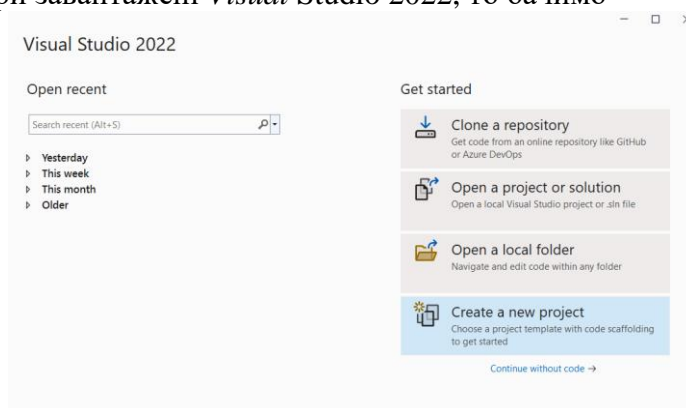


**Лабораторна робота № 7.****Тема:** Створення **Windows Form****Мета роботи:**

Ознайомлення з функціями класів: деструктори, індексатори, операції класу, операції перетворення типів.

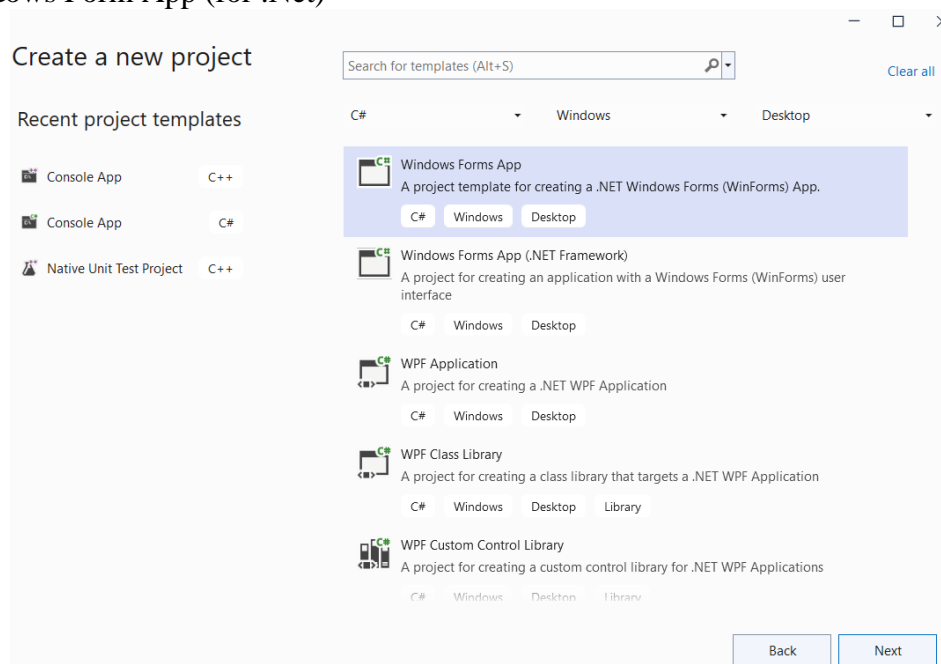
**Теоретичні відомості****Основи роботи з Visual Studio .NET**

Microsoft *Visual Studio .NET*- це інтегроване *середовище розробки* (Integrated *Development Environment (IDE)*) для створення, документування, завантаження та налагодження програм, написаних мовами *.NET*. Це потужний інструмент професійної розробки складних додатків, один із кращих. Набір його функцій надзвичайно великий. При завантаженні *Visual Studio 2022*, то бачимо



якщо натисни «Create a new project» тоді в наступному вікно вибираємо:

All language → C#, All platform → Windows, All project types → Desktop та Windows Form App (for .Net)



Вводимо ім'я та каталог збереження проекту

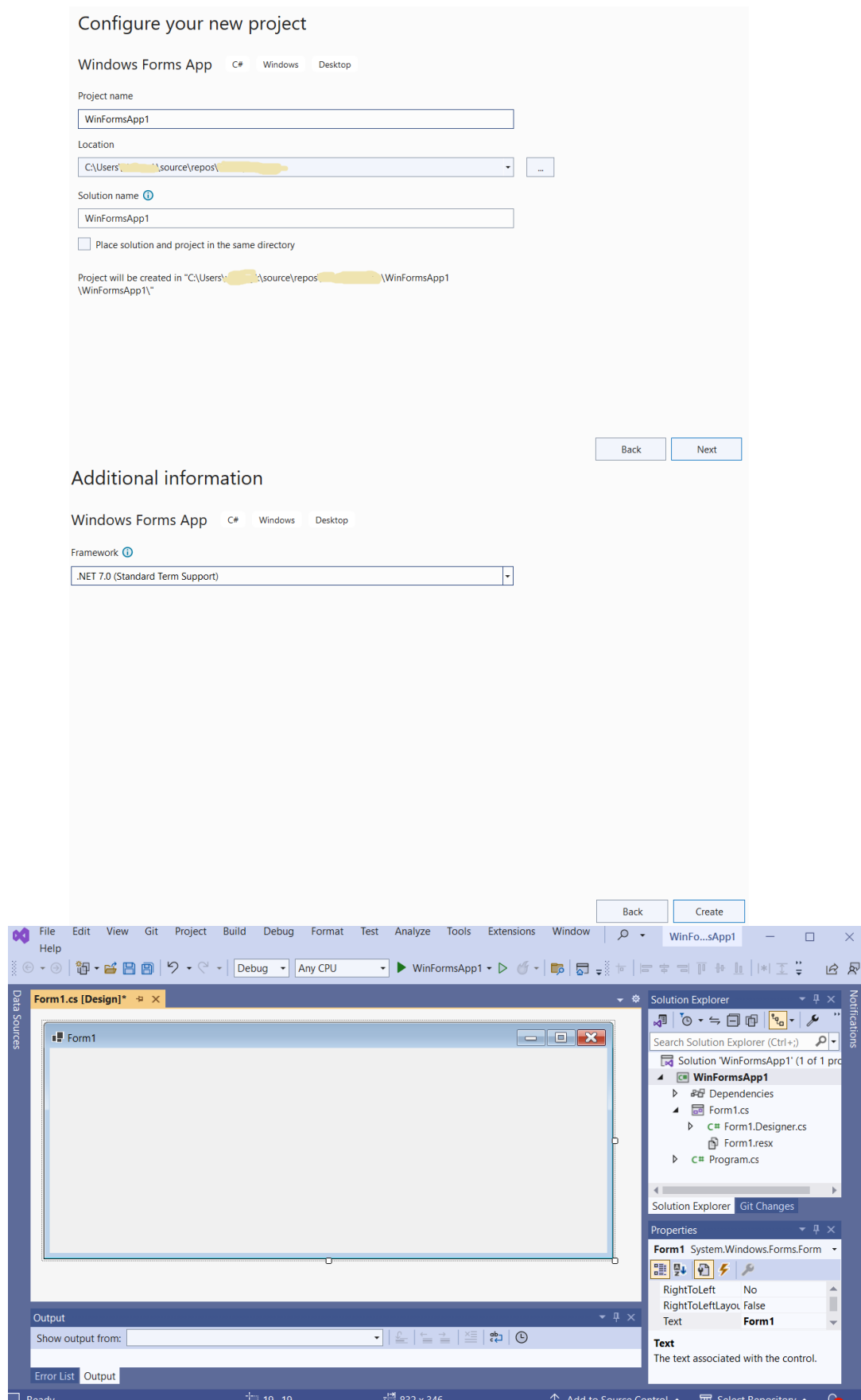


Рис.

Початкова сторінка (*Start Page*) містить наступні вкладки: *Projects*, *OnlineResources* та *My Profile*. За замовчуванням вибраний *список* недавніх проектів. На цій же вкладці розташовані кнопки *New Project*, *Open Project*, тощо. На вкладці *Online Resources* відображаються *групи новин*, *заголовки* та посилання ресурсів розроблювачів. Ця *опція* доступна, коли *комп'ютер* підключений до Інтернету.

