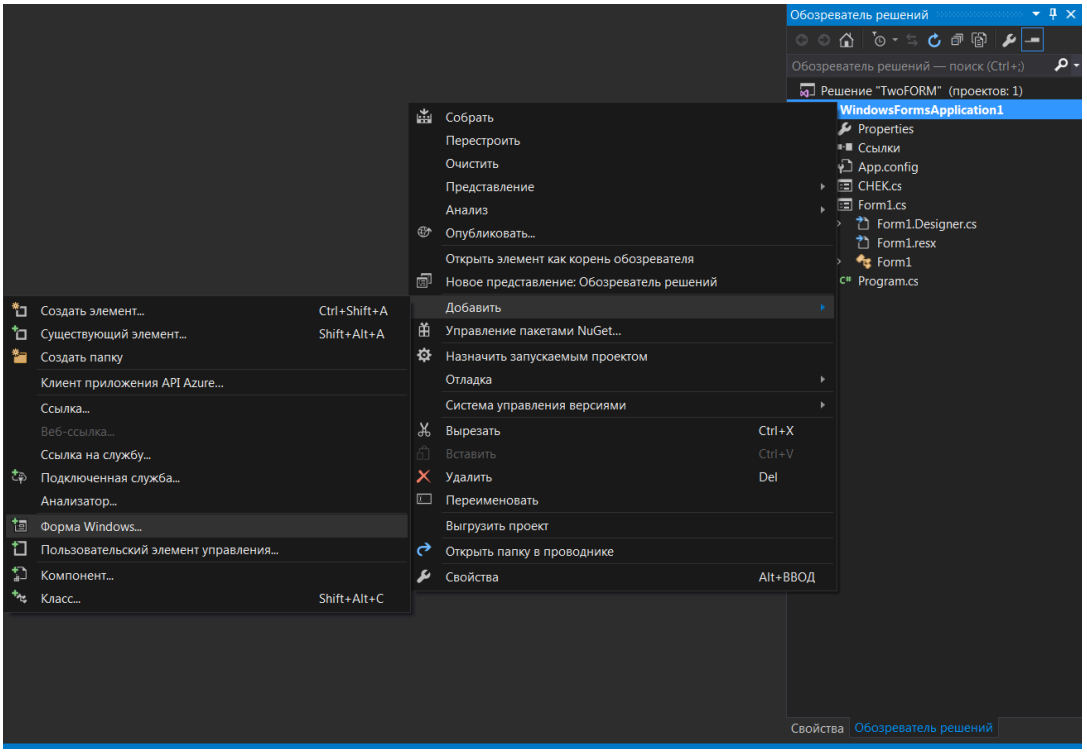
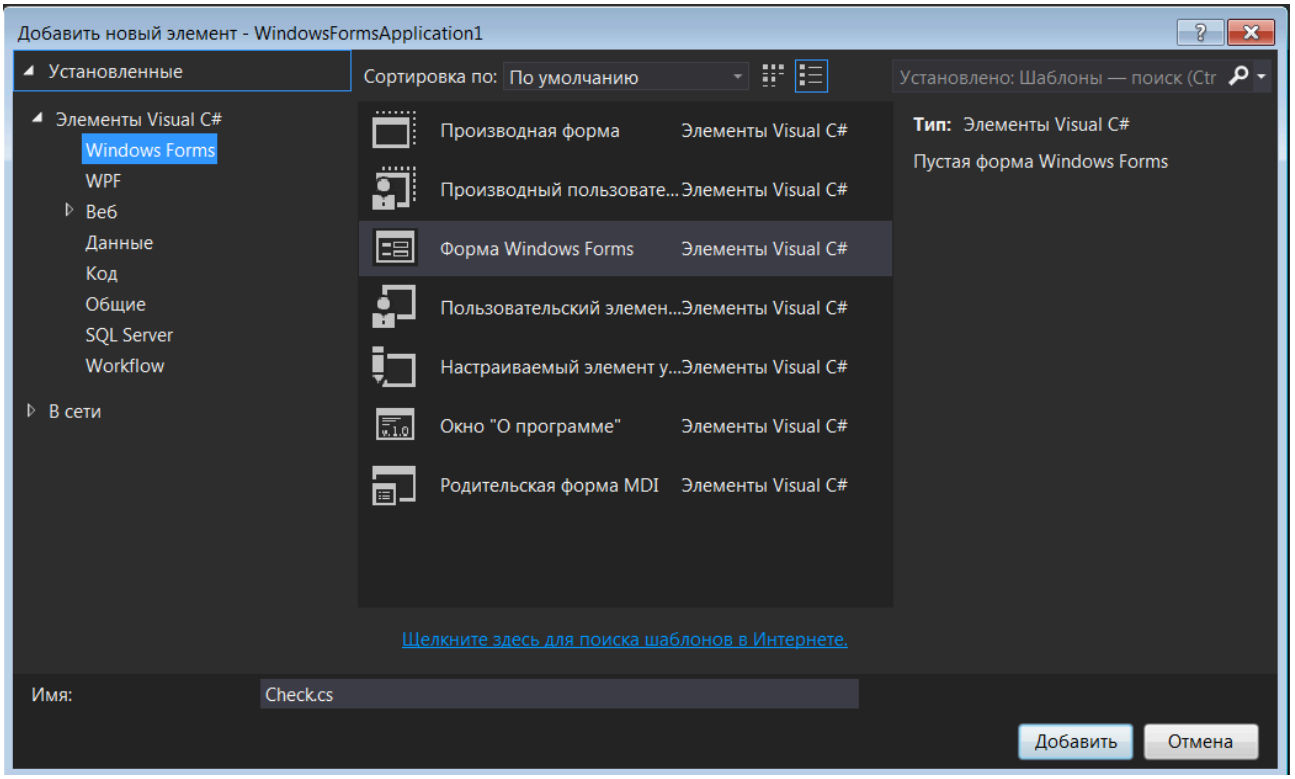
Додавання форми. Робота з кількома формами.

