ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ДЕННОЇ ОСВІТИ

ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА КОМП‘ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Допускається до захисту

Завідувач кафедри_____О.ОЛЬХОВСЬКА
(підпис)

«_____»_____2022 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО ДИПЛОМНОЇ РОБОТИ

на тему
Розробка програмного забезпечення з теми «Цикли мови C#»

зі спеціальності 122 «Комп’ютерні науки»
освітня програма «Комп’ютерні науки»
ступеня магістра

Виконавець роботи Недбайло Ярослав Ігорович _____ «___»_____2022р.
(підпис)
Науковий керівник к.ф.-м.н., доц., Чілікіна Тетяна Василівна
_____ «___»_____2022р.
(підпис)

ПОЛТАВА 2022 р.

ЗМІСТ

ВСТУП

Для реалізації своїх функцій у роботі, студент має під час навчання набути певних практичних навичок та теоретичних знань. Важливими умовами підвищення розвитку теорії і практики є створення системи завдань, їх послідовність, різноманітність та складність.

Комп'ютерний тренажер – це навчально-тренувальний пристрій, який симулює ситуацію, створює обставини наближені до реальних. Ця комп'ютерна програма створена щоб надати студенту певні уміння та навички

Метою роботи є побудова тренажеру з теми «Цикли мови C#» дистанційного навчального курсу та розробка його програмного забезпечення.

Об'єктом розробки є програмна реалізація навчального тренажеру за темою «Цикли мови програмування C#».

Предмет розробки – програмний продукт, що реалізує тренажер для закріплення знань за темою «Цикли мови C#».

При реалізації комп'ютерної програми використано мову програмування C++ і середовище розробки Visual Studio 2013.

Пояснювальна записка до дипломної роботи містить в собі: вступ, постановку задачі, інформаційний огляд, теоретичну та практичну частину, висновки, список літератури та додаток.

Пояснювальна записка містить: 40 сторінку, 24 рисунки, 1 додаток (на 23 сторінках) та 11 літературних джерел.

1 ПОСТАНОВКА ЗАДАЧІ

1.1 Постановка задачі для розробки тренажеру

Оскільки потрібно розробити навчальний тренажер «Цикли мови програмування C#», як складову дистанційного навчання з програмування, то потрібно виконати наступні умови

- провести порівняльний аналіз вже існуючих тренажерів,виділити їх позитивні та негативні сторони;
- розглянути цикли у мові програмування C#;
- розглянути теоретичні відомості про цикли;
- розробити алгоритм тренажера, що дозволить закріпити знання з теми «Цикли мови C#»;
- обгрунтувати вибір програмного забезпечення;
- описати процес програмної реалізації;
- описати роботу тренажера.
- записати інструкцію користувача

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Огляд робіт для виявлення сильних і слабких сторін тренажерів

Під час огляду програм для самостійного навчання були розглянуті найбільш актуальні навчальні тренажери інших студентів, які вже впровадженні в систему дистанційного навчання MOODLE ПУЕТ.

Першим було розглянуто тренажер Самовика С. М. з темою роботи «Розробка алгоритму та програмного забезпечення тренажера з теми «Угорський метод в задачі про призначення» дистанційного навчального курсу «Методи оптимізації та дослідження операцій» [3].

Далі - результати роботи Марченка Д. А. «Програмування навчального тренажера з теми «Переставні многогранники» дистанційного навчального курсу «Елементи комбінаторної оптимізації» [4].

Наступна була переглянута робота Усольцева С. І. з темою навчального тренажера «Розробка алгоритму, програми та дослідження тренажеру з теми «Модифікований симплекс-метод» дистанційного навчального курсу «Методи оптимізації та дослідження операцій» [5].

Далі розглянуто навчальний тренажер Ставкового М. Ю. на тему «Метод гілок та меж в задачі про найкоротший шлях» дисципліни «Методи оптимізації та дослідження операцій» [6].

Ще один тренажер Томченка О. В. на тему «Перший алгоритм Гоморі» з навчальної дисципліни «Методи оптимізації та дослідження операцій» [7].

Заключним було розглянуто навчальний тренажер Кильника В. В. на тему «Сортування методом перемішування» для дистанційного навчального курсу «Алгоритми та структури даних». Єдиний із розглянутих навчальних тренажерів, який написаний з допомогою мови розмітки веб-сторінo HTML, а для опису використано спеціальна мова CSS [8].

Позитивні риси оглянутих програм

Перевагами навчального тренажеру Самовика С. М. [3] є:

1. В наявності довідка для отримання інформації щодо користування тренажером, а також блок-схема і загальна інформація про розробку навчального тренажеру.
2. Доволі ручний інтерфейс у використанні користувачем.
3. Зрозуміло та безпомилково побудовані навчальні питання.
4. Послідовність питань побудована відповідно до розгляду лекції.

Переваги навчального тренажеру Марченка Д. А. [4] є:

1. Після вибору неправильної відповіді з’являється вірна.
2. На початку користування навчальним тренажер є можливість обрати одну з умов для тренінга.
3. Коли закінчено розв’язування, користувач отримує інформацію про правильність відповіді на кожному кроці.

Перевагами навчального тренажеру Усольцева С. І. [5] є:

1. Написання у поле вводу лише цифрової умови, що значно скорочує можливість створення помилки.