Главное вікно *Visual Studio.NET*, подібно іншим додаткам *Windows*, містить рядок *меню*, що включає в себе наступні категорії (коли ми перебуваємо на *Start Page*, частина категорій не видна – вона з'явиться пізніше, коли буде створений проект)

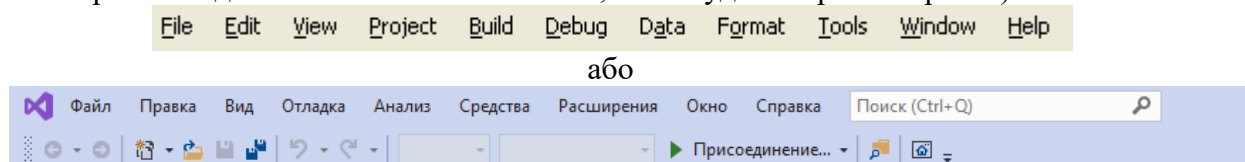



Рис. Рядок меню Visual Studio .NET

У цих категоріях розташовані наступні команди:

- File - відкриття, створення, додавання, закривання, друк, тощо.
- Edit - стандартні команди виправлення: копіювання, вставка, вирізання, тощо.
- View - команди для приховання й відображення всіх вікон і панелей інструментів.
- Project - команди для роботи із проектом: додавання елементів, форм, посилань, тощо.
- Build - команди компіляції програми.
- Debug - команди для налагодження програми.
- Data - команди для роботи з даними.
- Format - команди форматування розташовуваних елементів (вирівнювання, інтервал, тощо).
- Tools - команди додаткових інструментів і налаштування Visual Studio .NET.
- Window - керування розташуванням вікон.
- Help - довідка.

Якщо після внесених змін і при наступному запуску *Visual Studio .NET* виявляється незвичний вид програми, можна налаштувати його заново, запустивши *Start Page* з *меню Help/ShowStart Page*. Панелі, що ховаються, *розташовані* з боків вікна, закріпити їх на екрані, нажавши на значок , або зовсім забрати з екрана, а потім знову відобразити, використовуючи відповідний *пункт меню View* (або еквівалентне сполучення клавіш).

## Форми

Форма — це екранний *об'єкт*, що забезпечує функціональність програми. Створюємо проект, який буде *містити* власні форми. Запускаємо *Visual Studio .NET*, вибираємо *File/New/Project* — з'являється *діалогове вікно* (*Ctrl+Shift+N* приводить до того ж результату), у якому вибираємо *Visual C# Project* й *Windows Application*.  
Або

Рис. Створення нового проекту

У *поле Name* задаємо ім'я проекту — *FirstForm* і зберігаємо його в папку, обумовлену *полем Location*. Отриману папку можна згодом перемістити на інший *комп'ютер* і продовжити роботу — у ній будуть перебувати всі створювані нами файли цього проекту. На екрані з'явилася порожня *Windows-форма*.

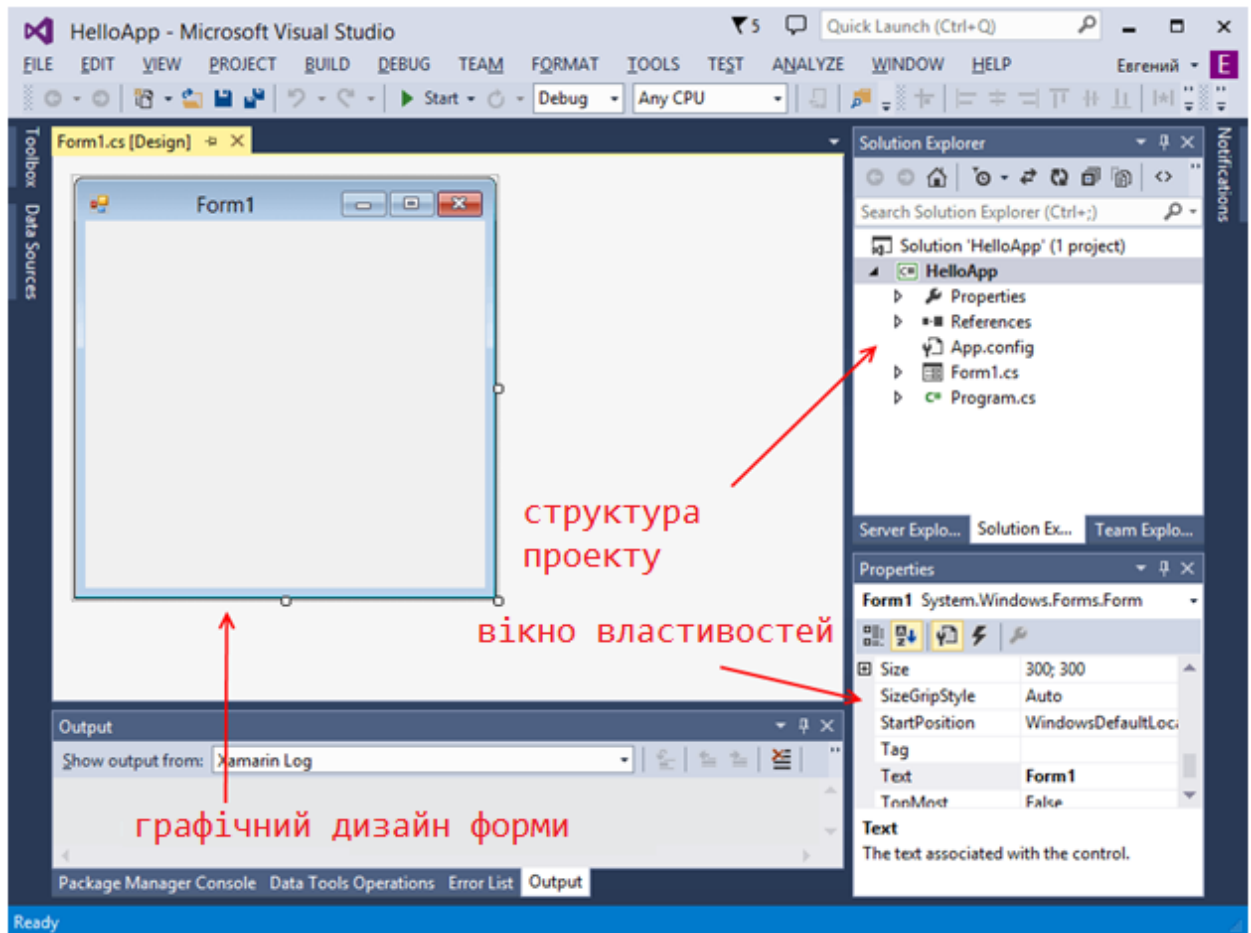


Рис. Головне вікно програми в режимі розробки **Solution Explorer**

Вікно *Solution Explorer* (провідник проекту, View → *Solution Explorer*, або сполучення клавіш Ctrl+Alt+L) містить компоненти, що входять до складу проекту.

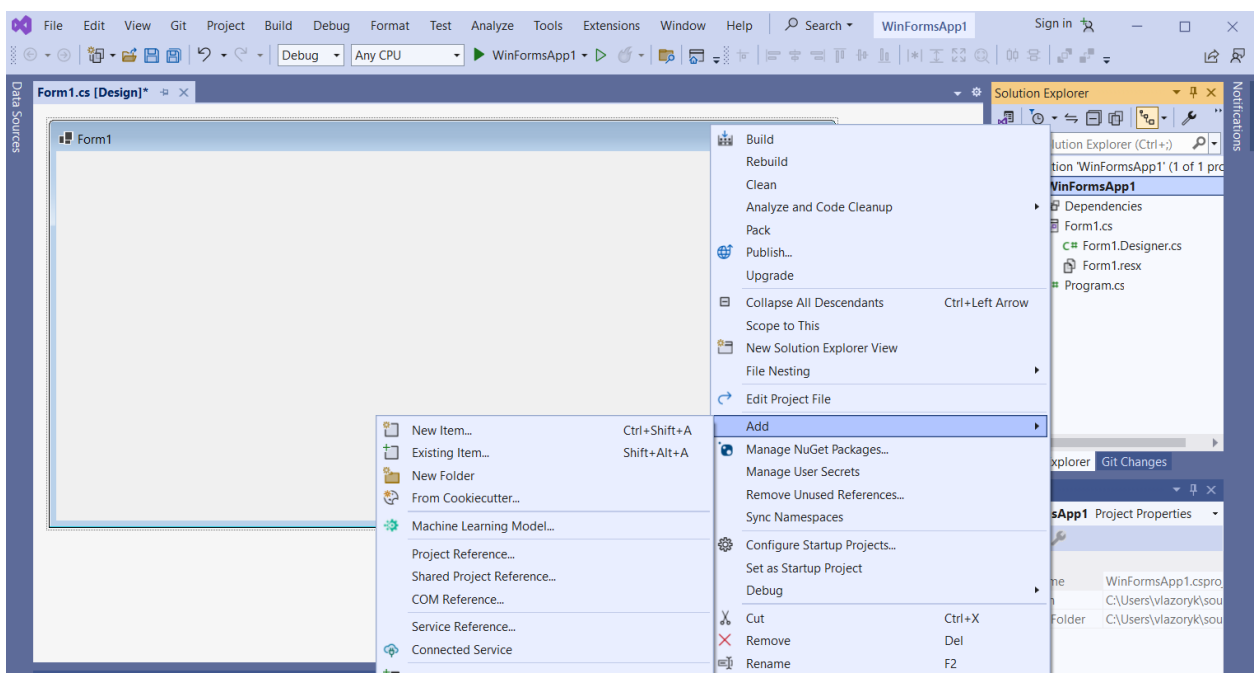
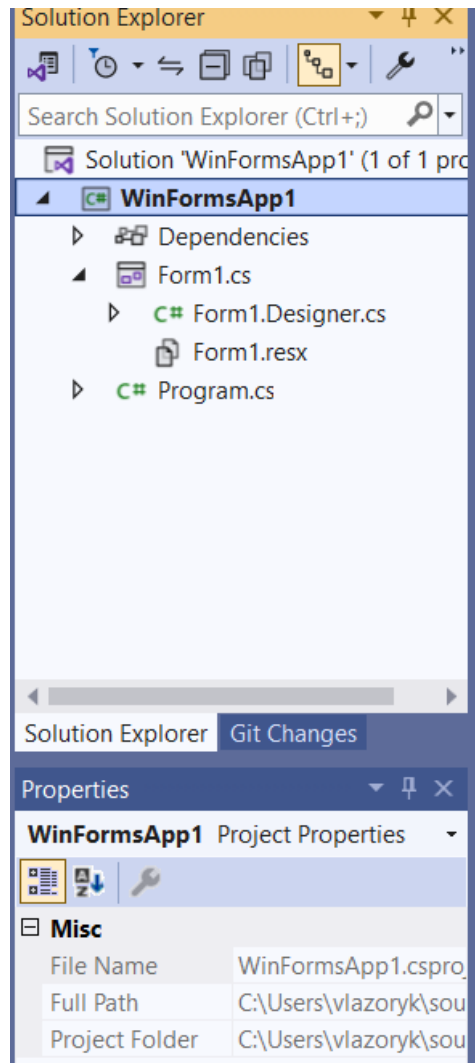