Щоб додати ще одну форму в проект у вікні: **Обозреватель решений => Добавить => Форма Windows**



Надамо новій формі назву **Check.cs** (створено новий клас Check)



Отже, у нас в проект була додана друга форма. Тепер спробуємо здійснити взаємодію між двома формами. Припустимо, перша форма після натискання на

кнопку буде викликати другу форму . По-перше, додамо на першу форму Form1 кнопку і подвійним клацанням по кнопці перейдемо в файл коду. Отже, ми потрапимо в обробник події натискання кнопки, який створюється за замовчуванням після подвійного клацання по кнопці:

```
private void button1_Click(object sender, EventArgs e)
{

}
```

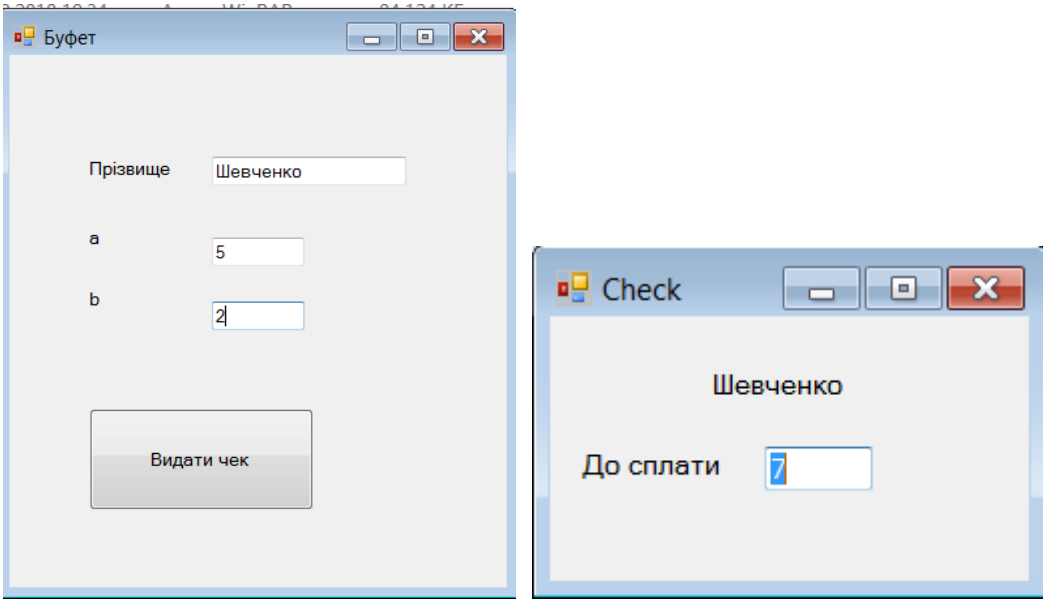
Тепер додамо в нього код виклику другої форми. У нас друга форма називається Check, тому спочатку ми створюємо об'єкт даного класу, а потім для його відображення на екрані викликаємо метод Show:

```
 Check checkForm;// оголошення поля checkForm типу Check
private void button1_Click(object sender, EventArgs e)
{

    checkForm = new Check ();
  // створили об'єкт класу Check,записали його в поле checkForm
    checkForm.Show();
   // визвали у об'єкта checkForm метод для відображення на екрані
    checkForm.label1.Text = textBox1.Text;
    int a = int.Parse(textBox2.Text);
    int b = int.Parse(textBox3.Text);
    checkForm.textBox1.Text = (a + b).ToString();
}
```

Щоб перша форма могла передати інформацію на другу форму, всі об'єкти форми **Check** (другої форми) повинні мати значення властивості **Modifiers** , що дорівнює **Public**.

В нашому прикладі таку властивість має об’єкт **label1(“Шевченко”)**, **textBox1(“7”)** з форми **Check**



Тепер зробимо навпаки - щоб друга форма впливала на першу. Поки друга форма не знає про існування першої. Щоб це виправити, треба другий формі якось передати відомості про першу формі. Для цього скористаємося передачею посилання на форму в конструкторі.

Отже перейдемо до другої формі і перейдемо до її коду - натиснемо правою кнопкою миші на форму і виберемо View Code (Перегляд коду). Поки він порожній і містить тільки конструктор. Оскільки C # підтримує перевантаження методів, то ми можемо створити кілька методів і конструкторів з різними параметрами і в залежності від ситуації викликати один з них. Отже, змінимо файл коду другий форми на наступний:

```
namespace WindowsFormsApplication1
{
    public partial class Check : Form
    {
        public Check()
        {
            InitializeComponent();

        }

        public Check(Form1 form)
        {
            InitializeComponent();
            form.BackColor = Color.Red;
        }
```

Фактично ми тільки додали тут новий конструктор `public Check(Form1 form)`, в якому ми отримуємо першу форму і встановлюємо її фон в червоний колір. Тепер перейдемо до коду першої форми, де ми викликали другу форму і змінимо його на наступний:

```
        Check checkForm;
        private void button1_Click(object sender, EventArgs e)
        {
            checkForm = new Check(this);
            checkForm.Show();
         }
```

Оскільки в даному випадку ключове слово this являє посилання на поточний об'єкт - об'єкт Form1, то при створенні другої форми вона буде отримувати її (посилання) і через неї керувати першою формою.

При роботі з декількома формами треба враховувати, що одна з них є головною - яка запускається першою в файлі Program.cs. Якщо у нас одночасно відкрито кілька форм, то при закритті головної закривається вся програма і разом з нею всі відкиті форми.