Перевагами навчального тренажеру Ставкового М. Ю. [6] є:

2. На початку роботи в навчальному тренажері користувач має змогу обрати один з трьох прикладів для проходження.
3. Створена можливість перегляду теоретичної інформації у довідці.

Переваги навчального тренажеру Томченка О. В. [7] є:

1. Наявність умови задачі на кожному етапі виконання завдання.
2. Можливість повернутись до попереднього питання.

Переваги навчального тренажеру Кильника В. В. [8] є:

1. Можливість створити приклад або розв’язати свій власний.
2. Виділення правильних та неправильних відповідей різним кольором.
3. Можливість використання вірно обчислених даних за потребою .

4. Демонстрація теоретичних відомостей за потребою користувача.

2.3 Вади розробок з розглянутих робіт

Недоліками навчального тренажеру Самовика С. М. [3] є:

1. Перехід до наступного кроку навчального тренажеру можливий тільки після введення всіх правильних відповідей. Також, коли вірна відповідь з’явилася на екрані, то навчання не є досить коректним.
2. Введення у клітини будь-які символи, що призводить до помилок при випадковому натисканні клавіші.
3. Занадто складним є введення відповідей для перевірки.

Недоліками навчального тренажеру Марченка Д. А. [4] є:

1. Застарілий дизайн інтерфейсу.
2. Відсутність підказок до завдань або посилання на теоретичні відомості.
3. Одноманітність типів завдань.

Недоліками навчального тренажеру Усольцева С. І. [5] є:

1. Неефективна поступова перевірка відповіді, що є марнуванням часу користувача.
2. Повна відсутність якихось довідок з теми роботи.

Недоліками навчального тренажеру Ставкового М. Ю. [6] є:

1. Відсутність можливості переходу до наступного кроку, якщо відповідь є неправильною.
2. Однотипні завдання в тренажері.

Недоліки навчального тренажеру Томченка О. В. [7] є:

1. Відсутністя можливість переходу до наступного кроку завдання без правильно вирішеного попереднього.
2. Перше питання навчального тренажера має не зрозуміле заповнення даних в порожні клітини.

Недоліками начального тренажеру Кильника В. В. [8] є:

1. Іноді не коректне відображення веб-сторінки.

2. Бракує можливості вибирати кількість початкових значень.

3. Велика кількість файлів при завантаженні навчального тренажеру.

Спільними недоліками розглянутих навчальних тренажерів [3-8] є:

1. Навчальні тренажери необхідно завантажувати на персональний комп’ютер, а також додатково встановлювати додаткове програмне забезпечення для запуску.

2. Інтерфейс навчальних тренажерів не є сучасним.

2.4 Алгоритмізація роботи тренажеру

Часто, при програмуванні задач, потрібно, щоб одна й таж послідовність команд виконувалась декілька разів. Для цього у мовах програмування застосовується поняття циклічного процесу або циклу. Алгоритм, в якому певна послідовність команд повторюється декілька разів з новими вхідними даними називається циклічним.

Циклічний процес організовується з допомогою операторів циклу. Мова програмування C# має в наявності зручні для роботи оператори циклу.

Опис операторів циклу у мові C#:

У мові C# існує 4 види операторів циклу

• цикл for;

• цикл while з передумовою;

• цикл do…while з постумовою;

• цикл foreach.

Кожен з операторів циклу має свої особливості застосування. Будь-який з вищенаведених операторів циклу може бути замінений іншим.

Цикл for

Найуніверсальніший з усіх трьох операторів циклу мови програмування C#. Якщо потрібно виконати точку кількість дій(ітерацій), то цикл for підходить якнайкраще.

Загальний форма оператора циклу for:

for(ініціалізація; вираз; приріст) оператор;

де

• ініціалізація – операція присвоювання, в якій встановлюється початкове значення змінної циклу. Ця змінна є лічильником, який керує роботою циклу. Кількість змінних, що керують циклом for, може дві і більше;

• вираз – умовний вираз, в якому перевіряється значення змінної циклу. На цьому етапі визначається подальше виконання циклу;

• приріст – визначає, як буде змінюватись значення змінної циклу після кожної ітерації.

Всі ці елементи циклу for повинні відділятися крапкою з комою.

Якщо цикл for призначений для багатократного виконання не одного оператора, а операторного блоку, то його загальний формат має такий вигляд:

```
for(ініціалізація; вираз; приріст)
{
        // послідовність операторів
       //
}
```

Цикл for виконуватиметься доти, доки обчислення елемента виразу дає істинний результат. Як тільки цей умовний вираз стане помилковим, цикл завершиться, а виконання програми продовжиться з оператора, що є наступним за циклом for.

Цикл while

Цей цикл називається циклом з передумовою. Його форма у загальному вигляді наступна:

```
while (вираз)
{
        // послідовність операторів
```

```
        // …
}
```

де вираз – будь-який допустимий вираз у мові C#.

Послідовність операторів виконується до тих пір, поки умовний вираз повертає значення true. Як тільки вираз стає рівним false, виконання циклу while припиняється і управління передається наступному за циклом while оператору.

Цикл do while

Цикл do … while доцільно використовувати у випадках, коли ітерацію потрібно зробити хоча б 1 раз. На відміну від циклів for та while, у циклі do…while умова перевіряється при виході з циклу (а не при вході в цикл).

Загальна форма оператора циклу do…while:

```
do
{
        // послідовність операторів
        // …
}
while (вираз) ;
```

Що таке нескінчений цикл?