Рис. Контекстне меню вікна **Solution Explorer**

При створенні нового проекту *Solution Explorer* містить компоненти, створені шаблоном



**Рис.** Компоненти, що входять до складу нового додатка

Папка *References* містить посилання на класи, використовувані в проекті за замовчуванням. Подвійний клацанням миші на підпапках *References* завантажує вікно *Object Browser* (провідник об'єктів, View —> *Object Browser*, або сполучення клавіш Ctrl+Alt+J). Вікно *Object Browser*, у свою чергу, є вичерпним засобом одержання інформації про властивості об'єктів, як абстрактний клас *brush* успадковується від класу *System.MarshalByRefObject* і містить методи *Clone*, *Dispose(bool)*, *Dispose* й *Finalize*.

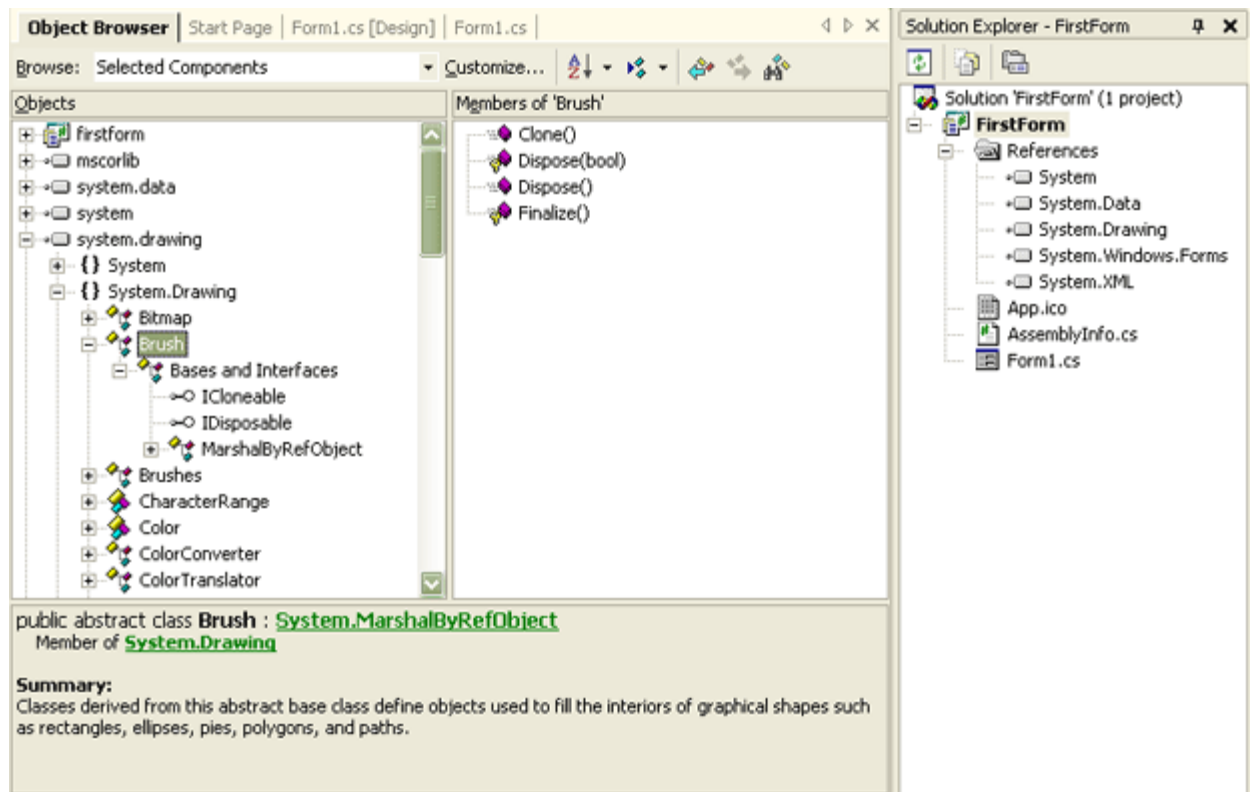


Рис. Вікно Object Browser

Можна одержувати короткий опис будь-якого методу, класу або властивості, просто клацнувши на ньому, — на інформаційній панелі негайно відобразиться коротка довідка. Для досвідченого розроблювача *Object Browser* — незамінний помічник у роботі, набагато більше зручний, чим довідка.

Файл *App.ico* містить зображення іконки, що на формі розташовано у верхньому лівому куті. Файл *AssemblyInfo.cs* містить інформацію про ваш додаток. При створенні дистрибутива (настановного пакета) у цьому файлі міститься інформація програми, використовувана в технічних цілях, а також цифровий ключ.

Вікно *Class View* — (огляд класів, View —> *ClassView*, або сполучення клавіш Ctrl+Shift+C), дозволяє переміщатися в коді по обраному об'єкті; містить методи, класи, дані всього листинга проекту.

