjonalnych skórek



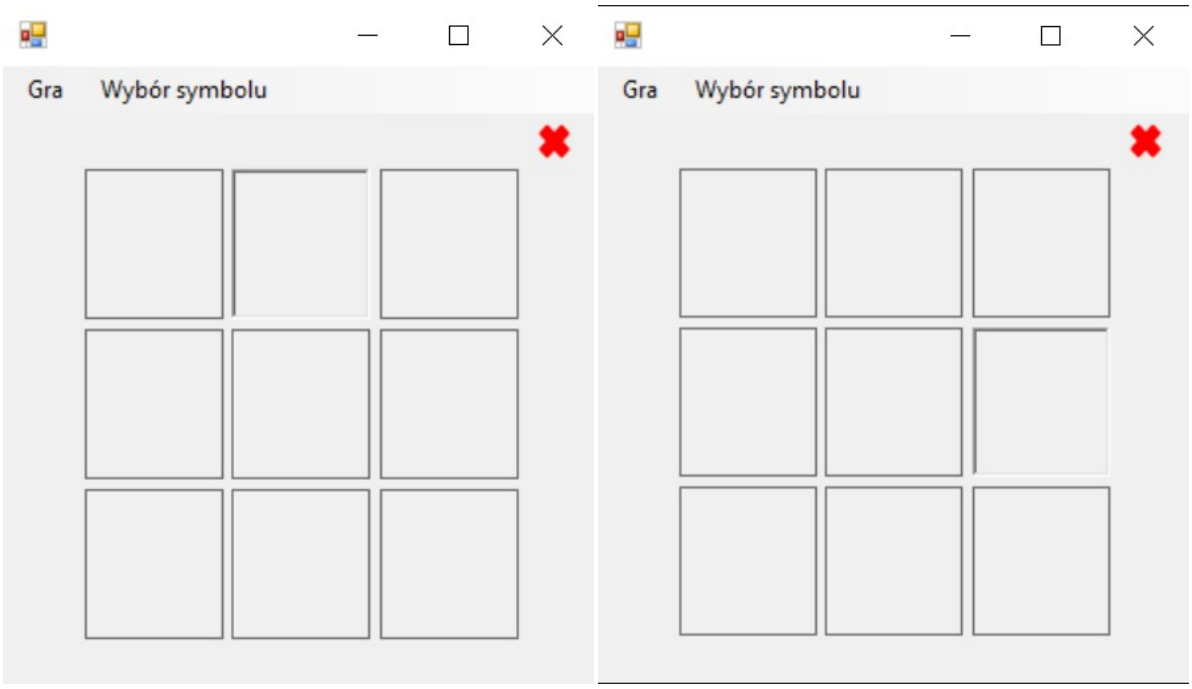
Rysunek 4: Menu wyboru skórek krzyżyków

2.4 Obsługa programu przy pomocy klawiatury

Obsługę klawiatury zrealizowano przy pomocy funkcji Form1KeyDown. W zależności od wciśniętej strzałki aktualizowany jest indeks aktywnego pola. Większość pól ma ustawioną własność Border jako FixedSingle, pole aktywne jako Fixed3D. Wciśnięcie klawisza Enter jest równoważne wciśnięciu aktywnego pola przyciskiem myszy.

```
    private void Form1_KeyDown(object sender, KeyEventArgs e)
        {
            PictureBox img = (PictureBox)tableLayoutPanel1.GetControlFromPosition(focus[0],
                focus[1]);
            img.BorderStyle = BorderStyle.FixedSingle;

            switch (e.KeyCode)
            {
                case Keys.Left:
                    focus[0] = Math.Abs((focus[0] + 3 - 1)) % 3;
                    break;

                case Keys.Right:
                    focus[0] = Math.Abs((focus[0] + 3 + 1)) % 3;
                    break;

                case Keys.Down:
                    focus[1] = Math.Abs((focus[1] + 3 + 1)) % 3;
                    break;

                case Keys.Up:
                    focus[1] = Math.Abs((focus[1] + 3 - 1)) % 3;
                    break;

                case Keys.Enter:
                    tile_Click(img, EventArgs.Empty);
                    break;
            }
```

```
            img = (PictureBox)tableLayoutPanel1.GetControlFromPosition(focus[0], focus[1]);
            img.BorderStyle = BorderStyle.Fixed3D;
        }
```