Нескінчений цикл – це цикл, який ніколи не закінчується.

При програмуванні циклічних процесів, програміст помилково може написати код циклу, який ніколи не закінчується.

Крім того, інколи потрібно, щоб цикли містили код спеціального завершення з допомогою інструкції break.

Приклад 1. Нескінчений цикл з оператором for:

```
for (; ;)
{
    // послідовність операторів
    // ...
}
```

Приклад 2. Нескінчений цикл while.

```
int x;
x = 5;
while (x<10)
{
    // нескінчений цикл while, значення x не зростає,
    // умова x<10 виконується завжди, неможливо вийти з циклу
}
```

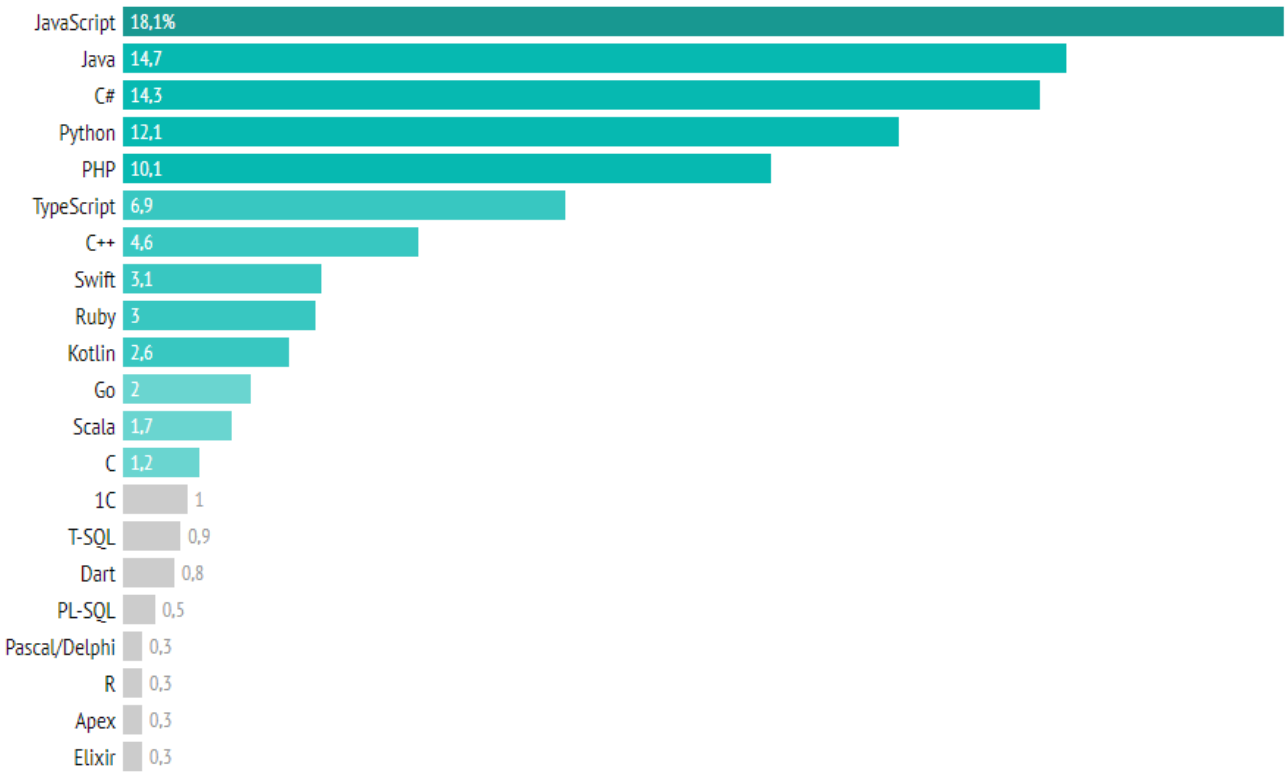
Вищенаведений фрагмент коду, з логічної точки зору, є помилковим. Такий цикл виконується без кінця. Тому що значення змінної x не зростає. Тобто x завжди буде менше 10 .

3 ТЕОРЕТИЧНА ЧАСТИНА

3.1 Обгрунтування вибору програмних засобів

В наш час існує багато мов програмування, сказати що якась краща або гірша було б напевно неправильно. Мова програмування це форма вираження людських думок, так само як для передавання візуальної інформації малюють, щоб порахувати використовують числа, а щоб передати словесну інформацію пишуть. Те ж саме стосується світу програмування — універсального набору інструментів просто-напросто не існує. Усі мови програмування створені для досягнення максимальної ефективності конкретних задач. Для кожного проекту є свій набір інструментів, котрий найкраще відпрацює саме для нього.

«Мова програмування C# була створена компанією Microsoft в 2000 році. Вона багато чого успадкувала від своїх батьків(C++, Java, Basic), але і привнесла багато чого нового. C# до сих пір підтримується Microsoft, і компанія періодично додає нові функції вдосконалюючи цю мову програмування. Користуючись світовою популярністю, мова ще довгий час матиме підтримку, нижче зображено таблицю з рейтингом мов програмування за актуальністю:



Ще однією з переваг мови C# є швидкість і можливість великого прибутку. Цією мовою написане ядро ОС Linux, Unix, різні бібліотеки, інтерпретатори багатьох сучасніших мов програмування, а також один із популярних ігрових рушіїв – Unity.

