**Условный оператор if else**

private void button1\_result\_Click(object sender, EventArgs e)

{

string value1 = textBox1\_value1.Text;

int x = Convert.ToInt32(value1);

string value2 = textBox2\_value2.Text;

int y = Convert.ToInt32(value2);

string sign = textBox3\_operation.Text;

char operation = Convert.ToChar(sign);

double result = 0;

if (operation == '+')

{

result = x + y;

}

else if (operation == '-')

{

result = x - y;

}

else if (operation == '\*')

{

result = x \* y;

}

else if (operation == '/')

{

if (y != 0)

{

result = x / y;

}

else

{

MessageBox.Show("Ошибка: на 0 делить нельзя");

}

}

label1\_result.Text = result.ToString();

**Условный оператор switch – case**

switch (operation)

{

case '+':

result = x + y;

break;

case '-':

result = x - y;

break;

case '\*':

result = x \* y;

break;

case '/':

if (y != 0)

{

result = x / y;

}

else

{

MessageBox.Show("Ошибка на 0 делить нельзя");

}

break;

default:

MessageBox.Show("Неизвестная ошибка");

return;

break;

**:?**

double result1 = (operation == '+' ? x + y :

operation == '-' ? x - y :

operation == '\*' ? x \* y: 0);

**Отображение результата во 2 форме**

Form2 form2 = new Form2();

form2.label1\_result2.Text = result.ToString();

form2.Show();

**методы класса control**

"Control" (или "элемент управления") — это общий термин, который описывает любой элемент пользовательского интерфейса, на который может взаимодействовать пользователь. Элементы управления могут включать в себя кнопки, текстовые поля, списки, переключатели

- \*\*Width\*\*: ширина элемента управления.

- \*\*Height\*\*: высота элемента управления.

**Bounds** — это свойство, которое определяет положение и размер элемента на форме, используя структуру типа Rectangle, которая содержит координаты (X, Y) и размеры (ширину и высоту) элемента.

**TabStop** — это логическое свойство, которое определяет, может ли элемент быть выбран с помощью клавиши Tab. Если оно установлено в true, элемент будет доступен для навигации через клавишу Tab.

**Bottom:**

* Свойство Bottom относится к элементам управления и представляет расстояние от верхней границы формы до нижней границы элемента управления.

Пример: Если у вас есть элемент управления, например, TextBox, и вы хотите разместить его внизу формы или с определенным отступом, вы можете использовать свойство Bottom.

**Font:**

* Свойство Font позволяет задать шрифт для элемента управления, например, для текста в кнопках, метках и текстовых полях.

- Top: Получает или устанавливает координату Y верхней границы элемента.

- Bottom: Получает или устанавливает координату Y нижней границы элемента.

- Left: Получает или устанавливает координату X левой границы элемента.

- Right: Получает или устанавливает координату X правой границы элемента.

Возможные значения BackgroundImageLayout

* ImageLayout.None: Изображение не будет отображаться.
* ImageLayout.Tile: Изображение будет повторяться по всей области.
* ImageLayout.Stretch: Изображение будет растянуто, чтобы заполнить всю область.
* ImageLayout.Center: Изображение будет размещено в центре области.
* ImageLayout.Zoom: Изображение будет масштабироваться, чтобы сохранить пропорции и заполнить область.

FlatStyle может принимать следующие значения:

1. Flat: Элемент управления отображается без границ и с плоским фоном.
2. Popup: Элемент управления отображается с эффектом всплытия, что делает его похожим на кнопку, которая может быть нажата.
3. Standard: Элемент управления отображается с обычными границами, как это обычно делается в Windows.
4. System: Элемент управления отображается в стиле, который соответствует текущей теме Windows.
5. private void Form1\_Load(object sender, EventArgs e)
6. {
7. this.Width = 550; //this - экземпляр класса определяет ширину элемента интерфейса
8. this.Height = 550;
9. this.Size = new Size(Width, Height);
10. this.BackgroundImage = Image.FromFile("C:/Users/mishu/source/repos/CR/tom\_jerry.jpg"); //Установка фона формы
11. this.BackgroundImageLayout = ImageLayout.Stretch; //Установка способа отображения (растянуть по всей форме)
12. textBox1\_value1.Bounds = new Rectangle(250, 30, 80, 30); // Устанавливаем позицию и размер первого текстового поля
13. textBox1\_value1.TabStop = true; //// Делает это текстовое поле доступным через Tab
14. textBox1\_value1.TabIndex = 0;
15. textBox2\_value2.TabIndex = 1;
16. textBox3\_operation.TabIndex = 2;
17. label1\_result.Font = new Font("Times New Roman", 14); // Устанавливаем шрифт
18. label1\_result.Location = new Point(255, button1\_result.Bottom + 20); //устанавливает позицию для элемента управления
19. //с использованием класса Point(x,y) x = 255 y = динамическое значение button1\_result.Bottom возвращает координату
20. //нижней границы кнопки
21. label1\_result.BackColor = Color.Green; //Устанавливаем задний фон на label - поле вывода результата
22. label1\_result.ForeColor = Color.Red; //Устанавливаем цвет текста label1
23. textBox2\_value2.Top = textBox1\_value1.Bottom + 15; //устанавливаем границу между нижней границей нижнего элемента и верхней границей верхнего элемента
24. textBox1\_value1.Left = textBox2\_value2.Left; //выравнивание текст бокса 1 и 2 по левому краю
25. label1\_result.TextAlign = ContentAlignment.MiddleCenter; //Установка и выравнивание текста внутри элемента управления
26. button1\_result.FlatStyle = FlatStyle.Flat; //стиль отображения элементов фон и границы
27. panel1.BorderStyle = BorderStyle.FixedSingle; //Устанавливает сплошную границу элемента