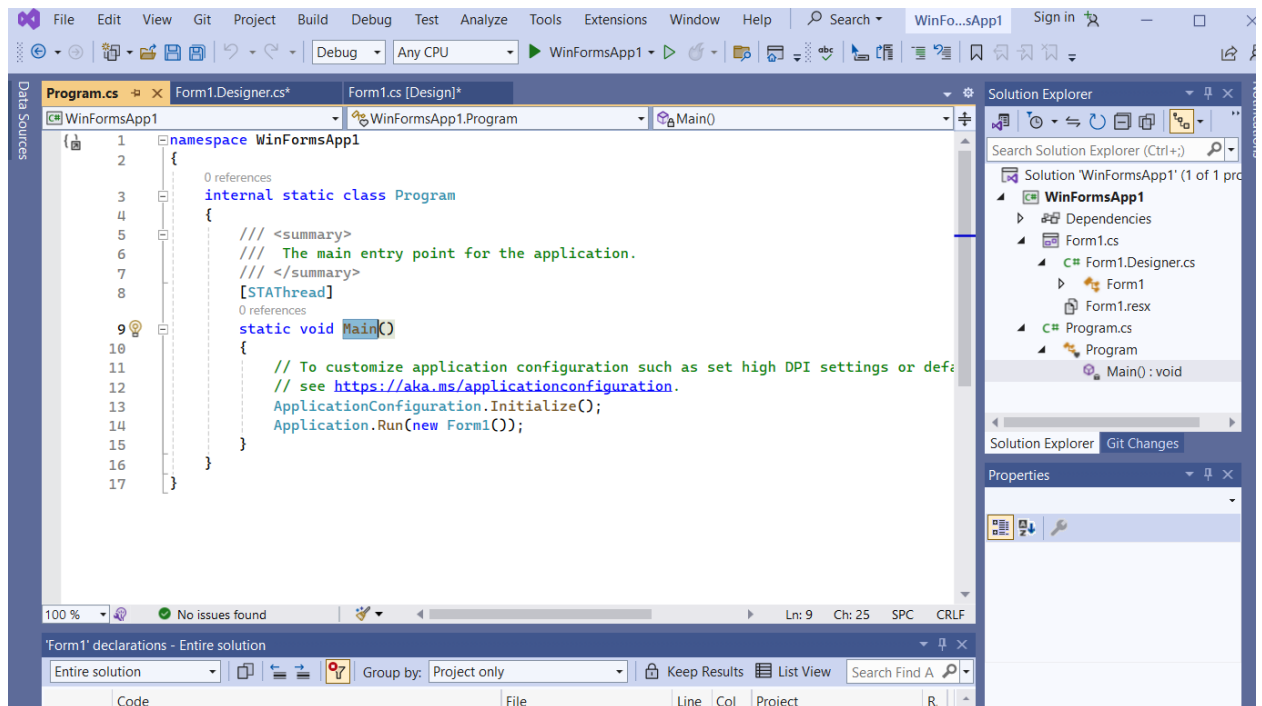


Рис. Вікно Class View

Вікно властивостей *Properties* — основний інструмент налаштування форми і її компонентів. Вміст цього вікна це список властивостей обраного в цей момент компонента або форми. Викликається це вікно декількома способами — у меню View вибираємо пункт Properties Window (або використаємо клавішу F4), на обраному об'єкті клацаємо правою кнопкою миші та у контекстному меню пункт Properties вибираємо об'єкт і натискаємо F4 або просто вибираємо об'єкт і переходимо у вікно Properties.

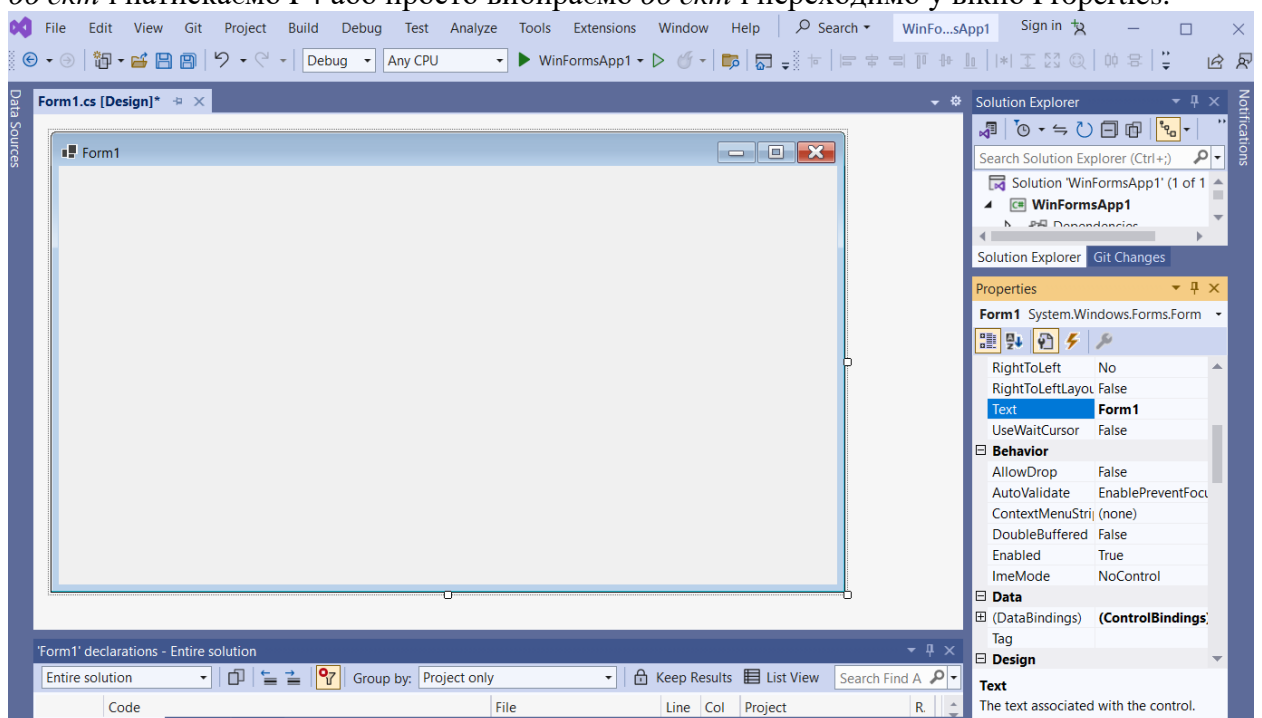





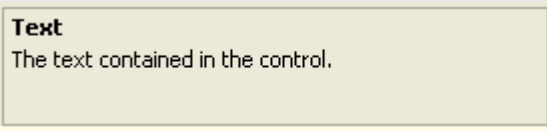


Рис. Вікно властивостей Properties

Опис інтерфейсу самого вікна Properties.


Таблиця 1.1.		
Елемент	Зображення	Опис
Object name		У поле цього списку виводиться назва даного обраного об'єкта, що є екземпляром якого-небудь класу. Тут Form1 — назва форми за замовчуванням, що успадковується від класуSystem.Windows.Forms.Form
Categorized		При натисканні на цю кнопку виробляється сортування властивостей обраного об'єкта по категоріях. Можна закривати категорію, зменшуючи число видимих елементів. Коли категорія схована, ви бачите знак (+), коли розкрита — (−)
Alphabetic		Сортування властивостей і подій об'єкта за абеткою
Properties		При натисканні на цю кнопку відображається перерахування властивостей об'єкта
Events		При натисканні на цю кнопку відображається перерахування подій об'єкта
Description Pane		Панель, на яку виводиться інформація про обрану властивість. У цьому випадку в списку властивостей форми було обране властивість Text


Вікно Properties дозволяє **визначати** в першу *чергу дизайн* форми і її елементів **керування**. В таблиці 1.2 **приводиться** опис деяких властивостей форми, звичайно **обумовлених** у режимі дизайну. При виборі значення властивості, **відмінного** від прийнятого *за замовчуванням*, воно виділяється жирним шрифтом, що полегшує надалі **визначення змін**.

Таблиця 1.2. Деякі властивості форми		
Властивість	Опис	Значення за замовчуванням



Name	Назва форми в проєкті. Це не заголовок форми, що ви бачите при запуску форми, а назва форми усередині проєкту, що ви будете використати в коді	Form1, Form2 і т.д.
AcceptButton	Установлюється значення кнопки, що буде спрацьовувати при натисканні клавіші Enter. Для того щоб ця властивість була активним, необхідна наявність принаймні однієї кнопки, розташованої на формі	None
BackColor	Кольори форми. Для швидкого перегляду різних варіантів просто клацайте прямо на назві "BackColor"	Control
BackgroundImage	Зображення на заднім тлі	None
CancelButton	Установлюється значення кнопки, що буде спрацьовувати при натисканні клавіші Esc. Для того щоб ця властивість була активним, необхідна наявність принаймні однієї кнопки, розташованої на формі	None
ControlBox	Установлюється наявність або відсутність трьох стандартних кнопок у верхньому правому куті форми: "Згорнути", "Розгорнути" й "Закрити"	
Cursor	Визначається вид курсору при його положенні на формі	Default
DrawGrid	Установлюється наявність або відсутність сітки із крапок, що допомагає формувати елементи керування. У кожному разі сітка видна тільки на стадії створення додатка	True
Font	Форматування шрифту, використовуваного для відображення тексту на формі в елементах керування	Microsoft Sans Serif; 8,25pt
FormBorderStyle	Визначення виду границь форми. Можливі варіанти:  None — форма без границь і рядка заголовка; FixedSingle — тонкі границі без можливості зміни розміру користувачем;	Sizable