Перелік випадків використання C#:

- Мова програмування C# переважно використовується для створення корпоративного програмного забезпечення, фінансових проектів, наприклад для банків і бірж, зокрема мобільних додатків, хмарних сервісів.
- C# у порівнянні з Java легше взаємодіє, з кодом програм, написаних на інших мовах. І саме на C# часто пишуться розширення для інших мов програмування, які використовуються у якості прошарку між бібліотекою C# і мовою, можливості якої під конкретні цілі планується розширювати.
- Доволі популярний blockchain на C#
- C# широко використовується в розробці ігор на Unity. Ви коли-небудь чули про Unity? Так, Unity найпопулярніший ігровий двигунець. Це означає, що сотні тисячі ігор, включаючи найпопулярніші створювалися за допомогою C#.
- C# добра під роботу із залізом, так званий embedded. Embedded system — спеціалізована комп'ютерна система або обчислювальний пристрій, призначений для виконання обмеженої кількості функцій, по Вікіпедії(світлофори, касові апарати, торгові автомати, телевізійні приставки, контрольно-вимірювальні прилади тощо.)
- популярна мова програмування C# однаково хороша для IoT, де так само кожен байт і кожна мілісекунда рахуються. IoT (Internet of Things) — це концепція всеохоплюючого інтернету, підключення до інтернету

холодильників, кондиціонерів, автомобілів і навіть кросівок з метою забезпечити своєму власникові більший комфот.

- наука та її прикладне застосування наприклад проведення складних експериментальних розрахунків, криптографія, розпізнавання образів тому подібне.

C# строго типізована, її простіше вивчати початківцям. Слід відзначити, що мова програмування високорівнева, це означає, що вона дещо схожа на англійську. Мова програмування C# має строгу статичну типізацію, підтримує поліморфізм, перевантаження операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML. Перейнявши багато що від своїх попередників — мов C++, Delphi, Модула і Smalltalk — у C#, опираючись на практику їхнього використання, навмисне виключили деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем у вище перелічених мовах програмування. C# керується чіткими строгими правилами використання пріоритетів, а значить абстрагує програмістів початківців робити дурні помилки і навчатися її, смаживши свій мозок. Синтаксис досить мінімалістичний C#. З ручним управлінням пам’яттю. Багатьом вказана обставина незручність, проте слідкування за правильністю функцій, розуміння передачі аргументів тісно пов’язане вивчення мови програмування C# (C Sharp). Оскільки синтаксис C# близький до C, C++ і Java, то, вільно володіючи C #, згодом можна вивчити і їх на одному подиху.

Цікавий факт: назва мови з’явилася під час проведення паралелі між мовами C# та C++. І цьому є два обґрунтування. Перше - знак "#" є складовою з чотирьох "+": C → C++ → C++++(C#). Друге — знак «#» є копією знака «дієз» у музичній нотації, що означає підвищення звуку, відповідного ноті, на півтон, за аналогією зі знаком «++» у мові C++, який позначає інкремент змінної. Оскільки знака «дієз» на клавіатурі немає, було вирішено використати знак «#». Але на друкованих матеріалах Microsoft використовує саме «дієз»».

3.2 Алгоритм роботи тренажеру

Крок 1. При запуску тренажеру відкривається головне вікно програми, і пропонується ознайомитись з довідковим матеріалом, або відразу перейти до тренінгу.

Крок 2. При натисканні на кнопку “Довідниковий матеріал” відкривається вікно з інформацією про цикл for.

Кроки 3-4. Перейти до наступного циклу можна натисканням клавіші “Продовжити”. Реалізована можливість повернутися до попереднього вікна.

Крок 5. У блок тренінгу можна потрапити з того ж вікна довідникового матеріалу натиснувши кнопку “Продовжити” при перегляді останнього циклу, або повернувшись у меню програми натиснувши кнопку “Головне меню” і “Тренінг” відповідно.

