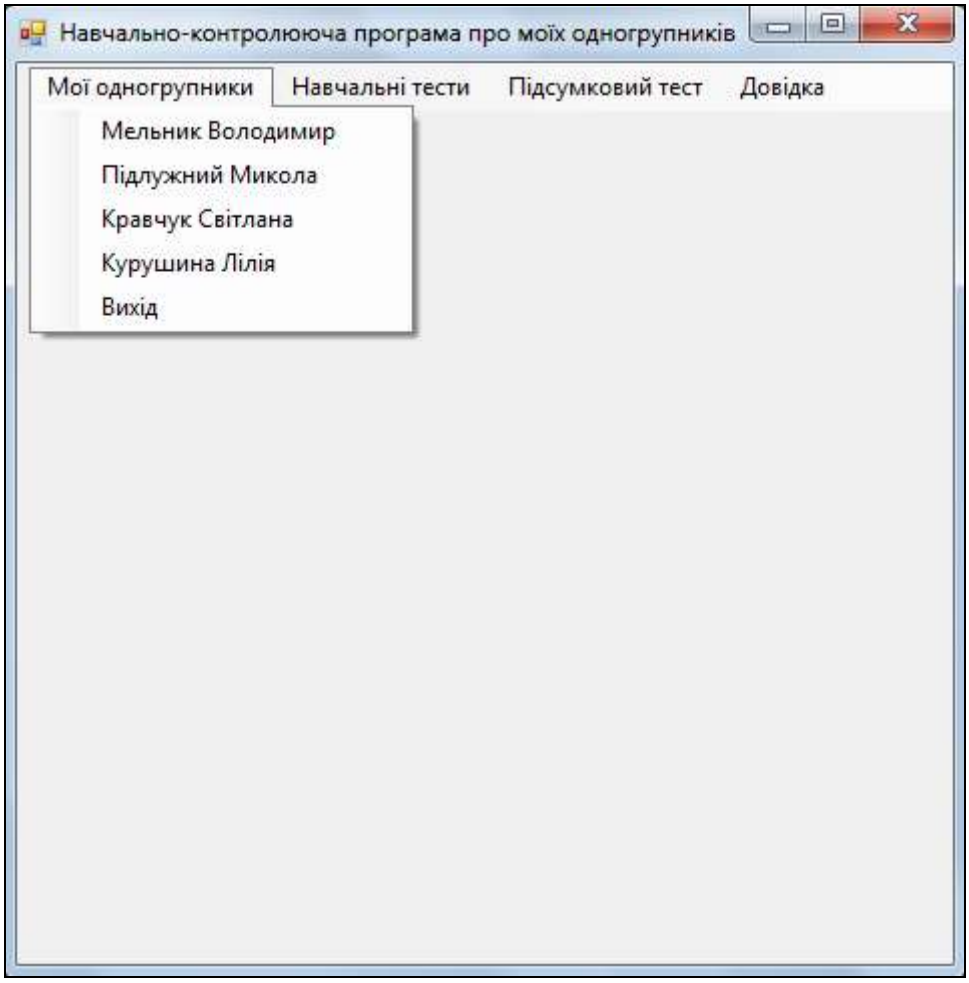
Завдання для розробки індивідуального навчально-дослідного проекту

1. Оберіть собі тему для розробки навчально-контролюючої програми. Створіть для її розробки новий проект. Стартову форму цього проекту перейменуйте на *FormMenu.cs*. Створіть у цій формі головне меню, передбачивши у ньому можливість вибору не менше трьох форм з теоретичними відомостями, не менше трьох навчальних тестів, підсумкового тесту та довідки (з відомостями про програму і про розробника). Наприклад, для програми про одногрупників це меню може виглядати так:



Розробка форм для подання теоретичних відомостей

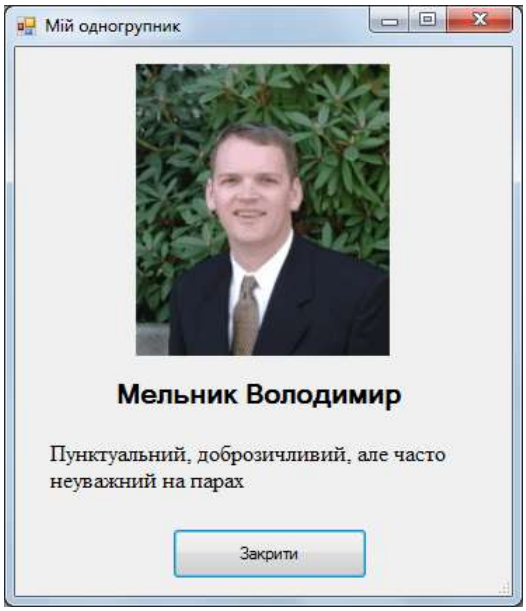
2. Вставте в ресурси проекту не менше трьох зображень для різних теоретичних відомостей.
3. Створіть у цьому проекті нову форму або імпортуйте сюди з проекту лабораторної роботи № 15 *Form2.cs* для відображення теоретичних відомостей. Перейменуйте її на *FormEducation.cs*. Забезпечте у цій формі відображення зображення і не менше двох текстових полів з описом теоретичних відомостей та різними характеристиками (назва зображення та тексти полів мають передаватися параметрами у форму). Забезпечте виклик цієї форми не менше ніж з трьох пунктів теоретичних відомостей головного меню з відповідними параметрами. Наприклад, в програмі про одногрупників конструктор цієї форми може виглядати так:

```
public partial class FormEducation : Form
 {public FormEducation(string Spivrob, string msg, string nameImg)
    {  InitializeComponent();
       this.Spivrob.Text = Spivrob;
       Message.Text = msg;
       System.Resources.ResourceManager rm =
            Properties.Resources.ResourceManager;
       Picture.Image = (Image)rm.GetObject(nameImg);
     }
  ...
}        .
```

Код для створення об’єкта класу цієї форми при виборі першого підпункту першого пункту меню може бути таким:

```
private void тема1ToolStripMenuItem_Click(object sender, EventArgs e)
 {(new FormEducation("Мельник Володимир",
   "Пунктуальний, доброзичливий, але часто неуважний на парах", "Foto1")).ShowDialog(); }
```

Тоді при виконанні програми вибір цього підпункту призведе до відображення такої форми:



Функціонування інших підпунктів першого пункту головного меню для створення аналогічних вікон з даними інших одногрупників забезпечується аналогічно.

4. Забезпечте відображення вікна кожних теоретичних відомостей лише один раз, як в лабораторній роботі № 16: якщо таке вікно не створено – його потрібно створити, якщо створено і згорнути – треба розгорнути, якщо не активне – сфокусувати. Для цього створіть у формі головного меню вказівки на кожну з форм теоретичних відомостей і модифікуйте процедуру виклику кожної з цих форм, наприклад, так:

```
public partial class FormMenu : Form
 {     Form TheorF1 = null, TheorF2 = null, TheorF3 = null;
       public FormMenu()
       {
           InitializeComponent();
       }

       private void тема1ToolStripMenuItem_Click(object sender, EventArgs e)
       {  if (TheorF1 == null || TheorF1.IsDisposed)
           { TheorF1 = new FormEducation("Мельник Володимир",
               "Пунктуальний, доброзичливий, але часто неуважний на парах", "Foto1");
             TheorF1.Show(); }
          else
           if (TheorF1.WindowState == FormWindowState.Minimized)
               TheorF1.WindowState = FormWindowState.Normal;
           else
            if (!TheorF1.Focused)
             TheorF1.Focus(); }
... }
```

Конструювання та організація взаємодії форм навчальних тестів

5. З метою забезпечення в подальшому можливості отримання кількості набраних балів з кожного вікна тестування опишіть під кодом будь-якої форми чи в окремому файлі відповідний інтерфейс, наприклад, так:

```
interface IQuationTest
 {double balQuestion(); }
```

6. Для підрахунку з різних форм загальної кількості пройдених питань, кількості набраних балів та максимально можливих балів, а також для відображення результатів кожного тесту створіть на

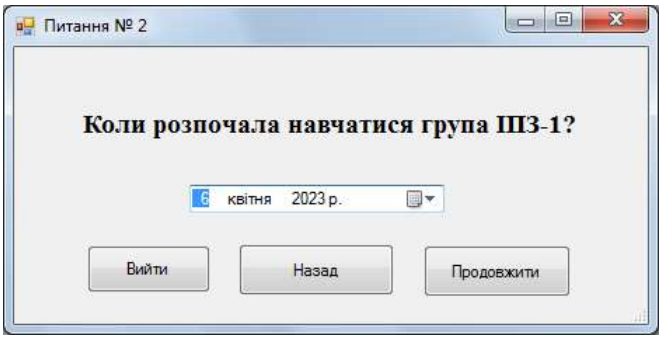
початку статичного класу *Program* у файлі *Program.cs* відповідні статичні змінні та метод, наприклад, так:

```
internal static class Program
   {   public static int countQuestionTest1;
       public static double balTest1, maxBalTest1;

       public static void resTest1()
       {MessageBox.Show("За результатами першого тесту ви відповіли на " +
         countQuestionTest1 + " питань і набрали\n" +
         balTest1.ToString() + " балів з " + maxBalTest1 + " можливих", "Результати",
         MessageBoxButtons.OK, MessageBoxIcon.Information); }
    ... }
```

