Лабораторна робота № 13

Тема. Реалізація наслідування класів-форм. Програмування віртуальних методів класів.

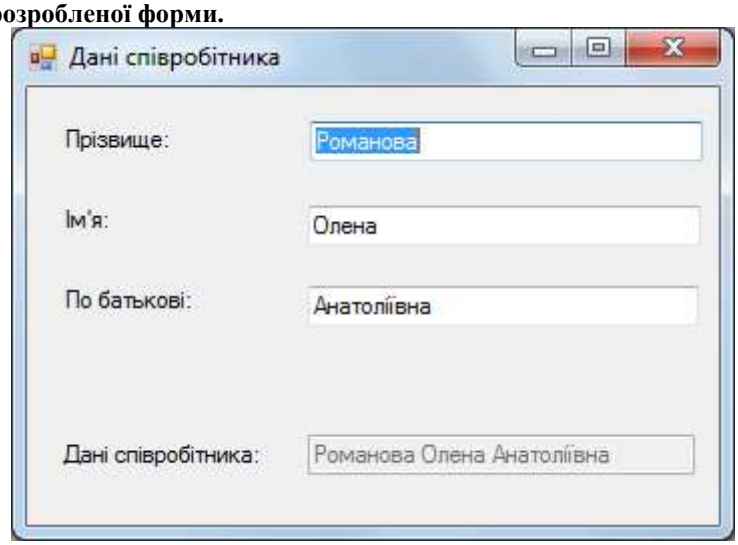
Мета. Формування вмінь і навиків програмування віртуальних методів класів. Закріплення вмінь і навиків програмування наслідування об’єктів, використання модулів, класів, об'єктів, файлів, підпрограм, функцій вводу-виводу та обробки рядків. Застосування вмінь і навиків програмування алгоритмів лінійної, розгалуженої та циклічної структури.

Контрольні запитання.

1. Як реалізувати наслідування класів?
2. Як змінити клас і назву для об'єкта-форми? Як змінити батьківський клас для класу?
3. Як перейти до тексту програми з візуальної форми? У файлах з якими розширеннями зберігаються описи класів та параметри зовнішнього вигляду форм?
4. Як описати, створити та використати візуальні компоненти батьківського класу?
5. Як звернутися до аналогічного методу батьківського класу?
6. Як забезпечити виклик з батьківського класу методу нащадка?

Завдання.

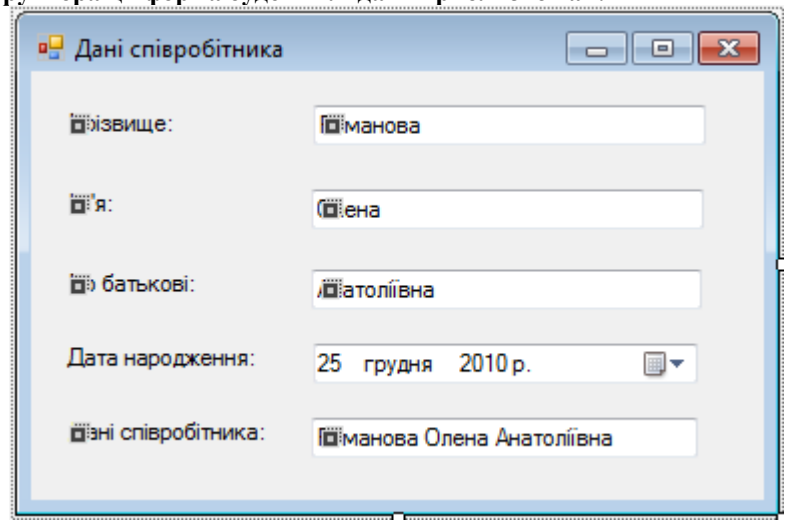
1. **Створіть новий проект, а в ньому – форму за зразком, наведеним нижче. Перейменуйте клас та файл форми на *FormBase*. Перейменуйте перших три поля відповідно на *PrizvFO, NameFO та FNameFO, а четверте – на Rez.* В перших три поля введіть ваші дані по замовчуванню. Забороніть доступ до четвертого поля в процесі експлуатації. Опишіть у класі форми публічну процедуру *formRez()* для автоматичного формування значення четвертого поля з перших трьох. Забезпечте її виклик як при редагуванні кожного з трьох перших полів (подія *TextChanged*), так і при завантаженні форми (подія *Load*). Забезпечте загальнодоступність поля результату *Rez* з класу форми та породжених класів. Для цього або в режимі конструктора виділіть це поле та встановіть значення його властивості *Modifiers* рівним *public* чи *protected* або перейдіть у файл опису візуальних компонентів і забезпечте в ньому загальнодоступність поля результату (замінивши модифікатор доступу *private* на *public* чи *protected*). Переконайтеся в дієздатності розробленої форми.**