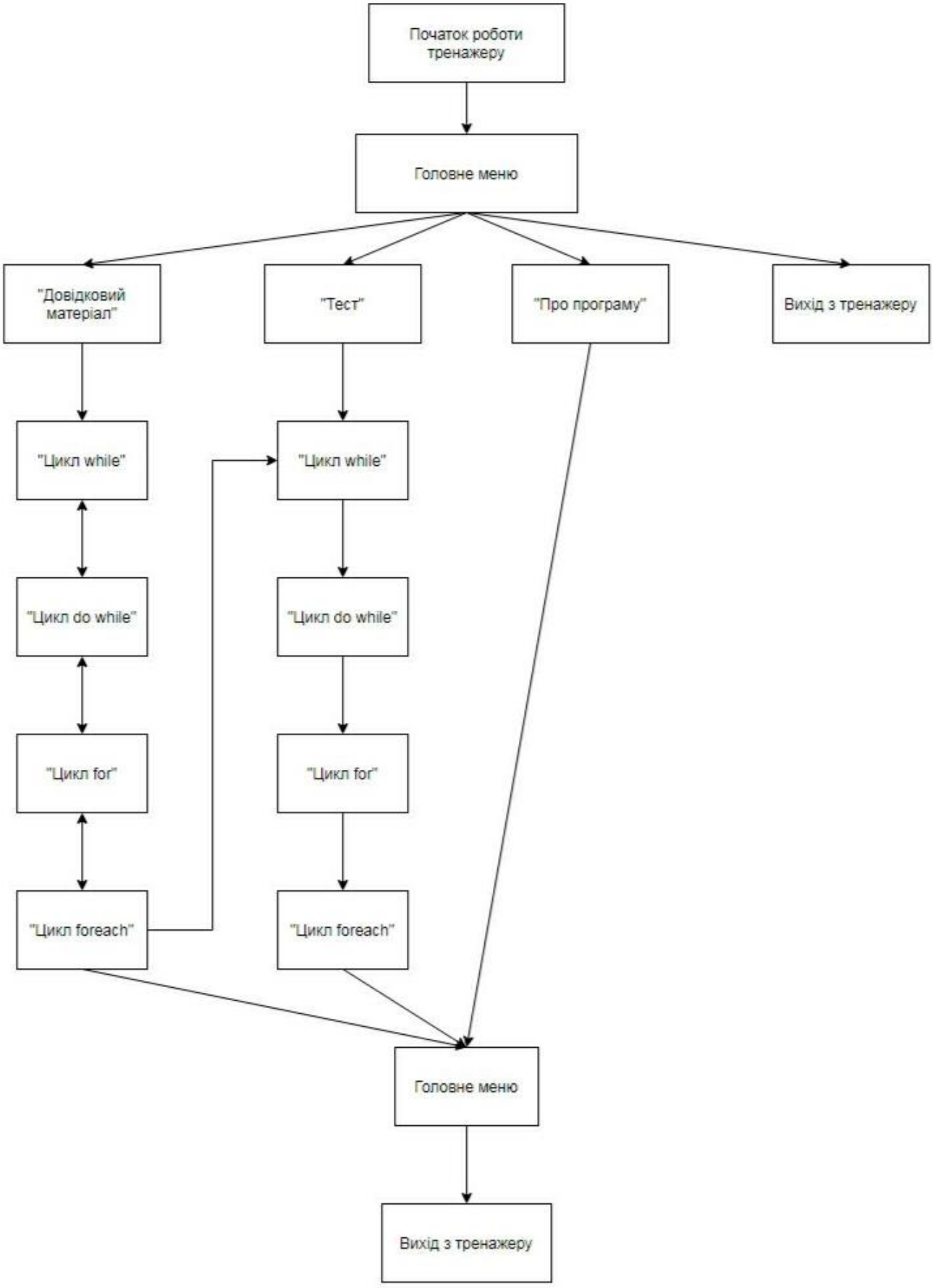
Крок 6. Відкриється вікно з п'ятьма питаннями про цикл for. Час для надання відповіді необмежений.

Кроки 7-8. Для переходу до наступного блоку питань потрібно відповісти щонайменше на 80% питань вірно і натиснути кнопку “Продовжити”.

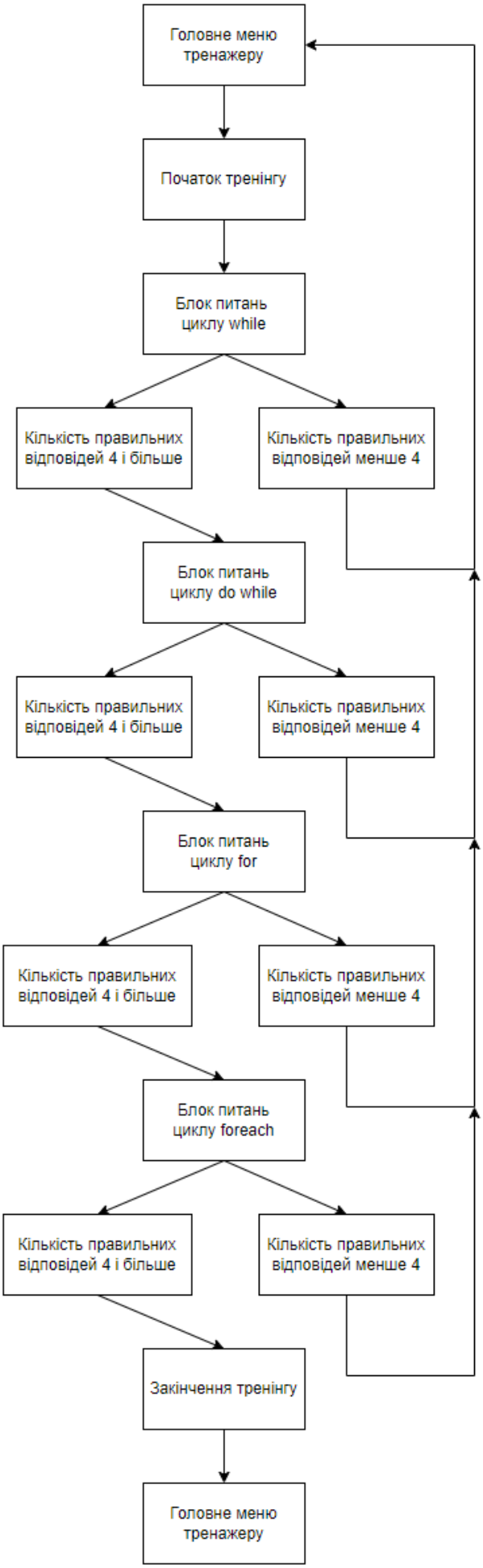
Крок 9. Після вибору відповідей на кожній сторінці і натискання кнопки “Продовжити” з'являється вікно з інформацією про кількість правильних відповідей. Після проходження всього тесту програма повертається в головне меню.