	<p>Fixed3D — границі без можливості зміни розміру із тривимірним ефектом;</p> <p>FixedDialog — границі без можливості зміни, без іконки додатка;</p> <p>Sizable — звичайні границі: користувач може змінювати розмір границь;</p> <p>FixedToolWindow — фіксовані границі, є тільки кнопка закриття форми. Такий вид мають панелі інструментів у додатках;</p> <p>SizableToolWindow — границі з можливістю зміни розмірів, є тільки кнопка закриття форми</p>	
Icon	Зображення іконки, розташовуваної в заголовку форми. Підтримуються формати .ico	 (Icon)
MaximizeBox	Визначається активність стандартної кнопки "Розгорнути" у верхньому правому куті форми	True
MaximumSize	Максимальний розмір ширини й висоти форми, що задає в пікселях. Форма буде приймати зазначений розмір при натисканні на стандартну кнопку "Розгорнути"	0;0 (У весь екран)
MinimizeBox	Визначається активність стандартної кнопки "Згорнути" у верхньому правому куті форми	True
MinimumSize	Мінімальний розмір ширини й висоти форми, що задає в пікселях. Форма буде приймати зазначений розмір при зміні її границь користувачем (якщо властивість FormBorderStyle має значення за замовчуванням Sizable)	0;0
Size	Ширина й висота форми	300; 300
StartPosition	Визначення розташування форми при запуску додатка. Можливі наступні значення: <p>Manual — форма з'являється у верхньому лівому куті екрана;</p> <p>CenterScreen — у центрі екрана;</p> <p>WindowsDefaultLocation — розташування форми за замовчуванням. Якщо користувач змінив розміри форми, то при наступному її запуску вона буде мати той же самий вид і розташування;</p> <p>WindowsDefaultBounds — границі форми приймають фіксований розмір;</p> <p>CenterParent — у центрі батьківської форми</p>	WindowsDefaultLocation
Text	Заголовок форми. На відміну від властивості Name, саме ця назва форми, що не використовується в коді	Form1, Form 2 і т.д.
WindowState	Визначення положення форми при запуску. Можливі наступні значення: <p>Normal — форма запускається з розмірами, зазначеними у властивості Size;</p> <p>Minimized — форма запускається з мінімальними розмірами, зазначеними у властивості MinimumSize;</p> <p>Maximized — форма розвертається на весь екран</p>	Normal

Кнопка вікна властивостей Events (Події)  перемикає вікно Properties у режим керування оброблювачами різних подій (наприклад, миші, клавіатури) і одночасно виводить *список* всіх подій компонента. *Подвійний щиклик* миші в *поле* значення події генерує оброблювач для нього й перемикає в режим коду.

Вікно *Toolbox* (*панель елементів*, View —> *Toolbox*(Вид —> *Панель Елементов*), або сполучення клавіш Ctrl+Alt+X) містить компоненти *Windows-форм*, називані також елементами керування, які розміщуються на формі. Воно складається з декількох закладок: *My User Controls*, *Components*, *Data*, *Windows Forms* й *General* (рис. 1.11):

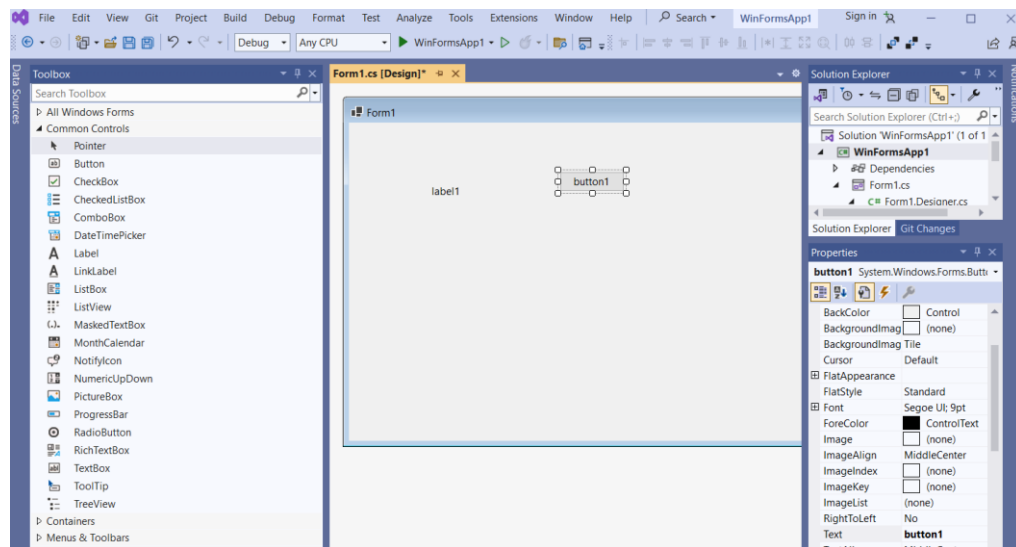
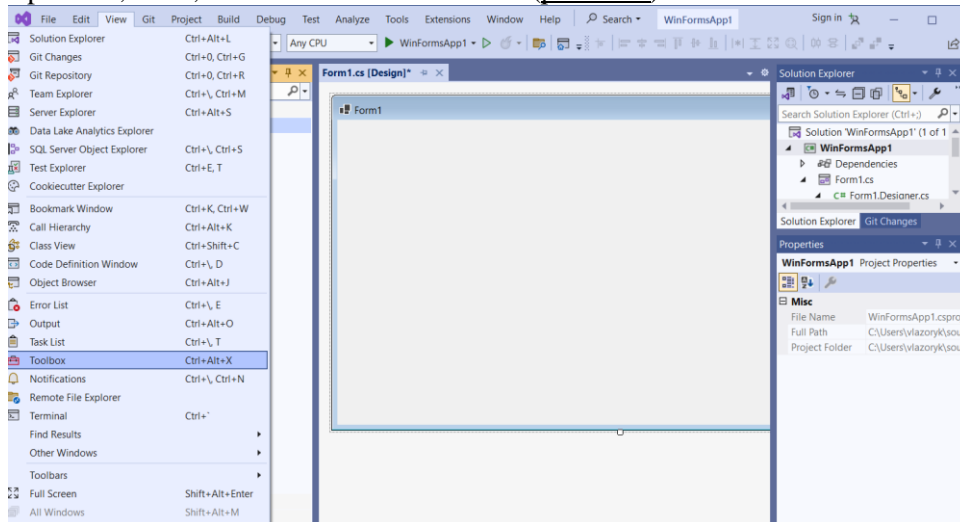
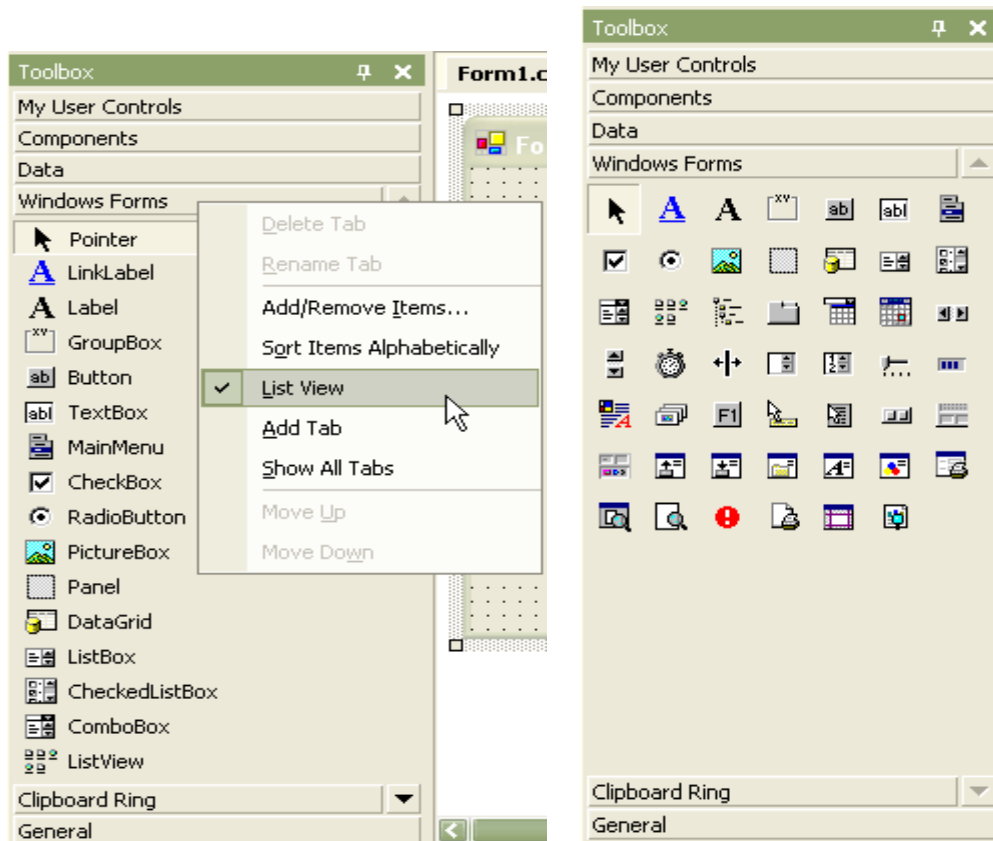
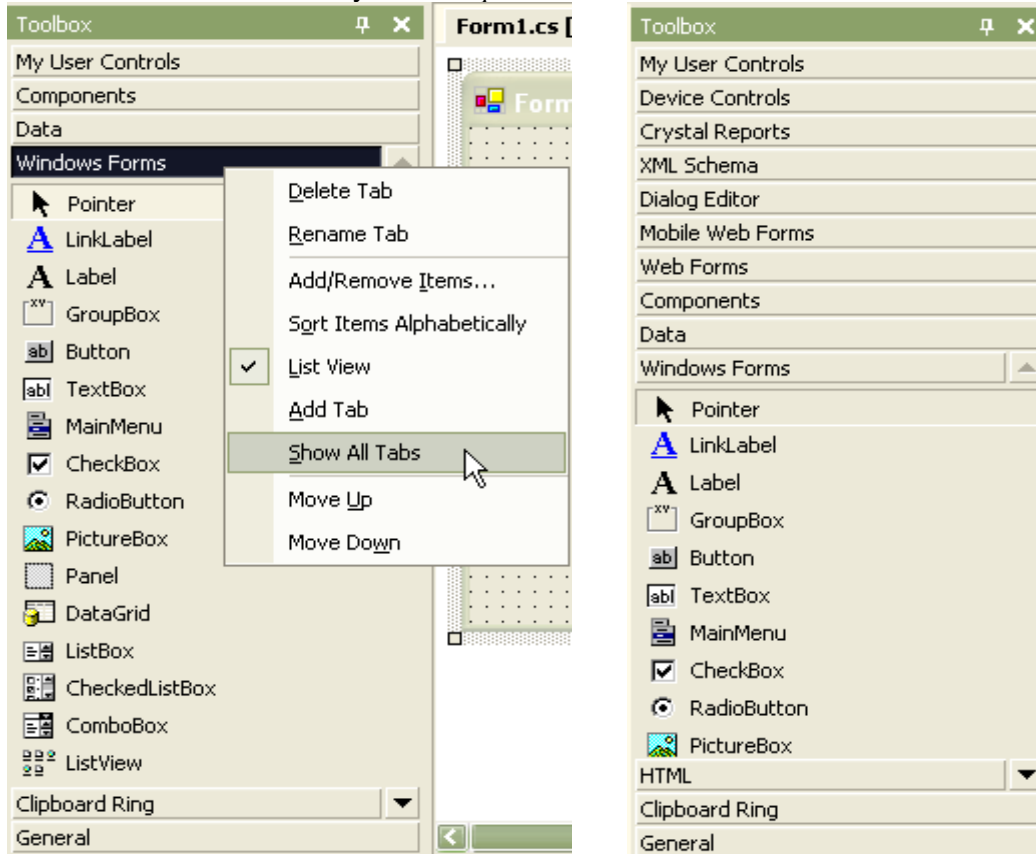