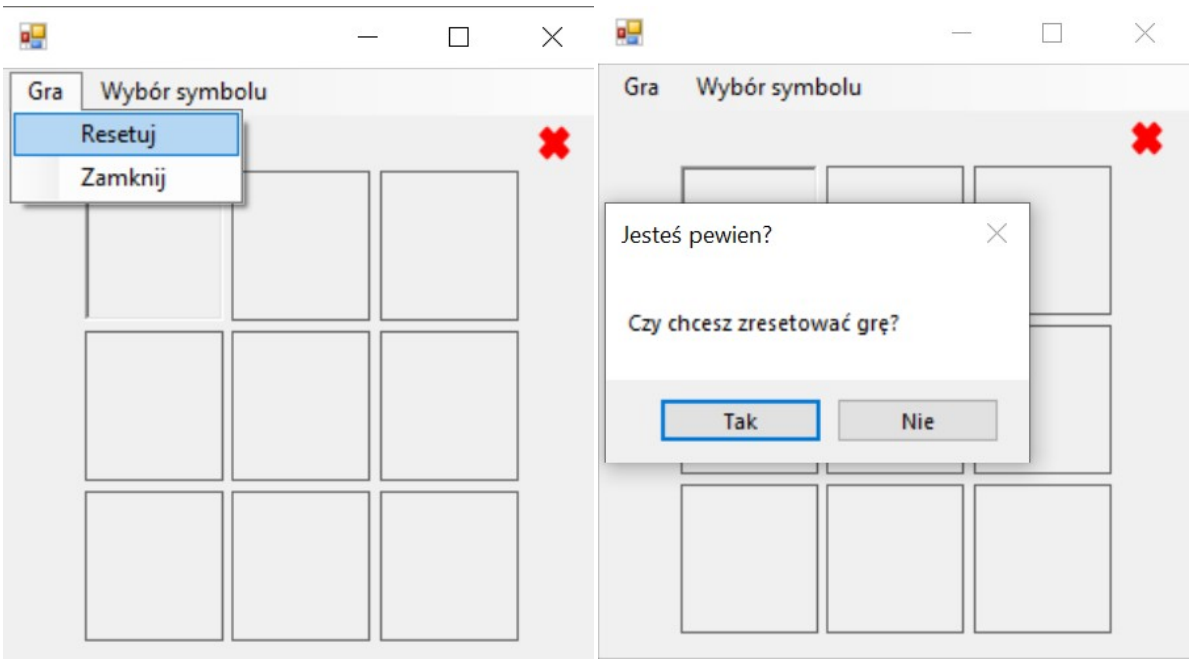
(a) Górne środkowe pole aktywne (b) Środkowe prawe pole aktywne

Rysunek 5: Przykłady wyboru pola klawiaturą

2.5 Rozwijane menu Gra

Rozwijane menu "Gra"pozwala na reset stanu gry lub zamknięcie programu.

```
private void zamknijToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DialogResult result = MessageBox.Show("Czy chcesz zamknąć grę?", "Jesteś pewien?",
                MessageBoxButtons.YesNo);
            if (result == DialogResult.Yes)
                Close();
        }

        private void Reset()
        {
            circleTurn = false;
            infoPicture.BackgroundImage = chosenCross.Image;
            for (int x = 0; x < 3; x++)
            {
                for (int y = 0; y < 3; y++)
                {
                    grid[x, y] = cellState.empty;
                    PictureBox img = (PictureBox)tableLayoutPanel1.GetControlFromPosition(x,
                        y);
                    img.BackgroundImage = null;
                }
            }
            turns = 0;
        }
```



(a) Rozwijane menu

(b) Potwiedzenie resetu gry

Rysunek 6: Działanie resetu stanu gry

3 Dyskusja osiągniętych wyników

Osiągnięty rezultat pozwala na grę w kółko i krzyżyk z innym graczem, przy jednym urządzeniu z użyciem myszki lub klawiatur, czyli główne założenie projektu. Zaletą rozwiązania jest łatwość w dodawaniu kolejnych obrazków reprezentujących kółka i krzyżyki. Wadą jest brak zmiany aktywnego pola przy interakcji myszką. Ten efekt występuje jedynie przy wybieraniu kratek klawiaturą. Program również nie zlicza wyniku obu graczy, nie pozwala na ich nie personalizacje, ani nie posiada licznika czasu. Te funkcje programu nie byłyby trudne do implementacji.