label1\_result.Visible = false; //видимость элемента

**Строковые методы**

string value1 = textBox1\_value1.Text.Trim(); //Trim() убирает пробелы слева и справа

1. **Insert(int startIndex, string value)**:
   * Вставляет строку value в текущую строку на позицию startIndex.
   * Пример: "Hello".Insert(5, " World") вернет "Hello World".
2. **Replace(string oldValue, string newValue)**:
   * Заменяет все вхождения подстроки oldValue на newValue.
   * Пример: "Hello World".Replace("World", "C#") вернет "Hello C#".
3. **Remove(int startIndex, int count)**:
   * Удаляет count символов из строки, начиная с позиции startIndex.
   * Пример: "Hello World".Remove(5, 6) вернет "Hello".
4. **Substring(int startIndex)**:
   * Возвращает подстроку, начиная с позиции startIndex до конца строки.
   * Пример: "Hello World".Substring(6) вернет "World".
5. **Split(char[] separator)**:
   * Разделяет строку на массив подстрок, используя указанные символы-разделители.
   * Пример: "Hello,World".Split(',') вернет массив ["Hello", "World"].
6. **PadLeft(int totalWidth, char paddingChar)**:
   * Дополняет строку слева до заданной ширины totalWidth, используя символ paddingChar.
   * Пример: "42".PadLeft(5, '0') вернет "00042".
7. **ToCharArray()**:
   * Преобразует строку в массив символов.
   * Пример: "Hello".ToCharArray() вернет массив ['H', 'e', 'l', 'l', 'o'].
8. **PadRight(int totalWidth, char paddingChar)**:
   * Дополняет строку справа до заданной ширины totalWidth, используя символ paddingChar.
   * Пример: "42".PadRight(5, '0') вернет "42000".

string answer = value2.Insert(2, value1); //строковый метод вставляет сроку в тек строку по индексу

Console.WriteLine(answer);

string values = value1.Substring(0, value1.Length / 2) + value2.Substring(value1.Length / 2); //возвращает подстроку с опр индекса и до конца

Console.WriteLine(values);

string answer = value2.Insert(2, value1); //строковый метод вставляет сроку в тек строку по индексу

Console.WriteLine(answer);

string values = value1.Substring(0, value1.Length / 2) + value2.Substring(value1.Length / 2); //возвращает подстроку с опр индекса и до конца

Console.WriteLine(values);

string replacedValue = value1.Replace('0','4');//заменяет все вхождения подстроки на новые

Console.WriteLine(replacedValue);

string removedValue = value1.Remove(0, 6);//удаляем 6символов, начиная с первого

Console.WriteLine(removedValue);

string[] splitValues = value2.Split('.'); //разделяем строку по точке

Console.WriteLine(splitValues);

char[] chars\_values = value2.ToCharArray(); //преобразуем строку в массив

Console.WriteLine(chars\_values);

label\_result.Text = $"Replaced: {replacedValue}\nRemoved: {removedValue}\nSplit:

{string.Join(", ", splitValues)}\nPadded: '{paddedValue}'";

label1\_result.Text = string.Join(",", splitValues);

int[] mas = new int[2];

**2 ответа 2 формы**

label1\_result.Text = result.ToString();

Form2 form2 = new Form2();

form2.label1\_result2.Text = result.ToString();

form2.Show();

/\*

double result1 = (operation == '+' ? x + y :

operation == '-' ? x - y :

operation == '\*' ? x \* y: 0); \*/

if (operation == "p1")

{

pad\_value1 = value1.PadLeft(value2.Length, '0');

label1\_result.Text = pad\_value1.ToString();

form2.label1\_result2.Text = pad\_value1.ToString();

form2.Show();

return;

}