Кроки 10-11. При натисканні клавіші “Про програму” з’являється вікно з інформацією про тренажер. При натисканні клавіші “Вихід” тренажер закривається.

3.3 Схема роботи тренажеру



Детальна схема роботи тренінгу



4 ПРАКТИЧНА ЧАСТИНА

4.1 Опис процесу програмної реалізації

Для реалізації поставленої задачі було обрано об’єктно-орієнтована мову програмування C# із застосування вільного середовища розробки Visual Studio.

Розробку розпочато зі створення форми CLR, що містить інші елементи управління Windows, рис. (4.1).

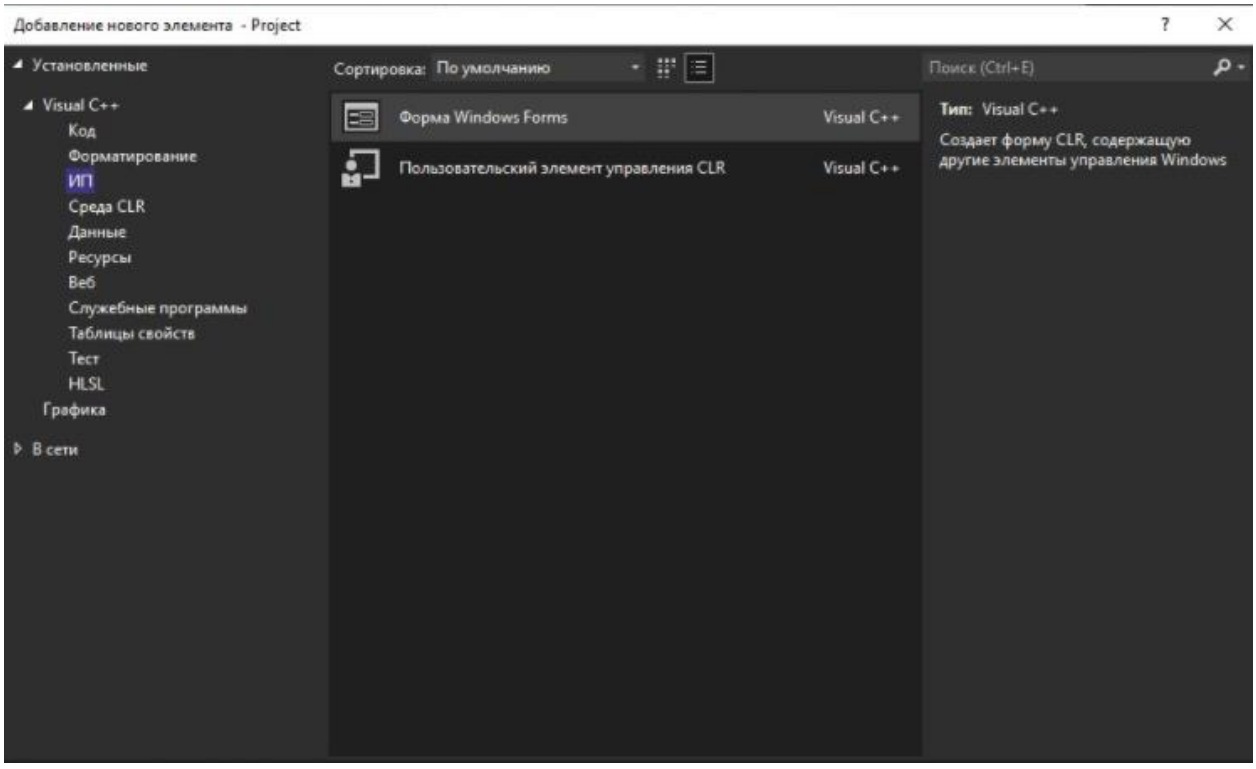


Рисунок 4.1 – Вибираємо тип файлу, ім’я і розташування.

Форма створена за допомогою платформи .NET Framework, рис. (4.2).