7. Створіть форми з різними елементами керування для окремих питань кожного навчального тесту. Наприклад, такі:

7.1. Форма для вибору дати для відповіді на питання може виглядати так:



Функція для підрахунку балів **описується в середині класу цієї форми**, яка аналізує рік, місяць та день обраної дати, може виглядати так:

```
public partial class Question3_2 : Form, IQuationTest
 { public Question3_2()
    {InitializeComponent(); }

   public double balQuestion()
    {double bal = 0;
     if (dateStart.Value.Year==2021 && dateStart.Value.Month == 9 &&
         dateStart.Value.Day == 1)
      {MessageBox.Show("Ви дали правильну відповідь та набрали 1 бал", "Питання № 2",
                       MessageBoxButtons.OK, MessageBoxIcon.Information);
       bal = 1; }
     else
      {MessageBox.Show("Ви помилилися!\nГрупа розпочала навчання 1.09.2021\n\n"+
                      "Набрано " + bal.ToString() + " балів з одного", "Питання # 2",
                      MessageBoxButtons.OK, MessageBoxIcon.Information); }
     Program.countQuestionTest1++;
     Program.balTest1 += bal;
     Program.maxBalTest1 += 1;
     return bal; }
```

Кнопка *Вийти* має завершувати тестування, тобто виводити правильну відповідь на поточне питання, закривати поточну форму і виводити загальні результати за навчальний тест. Тому процедура обробки її натиснення може бути такою:

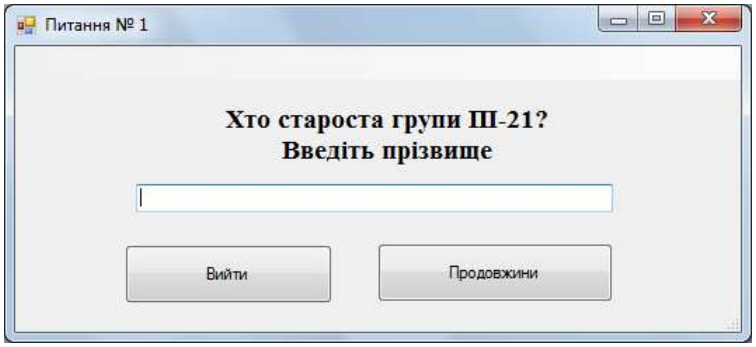
```
   private void button4_Click(object sender, EventArgs e)
    {balQuestion();
     Close();
     Program.resTest1(); }
```

Кнопка *Продовжити* має виводити правильну відповідь на поточне питання, закривати поточну форму та створювати і виводити форму наступного питання:

```
   private void button1_Click(object sender, EventArgs e)
    {balQuestion();
     Close();
     new Question3_3().Show(); }
```

Обробка натиснення кнопки *Назад* виконується аналогічно.

7.2. Форма для введення відповіді від користувача у текстовому полі може бути такою:

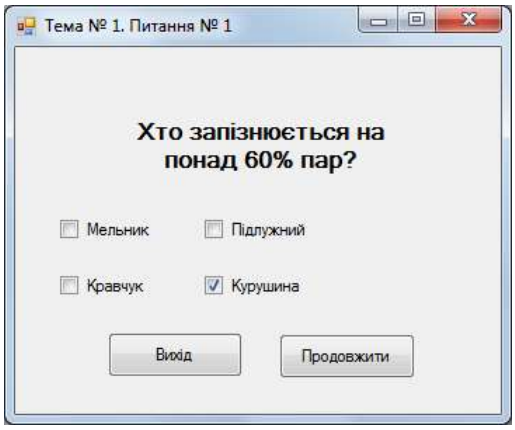


Тоді функція для підрахунку балів на це питання може виглядати так (тут *Starosta* – ім'я візуального об'єкта класу *TextBox*):

```
public double balQuestion()
 {double bal = 0;
  if (Starosta.Text.ToUpper()=="ТЕНЬКОВ")
   {MessageBox.Show("Ви дали правильну відповідь та набрали 1 бал", "Питання № 2",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
    bal = 1; }
  else
   {MessageBox.Show("Ви помилилися!\nСтароста групи - Теньков\n\nНабрано " +
                    bal.ToString() + " балів з одного", "Питання № 2",
                    MessageBoxButtons.OK, MessageBoxIcon.Information); }
  Program.countQuestionTest1++;
  Program.balTest1 += bal;
  Program.maxBalTest1 += 1;
  return bal; }
```

Кнопки переходу в цій та наступних формах пункту реалізуються, як в попередній формі.