При цьому текст модуля форми може виглядати, наприклад, так:

```
public partial class FormBase : Form
{
    public FormBase()
     {InitializeComponent();
     }

    virtual protected void formRez()
     {string s;
      s=PrizvFO.Text;
      if (NameFO.Text!="")
       {if (s!="")
          s+=" ";
        s+=NameFO.Text;
       }
      if (FNameFO.Text!="")
       {if (s!="")
          s+=' ';
        s+=FNameFO.Text;
       }
      Rez.Text = s;
     }

    private void Prizv_TextChanged(object sender, EventArgs e)
     {formRez();
     }

    private void Name_TextChanged(object sender, EventArgs e)
     {formRez();
     }

    private void FName_TextChanged(object sender, EventArgs e)
     {formRez();
     }

    private void FormBase_Load(object sender, EventArgs e)
    {formRez();
    }
}
```

2. **Додайте до вашого проекту ще одну форму Windows, в якій крім прізвища, імені та по батькові буде вводитися ще й дата народження, а додатково обчислюватися ще й кількість років. В режимі конструктора ця форма буде виглядати приблизно так:**



Для цього:

2.1. В оглядачі рішень виділіть файл проекту та оберіть в його контекстному меню пункт *Добавить – Форма Windows*, після чого у вікні створення нового елемента вкажіть *Форма Windows Form*;

2.2. Перейменуйте клас та файл цієї форми на *FormPorodgeno*. Забезпечте відображення цієї форми при завантаженні додатку замість завантаженням об'єкта форми *FormBase*. Для цього у файлі *Program.cs* замініть команду `Application.Run(new FormBase());` на `Application.Run(new FormPorodgeno());`

2.3. **З метою зменшення кількості повторюваних описів забезпечте породження класу цієї форми від *FormBase*.** Після цього в режимі конструктора форми мають автоматично з'явитися всі елементи керування базової форми з символом закріплення. Чому поле результату переміщувати і редагувати можливо, а вхідні поля – ні;

2.4. Для введення дати народження додайте у форму *FormPorodgeno* елемент керування класу *DateTimePicker*. Дайте йому назву *ДатаНародження*, та вкажіть вашу дату народження як значення по замовчуванню у властивості *Value*;

2.5. Перевизначте у класі *FormPorodgeno* публічну процедуру *formRez()* для додатково підрахунку кількості прожитих років та викличте її і при зміні поля *ДатаНародження* (подія *ValueChanged*), наприклад, так:

```
public partial class FormPorodgeno : FormBase
{
    public FormPorodgeno() : base()
    {
        InitializeComponent();
    }

    protected void formRez()
    {
        base.formRez();
        string s = Rez.Text;
        if (s != "")
         s+=" ";
        s+= " ("+((int)((double)(DateTime.Now-ДатаНародження.Value).Days/365.25)).ToString()
            +" років"+ ')';
        Rez.Text = s;
    }

   private void ДатаНародження_ValueChanged(object sender, EventArgs e)
    {formRez();
    }
}
```

Що означає зарезервоване слово *base*?

2.6. Завантажте додаток на виконання. Як відрізняється значення поля результату при редагуванні трьох перших полів і поля дати народження? Чому?

2.7. **Для відображення повної інформації про співробітника і при редагуванні перших трьох полів забезпечте виклик оновленої процедури *formRez()* в межах нащадка *FormPorodgeno* і з батьківського класу *FormBase*. Для цього в модулі батьківського класу *FormBase* перед описом процедури *formRez()* вкажіть зарезервоване слово *virtual* (якщо ви цього не зробили раніше), а в модулі класу *FormPorodgeno* перед описом цієї ж процедури – зарезервоване слово *override*.** Що змінилося у функціонуванні об'єкта форми *FormPorodgeno*? Яке зв'язування методів при цьому реалізується?

3. **Конкурс на 5 балів до рейтингу за найшвидше розв'язання завдання: забезпечте одним додатковим рядком коду відкриття при завантаженні програми не лише екземпляра форми *FormPorodgeno*, а й екземпляра форми *FormBase*.**