Рис. Вікно Toolbox

Найбільше часто вживаною закладкою є *Windows Forms*. Для розміщення потрібного елемента керування досить просто клацнути на ньому у вікні *Toolbox* або, схопивши, перетягнути його на форму. Перемикання виду значків дозволяє розмістити їх без смуги прокручування



**Рис.** Подання елементів у вигляді списку значків  
У вікні *Toolbox* доступне відображення всіх закладок.



**Рис.** Установлення галочки "Показати всі закладки" та повний список закладок

Закладка *My User Controls* дозволяє зберігати власні списки елементів керування — якщо ви найбільше часто використовуєте лише трохи з них, мабуть, має сенс перетягнути на цю закладку потрібні елементи. Або створити свою власну закладку

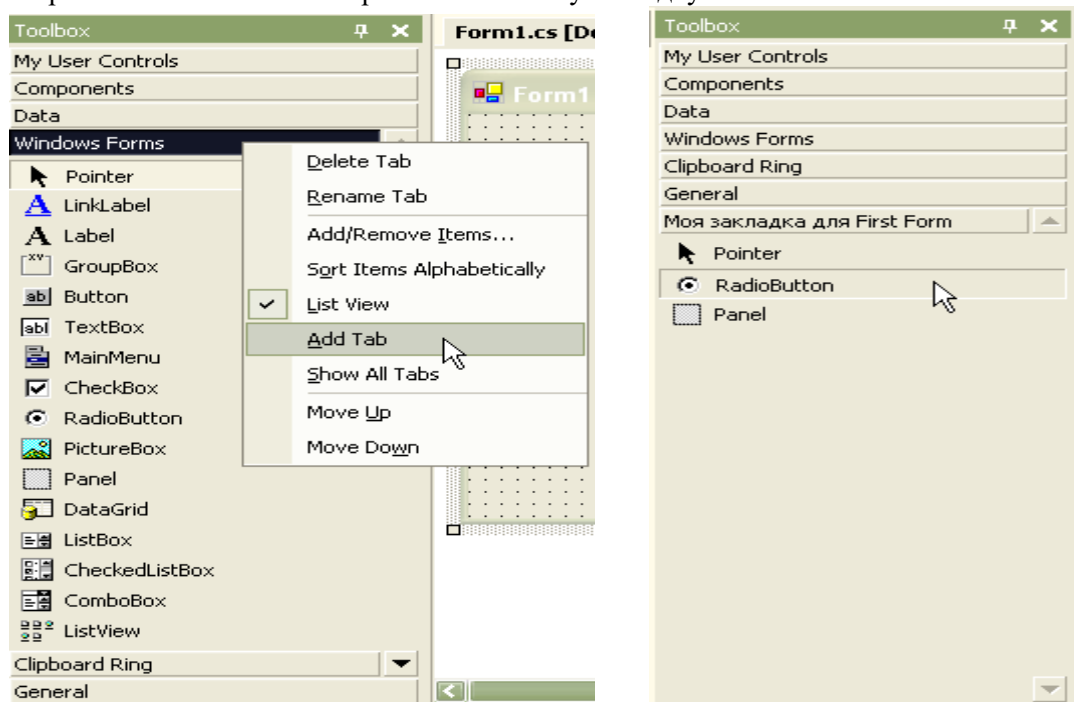


Рис. 1.16. Додавання закладки

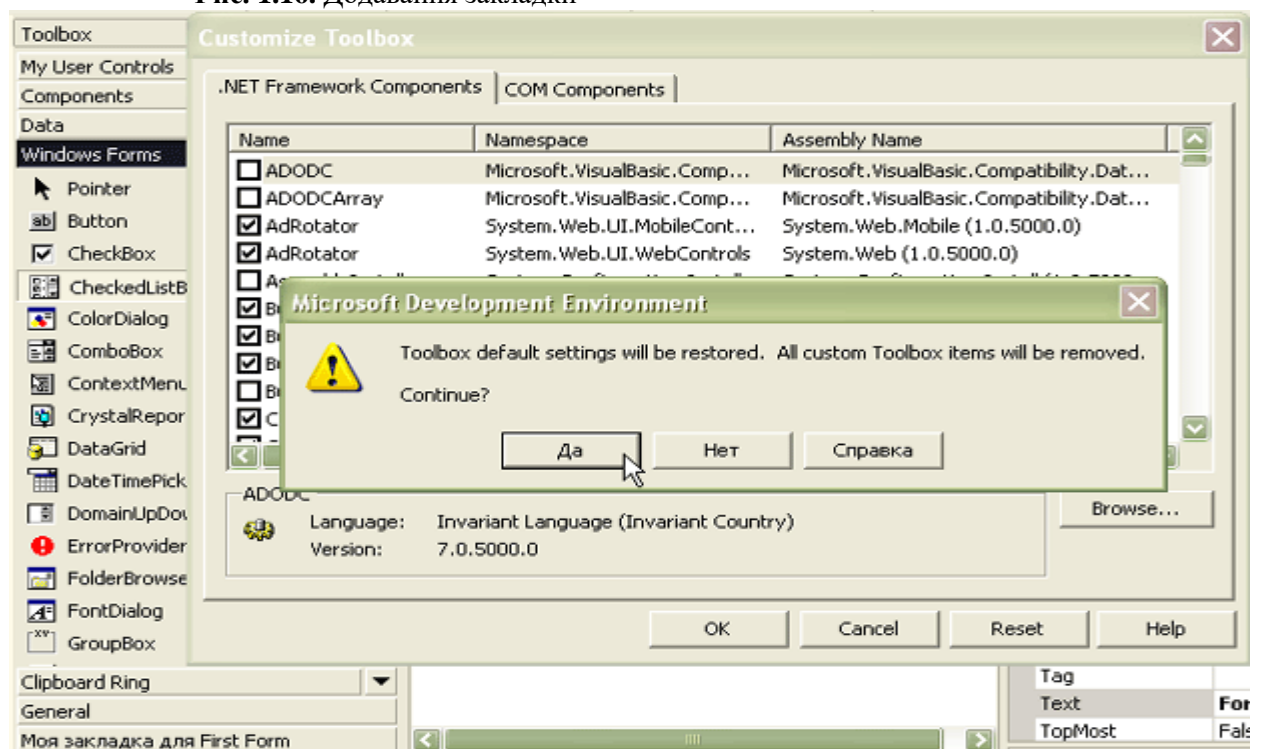


Рис. 1.18. Відновлення значень за замовчуванням

Створені в такий спосіб закладки можна перейменувати або видалити, вибравши в контекстному меню пункти *Rename Tab* й *Delete Tab* відповідно.

