1. Pierwsza metoda programu wykorzystuje instrukcję warunkową, dzięki której tworzony jest każdorazowo delegat EventHandler, który identyfikuje metodę click_on_button dostarczająca odpowiedź na zdarzenie naciśnięcia na przycisk.

```
        private void Form1_Load(object sender, EventArgs e)
            {
            foreach (Control c in panel2.Controls)
                {
                if (c is Button)
                    {
                    c.Click += new System.EventHandler(Click_on_button);
                    }
                }
            }
```

2. Druga metoda wykorzystuje instrukcję warunkową if. Jeśli przycisk posiada pustą wartość tekstową jego znak zmieniany jest na X lub O, zmiana pomiędzy X lub O zachodzi na podstawie dzielenia modulo. Przy każdoczesnym wybraniu przycisku rośnie wartość zmiennej xo. W metodzie tej znajduje się również odwołanie do metody getTheWinner() odpowiedzialnej za wywołanie efektu po wygraniu tury przez jednego z graczy.

```
public void getTheWinner()
            {
            if (!button1.Text.Equals("") && button1.Text.Equals(button2.Text) &&
button1.Text.Equals(button3.Text)) // 1 2 3
                {
                WinEffect(button1, button2, button3);
                win = true;
                }
            if (!button4.Text.Equals("") && button4.Text.Equals(button5.Text) &&
button4.Text.Equals(button6.Text)) // 4 5 6
                {
                WinEffect(button4, button5, button6);
                win = true;
                }
            if (!button7.Text.Equals("") && button7.Text.Equals(button8.Text) &&
button7.Text.Equals(button9.Text)) // 7 8 9
                {
                WinEffect(button7, button8, button9);
                win = true;
                }
            if (!button1.Text.Equals("") && button1.Text.Equals(button4.Text) &&
button1.Text.Equals(button7.Text)) // 1 4 7
                {
                WinEffect(button1, button4, button7);
                win = true;
                }
            if (!button2.Text.Equals("") && button2.Text.Equals(button5.Text) &&
button2.Text.Equals(button8.Text)) // 2 5 8
                {
                WinEffect(button2, button5, button8);
                win = true;
                }
            if (!button3.Text.Equals("") && button3.Text.Equals(button6.Text) &&
button3.Text.Equals(button9.Text)) // 3 6 9
                {
                WinEffect(button3, button6, button9);
                win = true;
                }
            if (!button1.Text.Equals("") && button1.Text.Equals(button5.Text) &&
button1.Text.Equals(button9.Text)) // 1 5 9
```

```
                {
                WinEffect(button1, button5, button9);
                win = true;
                }
            if (!button3.Text.Equals("") && button3.Text.Equals(button5.Text) &&
button3.Text.Equals(button7.Text)) // 3 5 7
                {
                WinEffect(button3, button5, button7);
                win = true;
                }

            if (ButtonsLenght() == 9 && win == false)
                {
                label1.Text = "Nikt nie wygrał";
                }
            }
```

3. Trzecia metoda odpowiedzialna jest za wyświetlanie komunikatu w przypadku braku wygranej – wypełnieniu wszystkich pól przy braku ułożenia w jednej linii X lub 0.

```
public int ButtonsLenght()
            {
            int buttonsLenght = 0;
            foreach (Control c in panel2.Controls)
                {
                if (c is Button)
                    {
                    buttonsLenght += c.Text.Length;
                    }
                }
            return buttonsLenght;
            }
```

4. Czwarta metoda odpowiedzialna jest za efekt podświetlenia przycisków w przypadku ułożenia jednakowego znaku w jednej linii.

```
        public void WinEffect(Button b1, Button b2, Button b3)
            {
            b1.BackColor = Color.Blue;
            b2.BackColor = Color.Blue;
            b3.BackColor = Color.Blue;
            label1.Text = b1.Text + " Wygrał";
            }
```

5. Piąta metoda odpowiedzialna jest za nową rozgrywką, przyciski są w ten sposób zerowane.

```
        private void NewPartie_Click(object sender, EventArgs e)
            {
            xo = 0;
            win = false;
            label1.Text = "Graj";
            foreach (Control c in panel2.Controls)
                {
                if (c is Button)
                    {
                    c.Text = "";
                    c.BackColor = Color.White;
                    }
                }
            }
```

Diagram klas