7.3. Форма з можливістю вибору декількох варіантів реалізується за допомогою прапорців класу *CheckBox* і може виглядати так:

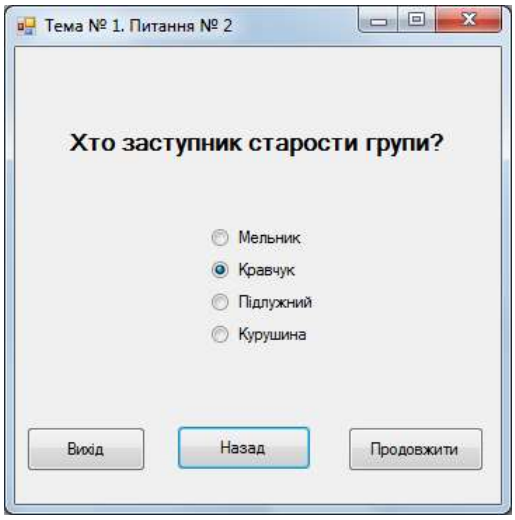


Нехай правильна відповідь на це питання реалізується встановлення першого і другого прапорців. Тоді за встановлення кожного з цих прапорців і за зняття третього і четвертого прапорців будемо нараховувати 0.25 бала, щоб з одного боку за це питання можна було набрати максимум 1 бал, а з іншого при повністю неправильній відповіді – не отримати від'ємних балів. Функція для підрахунку балів на це питання може бути такою:

```
public double balQuestion()
 {double bal = 0;
  if (check1.Checked && check2.Checked &&
      !check3.Checked && !check4.Checked)
   {MessageBox.Show("Ви дали правильну відповідь та набрали 1 бал", "Питання # 1",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
    bal = 1; }
  else
   {if (check1.Checked) bal += 0.25;
```

```
    if (check2.Checked) bal += 0.25;
    if (!check3.Checked) bal += 0.25;
    if (!check4.Checked) bal += 0.25;
    MessageBox.Show("Ви помилилися!\nНа понад 60% пар спізнюються Мельник "+
              "та Кравчук!\n\nНабрано "+bal.ToString()+ " балів з одного",
              "Питання # 1", MessageBoxButtons.OK, MessageBoxIcon.Information); }
  Program.countQuestionTest1++;
  Program.balTest1 += bal;
  Program.maxBalTest1 += 1; ;
  return bal; }
```

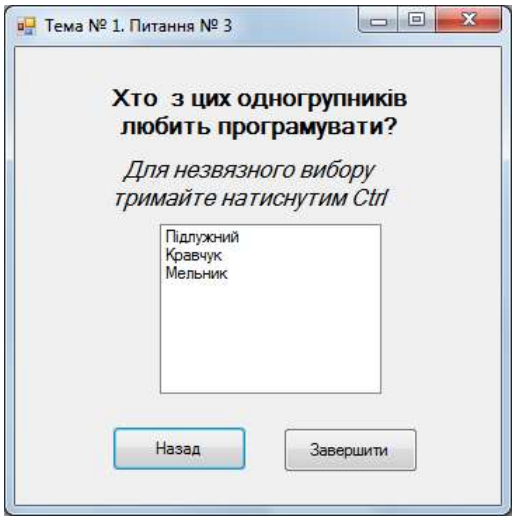
7.4. Форма з забезпеченням вибору одного варіанту з декількох заданих за допомогою перемикачів реалізується за допомогою об'єктів класу *RadioButton* і може бути такою:



Правильна відповідь на таке питання реалізується вибором відповідного перемикача. Якщо задати третьому перемикачу з цієї форми ім'я *zastupnyk*, то функцію для підрахунку балів за це питання можна реалізувати так:

```
public double balQuestion()
 {double bal = 0;
  if (zastupnyk.Checked)
   {MessageBox.Show("Ви дали правильну відповідь та набрали 1 бал", "Питання # 2",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
    bal = 1; }
  else
   {MessageBox.Show("Ви помилилися!\nЗаступник старости групи - Підлужна!" +
            "\n\nНабрано " + bal.ToString() + " балів з одного", "Питання # 2",
            MessageBoxButtons.OK, MessageBoxIcon.Information); }
  Program.countQuestionTest1++;
  Program.balTest1 += bal;
  Program.maxBalTest1 += 1;
  return bal; }
```

7.5. І, нарешті, форма з можливістю незв'язного вибору декількох позицій у списку реалізується за допомогою об'єкта класу *ListBox*, в якому тексти позицій задаються в колекції *Items*, а сама можливість незв'язного вибору забезпечується встановлення у властивості *SelectionMode* значення *MultiExtended*. Ця форма може виглядати так:



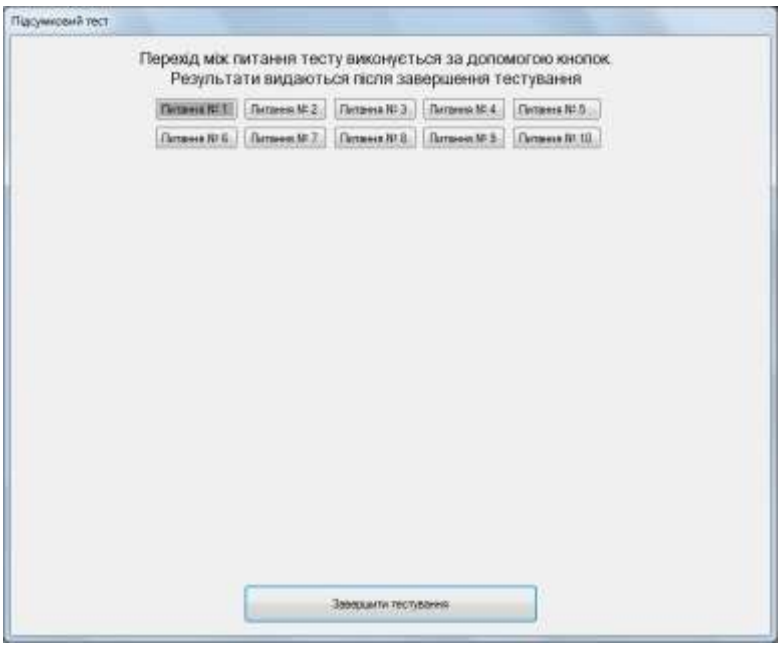
Якщо правильна відповідь на питання цієї форми полягає у виборі першого та третього елементів списку, то функція для підрахунку набраних балів може бути такою:

```
public double balQuestion()
{double bal = 0;
 if (listStudent.SelectedIndices.Count==2  &&
     listStudent.SelectedIndices[0]==0 &&
     listStudent.SelectedIndices[1]==2)
  {MessageBox.Show("Ви дали правильну відповідь та набрали 1 бал", "Питання № 3",
                   MessageBoxButtons.OK, MessageBoxIcon.Information);
   bal = 1; }
 else
  {if (listStudent.GetSelected(0)) bal += 0.33;
   if (!listStudent.GetSelected(1)) bal += 0.33;
   if (listStudent.GetSelected(2)) bal += 0.34;
   MessageBox.Show("Ви помилилися!\nПрограмувати люблять Підлужний та Мельник!" +
              "\n\nНабрано " + bal.ToString() + " балів з одного", "Питання № 3",
              MessageBoxButtons.OK, MessageBoxIcon.Information); }
 Program.countQuestionTest1++;
 Program.balTest1 += bal;
 Program.maxBalTest1 += 1;
 return bal; }
```

В цій функції *listStudent* – це ім'я об'єкта списку, властивість *SelectedIndices* – масив індексів обраних елементів, а *GetSelected* – це метод, що вказує на виділення елемента з вказаним індексом, хоча зрозуміло, що підрахувати набрані бали можна й за допомогою інших властивостей та методів списку.