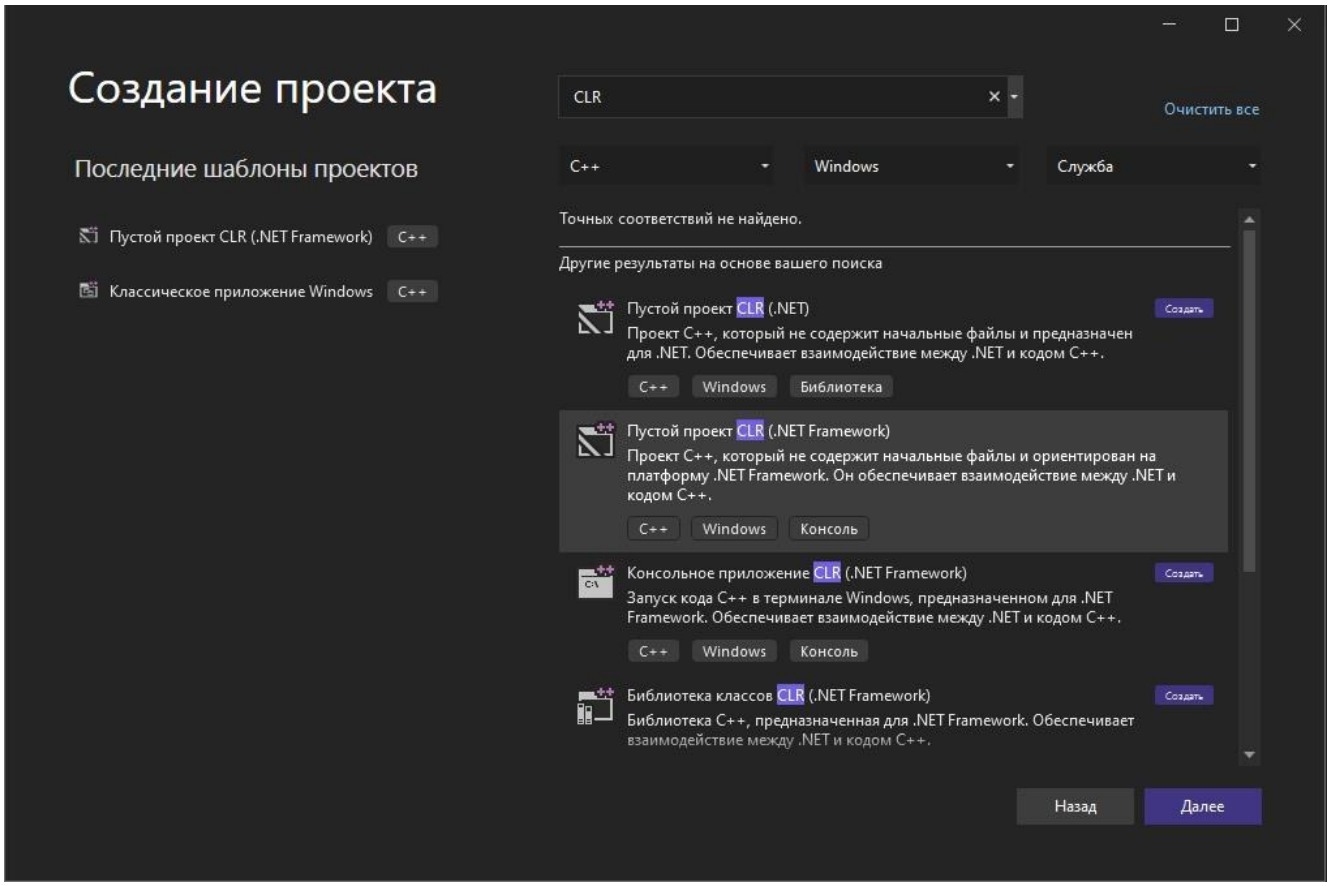


Рисунок 4.2 – Вибір проекту у Visual Studio.

Цей код відповідає за завантаження інформації при відкритті форми. Як тільки з'являється форма, код автоматично спрацьовує та відкриває теоретичні відомості (інформацію про цикли).

textBox1->Text = "Оператор циклу while або цикл while – цикл, що повторює одну і ту ж дію, поки умова продовження циклу while залишається істиною.";

textBox1->Text += Environment::NewLine + "Умова продовження циклу має бути істиною, як тільки умова стала хибною, виконується вихід з циклу. Також як і в умовних операторах вибору, фігурні дужки можуть опускатися в тому випадку, якщо тіло циклу – це один оператор. Але як правило в циклі виконується декілька

операторів, так як крім виконання корисної дії необхідно робити умову циклу while хибною, інакше цикл буде нескінченним, а це, в свою чергу, приведе до зависання програми."

textBox1->Text += Environment::NewLine + "Нижче наведено синтаксис циклу while:";

pictureBox2->Visible = false;

pictureBox3->Visible = false;

За перевірку інформації на цій форм відповідає наступна частина коду . Так як інформація не збігається з інформацією при запуску форми, вона змінюється на потрібну, описану в текстовому вікні, а також змінює картинку на іншу.

if (f = 1)

{

label1->Text = "Цикл do while";

pictureBox1->Visible = false;

pictureBox3->Visible = false;

textBox1->Text = "Цикл do while відрізняється від циклу while тим, що в do while спочатку виконується тіло циклу, а потім перевіряється умова продовження циклу. Через таку особливість do while називають циклом з постумовою. Таким чином, якщо умова do while завідомо хибна, то хоча б один раз блок операторів в тілі циклу do while виконається. В підсумку do while відрізняється від циклу while структурою.";

textBox1->Text += Environment::NewLine + "На початку циклу do while пишеться зарезервоване слово do, після йдуть фігурні дужки, які можна опускати, у разі використання одного оператора в тілі циклу do while.Після закривання

фігурної дужки, позначаючої кінець тіла циклу do while, стоїть умова циклу do while, після якої обов'язково потрібно ставити крапку з комою.";

pictureBox2->Visible = true;

}

Ця частина коду перевіряє інформацію алгоритмом перевірки аналогічно до попереднього коду. Якщо буде хоча б один збіг – відбудеться зміна тексту та картинки.

else

{

if (g = 2)

{

label1->Text = "Цикл for";

pictureBox1->Visible = false;

pictureBox2->Visible = false;

pictureBox3->Visible = true;

textBox1->Text = "У мові C++ цикл for може мати дуже широку реалізацію та застосування. Цикл for ще називається циклом з параметром. Цикл for виконується до тих пір, поки значення вираз рівне true.";

textBox1->Text += Environment::NewLine + "Як тільки значення вираз стане false, виконання циклу припиняється і виконується оператор, що слідує за циклом for. Загальна форма оператора циклу for:";