Якщо в результаті всіх експериментів ви виявите, що поточний вид вікна *Toolbox* сильно відрізняється від первісного, для відновлення значень за замовчуванням виберіть у контекстному меню будь-якої закладки пункт *Add/Remove Items.....* У вікні, що з'явилося, натисніть на кнопку *Reset*. З'являється вікно попередження — "Настроювання *Toolbox* будуть

відновлені. Всі користувацькі закладки будуть вилучені. Продовжувати?" Погодившись із попередженням, ви побачите вид *Toolbox* за замовчуванням.

*Продуктивність* розробки додатка багато в чому залежить від зручності налаштування користувацького середовища. Одним з ергономічних варіантів вважається загальне розташування вікон, що ховаються, що не заважає головну частину проекту (рис. 1.19).

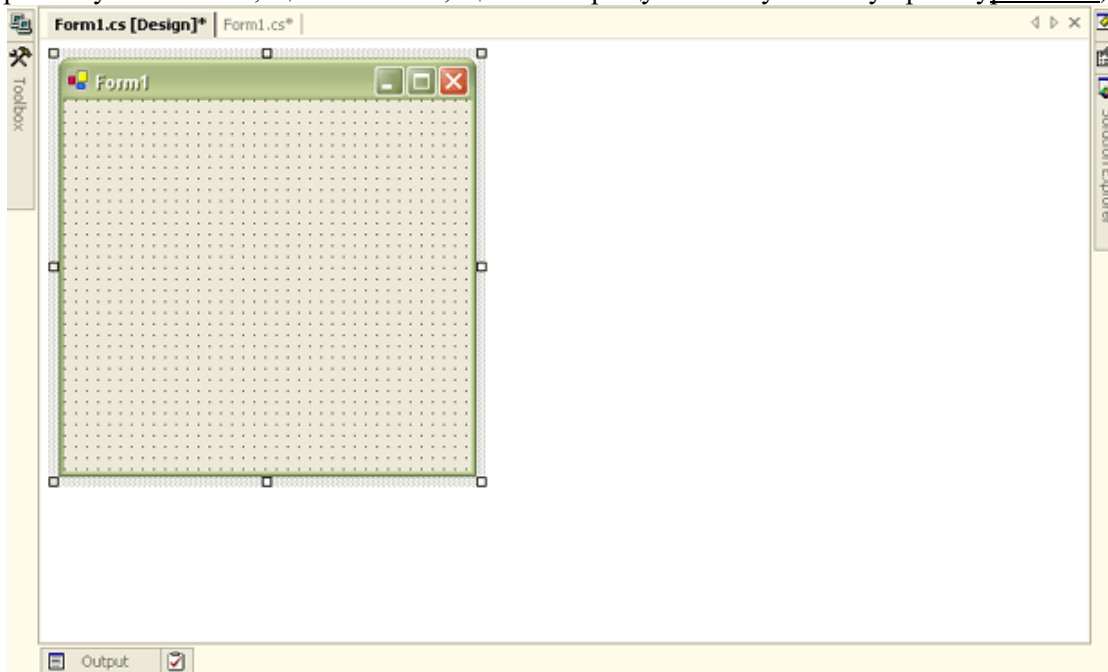


Рис. . Ергономічне розташування вікон, максимально звільняючи робочу область

### Режими дизайну й коду

При створенні нового проекту завантажується режим дизайну — форма являє собою основу для розташування елементів керування. Для роботи із програмою варто перейти в режим коду. Це можна зробити декількома способами: клацнути правою кнопкою миші по будь-якій частині форми й у меню, що з'явився, вибрати View Code, у вікні *Solution Explorer* зробити те ж саме на компоненті *Form1.cs* або просто двічі клацнути на формі — при цьому згенерується метод *Form1\_Load*. Після хоча б однократного переходу в режим коду в цьому проекті з'явиться вкладка *Form1.cs\** натискаючи на яку, теж можна переходити в режим коду. Для переходу в режим коду також можна використати клавішу F7, а для повернення в режим дизайну — сполучення Shift+F7.

Розглянемо деякі блоки режису коду. Перший блок визначає, які простори імен використовуються в цьому проекті:

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
```

Для перегляду інформації про вміст кожного із цих просторів можна скористатися вікном *Object Browser*. Далі **визначається** власний *простір* імен, ім'я якого збігається з назвою проекту:

```
namespace FirstForm
```

При необхідності ця назва можна міняти. *Клас* форми *Form1*, упакований від *System.Windows.Forms.Form*, містить у собі майже весь код:

```
public class Form1 : System.Windows.Forms.Form
{
...
}
```

Усередині цього класу перебуває *конструктор* форми:

```
public Form1()  
{  
    //  
    // Required for Windows Form Designer support  
    //  
    InitializeComponent();  
    //  
    // TODO: Add any constructor code after InitializeComponent  
call  
    //  
}
```

Подія Initiliazе відбувається в момент запуску додатка; код, що додає після InitializeComponent, може змінювати вміст форми або *елементи керування* в момент запуску.

Область Windows Form Designer generated code містить код графічного інтерфейсу елементів керування й форми, автоматично генерируемый середовищем. Порожня форма містить опис розмірів і заголовка. Клацніть на знак (+) для перегляду цієї області:

```
#region Windows Form Designer generated code  
    /// <summary>  
    /// Required method for Designer support - do not modify  
    /// the contents of this method with the code editor.  
    /// </summary>  
    private void InitializeComponent()  
{  
    this.components = new System.ComponentModel.Container();  
    this.Size = new System.Drawing.Size(300,300); // розмір форми  
в пикселях  
    this.Text = "Form1";// заголовок форми.  
}  
#endregion
```

Можна міняти значення параметрів, створювані середовищем, і тоді зміни негайно відображаються на графічному інтерфейсі.

Метод Main реалізує головну точку входу в програму — тобто *місце*, звідки починається виконання написаного нами коду:

```
static void Main()  
{  
    Application.Run(new Form1());  
}
```

При налагодженні більших програм зручно використати нумерацію рядків, яку можна включити в пункті *меню* Tools/Options.../TextEditor/C# – на формі Display — галочка Line Numbers.

### *Завдання до лабораторної роботи:*

#### **Порядок виконання роботи:**

- 1) Клонувати репозиторій C# **Windows Form**.  
<https://classroom.github.com/a/dczlp92m>
- 2) Розробити програму для вирішення задач.
- 3) Підготувати звіт в електронному виді надіслати  
мудл(<https://moodle.chnu.edu.ua/course/view.php?id=3371>).

**Завдання 1. Варіанти задач. Створити форму для виконання задачі.**

- 1.1. Реалізувати виконання арифметичних операцій(калькулятор). На формі розташувати два поля для введення чисел, та поле для виводу результату. Розташувати 4 радіо-кнопки, які визначають арифметичну операці. При натисненні на кнопку «Розрахувати» над числами поля1 та поля2 здійснюється вибрана арифметична дія, результат якої виводиться в поле3.
- 1.2. Реалізувати програму, що дозволє створювати примітивні малюнки. Реалізувати вибір мінімум чотирьох кольорів. При натисканні на ліву кнопку миші почати малювання. При натисканні кнопки «Нова картинка» стерти всі попередні малюнки та почати малювання нової картини.
- 1.3. Реалізувати програму, в якій можна додавати та видаляти елементи випадаючого списку. Додавати та видаляти елементи за допомогою TextBox, та двох кнопок – «Додати» та «Видалити».
- 1.4. Створити програму, що імітує роботу світлофора – через деякий проміжок часу, що може бути заданий користувачем, на світлофорі загоряються червоний, жовтий та зелений кольори.
- 1.5. Створити програму, що буде додавати речення із компонента TextBox в елемент RichTextBox з додаванням часу операції.
- 1.6. Створити програму, що вміщує компонент TextBox та кнопки «Інформація» та «Вихід». На натисканні на кнопку «Інформація» в текстове поле записується поточне положення форми на екрані. При підведенні миші до кнопки «Вихід» кнопка переміщується від курсору.
- 1.7. Створити програму, що вміщує компонент TextBox та кнопки «Дублювати» та «Вихід». При натисканні на кнопку «Дублювати» створюється нове вікно, що подібне до попереднього. Нове вікно активується, старе залишається відкритим. При натисканні на кнопку «Вихід» зачинається поточне вікно, якщо воно останнє – здійснюється вихід з програми.
- 1.8. Реалізувати програму, що буде виводити на екран секунди в десятинному (кнопка1), в бінарному представленні (кнопка2), або в шіснадцятковому представленні (кнопка2).
- 1.9. Реалізувати програму, яка вміщує на формі елемент RichTextBox, поле текстового введення та дві кнопки. По нажатисненні першої кнопки виводить букви з вікна введення у випадковому порядку в елемент RichTextBox. По натисненні другої кнопки замінити набір символів з вікна текстового введення на обрернений – тобто, якщо в вікні текстові введення були букви А, Б, В, то замінити їх на весь алфавит без букв А,Б,В.

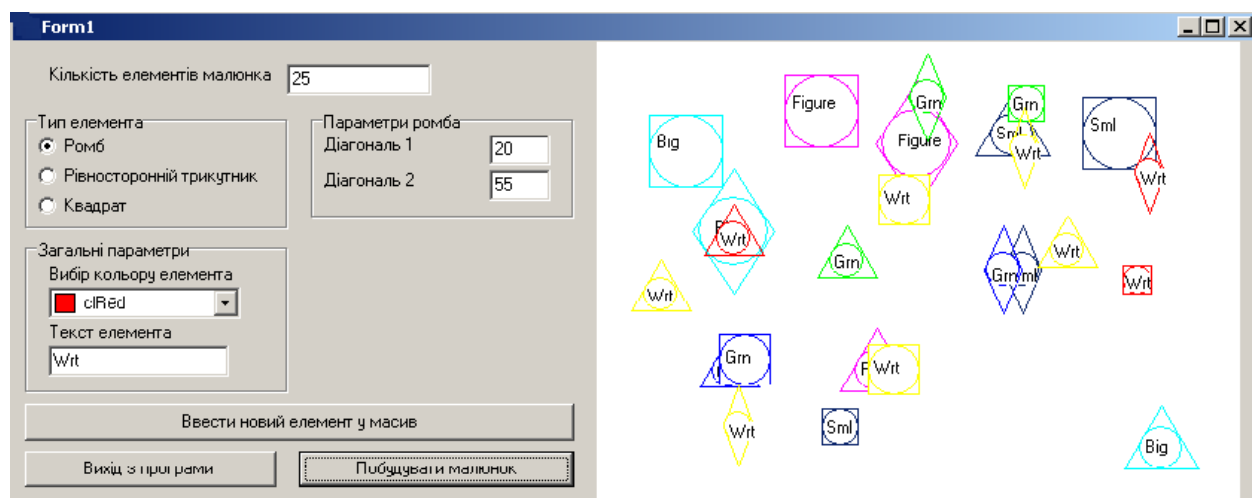


- 1.10. Побудови на формі анімаційної картини обертання одного компонента Panel навколо іншого Button. У процесі обертання повинні випадковим образом змінюватися властивості компонентів малювання Color і Width.
- 1.11. Побудови на формі компонента Panel з можливістю його масштабування й зміни властивостей Shape, Pen.Color, Pen.Style, Brush.Color, Brush.Style. Для установки коефіцієнта масштабування використати компонент Edit, для установки властивостей – компоненти ComboBox.
- 1.12. Заповнення компонента PictureBox примітивами випадкового розміру й кольору. Для вибору виду примітива (точка, коло, квадрат) використати компонент ComboBox, а для установки розміру примітива Edit.
- 1.13. Циклічного почергового виведення на компонент PictureBox через заданий проміжок часу, який обумовлюється компонентом Timer, послідовності метафайлів, визначених в компоненті Memo.
- 1.14. Побудови на формі анімаційної картини обертання одного компонента Panel навколо іншого. У процесі обертання повинні випадковим образом змінюватися властивості BackgroundImage і Color компонентів.
- 1.15. Малювання за допомогою миші на компоненті PictureBox графічних примітивів: відрізка, прямокутника, еліпса, кола. Додаток повинен містити компоненти RadioButton і RadioGroup для вибору типу примітива, що малюється.

Завдання 2. Варіанти задач. Створити форму для виконання задач.

- 2.1. Побудови на формі графіка функції  $y=(1-x^2)(x-2)$  ( $-2 < x < 2$ ). Додаток повинен включати компоненти RadioButton і RadioGroup для вибору кольорів, стилів і розмірів написів та побудови осей координат. Графік повинен виводитися в анімаційному режимі, забезпечуваному компонентом Timer.
- 2.2. Побудови на компоненті PictureBox графіка функції  $y=\sin(x)/x$  ( $0 < x < 4$ ). Додаток повинен включати компонент FontDialog для вибору кольорів, стилів і розмірів написів. Графік повинен виводитися в анімаційному режимі, обумовленому компонентом Timer.
- 2.3. Повної колірної інверсії і інверсії колірних складових зображення у форматі BMP. Програма повинна включати компоненти OpenFileDialog, SaveFileDialog для завантаження й збереження зображення й компоненти RadioButton і RadioGroup для вибору режиму інверсії.

- 2.4. Виділення колірних складових зображення у форматі BMP. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження та збереження зображення, компоненти RadioButton і RadioGroup для вибору виділюваної складової.
- 2.5. Пригнічення колірних складових зображення у форматі BMP. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження й збереження зображення й компоненти RadioButton і RadioGroup для вибору складової, що пригнічується.
- 2.6. Формування зображення, дзеркального до заданого зображення. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження та збереження зображення.
- 2.7. Повороту зображення квадратної форми на кут, кратний  $45^{\circ}$ . Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження й збереження зображення.
- 2.8. Перетворення кольорового зображення у зображення, яке представлено у шкалі сірого. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження та збереження зображення.
- 2.9. Перетворення кольорового зображення у монохромне. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження й збереження зображення.
- 2.10. Реалізацію інструмента піпетка. Додаток повинен включати компоненти OpenFileDialog, SaveFileDialog для завантаження й збереження зображення, компонент PictureBox для відображення зразка кольору і компоненти Label для виводу значень інтенсивностей RGB компонент кольору.



Завдання 3. Варіанти задач. Написати програму з використання **Windows Form** виведення малюнка, згідно варіанту. В програмі розробити форму

для ведення даних про об'єкти малюнка (тип об'єкта, розміри, колір тощо) та для виведення малюнка в об'єкт Image. Малюнок будується як набір елементів(типи елементів задаються згідно варіанту), координати розташування елемента в об'єкт PictureBox задавати за допомогою датчика випадкових чисел в межах полотна об'єкт PictureBox. Кожний елемент є об'єктом одного з похідних класів. Набір – задається, як масив на абстрактний базовий клас. В базовому класі передбачити віртуальні функцію малювання, функцію переміщення та інші функції. Базовий клас фігура похідні класи :

- 3.1. Квадрат, прямокутник, еліпс та ромб.
- 3.2. Дуга, сектор, еліпс та прямокутник із заокругленими кутами.
- 3.3. Квадрат, прямокутник, трикутник та коло.
- 3.4. Коло, квадрат, правильний трикутник та зірку.
- 3.5. Круг, сектор, зафарбований прямокутник та зірку.
- 3.6. Коло, квадрат та прямокутник з текстом в середині.
- 3.7. Квадрат, правильний трикутник та коло з текстом в середині.
- 3.8. П'ятикутник, прямокутник, трикутник та ромб.
- 3.9. Зафарбований еліпс, трикутник, прямокутник та зірку.
- 3.10. Шестикутник, ромб, трикутник та дугу.
- 3.11. Прямокутник із заокругленими кутами, еліпс, дуга та квадрат.
- 3.12. Прямокутник із заокругленими кутами, ромб, коло та дугу.
- 3.13. П'ятикутник, ромб та правильний трикутник з текстом в середині.
- 3.14. Правильні п'яти-, шести- та восьмикутники.
- 3.15. Шестикутник, ромб та коло з текстом в середині.
- 3.16. Квадрат, трапеція та ромб вписані в коло.
- 3.17. Прямокутник, правильний п'ятикутник та трикутник вписані в коло.

**Контрольні питання:**

1. Як створити нове Windows додаток в середовищі .NET?
2. Як додати новий компонент на форму?
3. Які основні властивості і події має форма?
4. Які основні властивості і події мають елементи управління?
5. Як можна визначити поточний стан курсору мишки на формі?