Розробка форми та вкладених форм підсумкового тесту

8. Створіть форму з не менше, ніж десятьма перемикачами класу *RadioButton* у верхній чи лівій її частині для вибору питань підсумкового тесту. Для відображення цих перемикачів у вигляді пов'язаних кнопок встановіть для них у властивості *Apperance* значення *Button*. Задайте підписи цим кнопкам з зазначенням номера питання, а для властивості *Checked* значення *false* (тобто, по замовчуванню кнопки не натиснуті). В нижній частині форми створіть кнопку для завершення підсумкового тестування. Самостійно приховайте для цієї форми кнопку закриття. Виглядати ця форма може так:



9. Для зберігання вибраних відповідей користувача під час переходів між різними питаннями створіть в конструкторі форми підсумкового тесту не менше десяти форм окремих питань (по одній для кожної кнопки вибору) та збережіть вказівки на них в масиві вказівок на форми. Ці форми будуть приховуватися/відображатися при переходах між кнопками вибору, а закриватися – лише після виводу загальних результатів тестування. Для створення цих вкладених форм для окремих питань і зберігання вказівок на них в масиві реалізуйте також процедуру *createForm*, яка за індексом буде створювати відповідну форму, приховувати наявні в ній кнопки і межу для створення ефекту належності формі підсумкового тестування. Це можна реалізувати, наприклад, так:

```
public partial class FinalTest : Form
{   const int countQuestion = 10; // кількість питань підсумкового тестування
    Form[] masForm = new Form[countQuestion];

    public FinalTest()
    {   InitializeComponent();
        for (int i = 0; i < countQuestion; i++)
            createForm(i); }

    private void createForm(int index)
    {   Form f=null;
        switch (index)
        { case 0:f = new Quation1_3(); break;
          case 1:f = new Quation1_1(); break;
          case 2:f = new Quation1_2(); break;
          ...
        }
        if (f != null)
        { foreach (Control c in f.Controls)
                if (c is Button)
                    c.Visible = false;
          f.FormBorderStyle=FormBorderStyle.None;
        }
        masForm[index] = f; }
... }
```

10. Для відображення при натисненні обраної кнопки вибору відповідного їй питання встановіть для кожної кнопки відповідний їй послідовний унікальний індекс, починаючи від нуля, у властивості *Tag*. Створіть процедуру обробки натиснення довільної кнопки вибору і забезпечте виклик цієї процедури всіма іншими кнопками вибору. Ця процедура має приховувати всі виведені пов'язані форми питання і відображати лише форму, відповідну властивості *Tag* обраної кнопки вибору. Виглядати вона може, наприклад, так:

```
private void radioButton1_Click(object sender, EventArgs e)
 {// змінюємо колір фону кнопки, вказуючи на те, що питання переглядалося
```

```
((RadioButton)sender).BackColor = System.Drawing.SystemColors.AppWorkspace;
//приховуємо всі виведені форми питань
for (int i = 0; i < countQuestion; i++)
 if (masForm[i] != null && !masForm[i].IsDisposed && masForm[i].Visible)
   masForm[i].Visible = false;
//відображаємо лише форму, відповідну натиснутій кнопці
int index = Convert.ToInt32(((RadioButton)sender).Tag);
Form f = masForm[index];
//якщо форма питання закрита – то створюємо її знову
if (f==null || f.IsDisposed)
 {createForm(index);
  f = masForm[index]; }
if (f == null) return;
f.Show(); }
```

11. Використовуючи інтерфейс *IQuationTest*, реалізуйте обробку події натиснення кнопки завершення підсумкового тестування, забезпечивши в ній підрахунок набраних балів. Після цього в процедурі виведіть результати тестування, закрийте саму форму тестування та пов'язані форми питань. Реалізувати це можна, наприклад, так:

```
private void button1_Click(object sender, EventArgs e)
 {int countQuestionTest = 0;
  double sumBalTest = 0;
  for (int i = 0; i < countQuestion; i++)
   if (masForm[i] != null && !masForm[i].IsDisposed &&
       ((RadioButton)(Controls["RadioButton" + (i + 1).ToString()])).BackColor ==
        System.Drawing.SystemColors.AppWorkspace)
    {countQuestionTest++;
     sumBalTest += ((IQuationTest)masForm[i]).balQuestion(); }
  MessageBox.Show("За результатами підсумкового тесту ви відповіли на " +
                  countQuestionTest + " питань і набрали " +
                  sumBalTest.ToString() + " балів", "Результати",
                  MessageBoxButtons.OK, MessageBoxIcon.Information);
  Close();
  for (int i = 0; i < countQuestion; i++)
   if (masForm[i] != null && !masForm[i].IsDisposed)
    masForm[i].Close(); }
```

12. Самостійно створіть форми для підпунктів *Про розробника* та *Про програму* пункту головного меню *Довідка* розробленої навчально-контролюючої програми.