}

else

```
{
f = f + 1;
g = g + f;
}
}
f = f + 1;
g = g + f;
```

Перегляд вікна з довідковим матеріалом завершується натисканням на форму з ним. Це реалізовано з допомогою:

```
private: System::Void button3_Click(System::Object^ sender,
System::EventArgs^ e) {
MessageBox::Show("Теоретичний матеріал пройдений!");
MyForm1^ form = gcnew MyForm1();
form->Show();
this->Close();
}
```

Перехід на вікно тренінгу програмно реалізуемо за допомогою функції MessageBox , зокрема

```
MessageBox::Show("Теорія пройдена!");
MyForm1^ form = gcnew MyForm1();
form->Show();
this->Close();
```

Перевірка кількості правильних відповідей, реалізується з допомогою наступного блоку

```
int s = s;
if (radioButton1->Checked && radioButton7->Checked && radioButton12->Checked && radioButton13->Checked && radioButton20->Checked ) {
s == 5;
MessageBox::Show("Ви відповіли на п'ять питання!");
this->Close();
}
```

Як результат виводиться вікно з повідомленням, що вказує на кількість правильних відповідей (у даному випадку правильність відповідей дорівнює 5)

Перевірка кількості вірних варіантів відбувається з допомогою наступного блоку, в якому аналізується відповідь і вираховується відсоток вірних. Як результат роботи - виводить повідомлення про кількість вірних відповідей (у прикладі - правильно відповіли на 4 запитання.)

```
else
{
if (radioButton1->Checked && radioButton7->Checked && radioButton12->Checked && radioButton13->Checked || radioButton1->Checked && radioButton7->Checked && radioButton12->Checked && radioButton20->Checked || radioButton1->Checked && radioButton7->Checked && radioButton13->Checked && radioButton20->Checked || radioButton7->Checked && radioButton12->Checked && radioButton13->Checked && radioButton20->Checked || radioButton1->Checked && radioButton12->Checked && radioButton13->Checked && radioButton20->Checked)
{
s == 4;
```

```
MessageBox::Show("Ви відповіли вірно на чотири питання!");
this->Close();
}
```

Якщо кількість вірних відповідей більше або дорівняє 80%, відбувається перехід на інше вікно тренінгу (питання по іншому циклу), це реалізовано за допомогою наступної умови у коді програми:

```
int s = s;
if (radioButton1->Checked && radioButton7->Checked && radioButton12->Checked && radioButton13->Checked && radioButton20->Checked) {
s == 5;
radioButton1->BackColor = BackColor.Green;
radioButton7->BackColor = BackColor.Green;
radioButton12->BackColor = BackColor.Green;
radioButton13->BackColor = BackColor.Green;
radioButton20->BackColor = BackColor.Green;
radioButton2->BackColor = BackColor.Red;
radioButton3->BackColor = BackColor.Red;
radioButton4->BackColor = BackColor.Red;
radioButton5->BackColor = BackColor.Red;
radioButton6->BackColor = BackColor.Red;
radioButton8->BackColor = BackColor.Red;
radioButton9->BackColor = BackColor.Red;
radioButton10->BackColor = BackColor.Red;
```

```
radioButton11->BackColor = BackColor.Red;
radioButton14->BackColor = BackColor.Red;
radioButton15->BackColor = BackColor.Red;
radioButton16->BackColor = BackColor.Red;
radioButton17->BackColor = BackColor.Red;
radioButton18->BackColor = BackColor.Red;
radioButton19->BackColor = BackColor.Red;
MessageBox::Show("Ви відповіли на п'ять питання!");
MyForm5^ form = gcnew MyForm5();
form->Show();
this->Close();
```

Опис та підключення компонентів форми відбувається з допомогою наступного коду.

```
{
private: System::Windows::Forms::Button^ button2;
protected:
private: System::Windows::Forms::Button^ button1;
private: System::Windows::Forms::Label^ label2;
private: System::Windows::Forms::PictureBox^ pictureBox1;
public: System::Windows::Forms::TextBox^ textBox1;
private:
private: System::Windows::Forms::Label^ label1;
```

private: System::Windows::Forms::PictureBox^ pictureBox2;

private: System::Windows::Forms::Button^ button3;

private: System::Windows::Forms::Button^ button4;

}

Кожен компонент також повинен бути підключений та описаний індивідуально. Для більш успішного запуску кожен компонент повинен мати своє унікальне ім'я, що не повторюється до цього.

Було створено 4 методи:

1. Метод кнопки.

private: System::Windows::Forms::Button^ button1;.

2. Метод label.

private: System::Windows::Forms::Label^ label2;.

3. Відкритий метод текстового вікна.

public: System::Windows::Forms::TextBox^ textBox1;.

4. Метод картинки.

private: System::Windows::Forms::PictureBox^ pictureBox2;.

void InitializeComponent(void)

{

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(MyForm::typeid));

```
this->button2 = (gcnew System::Windows::Forms::Button());
this->button1 = (gcnew System::Windows::Forms::Button());
this->label2 = (gcnew System::Windows::Forms::Label());
this->pictureBox1 = (gcnew System::Windows::Forms::PictureBox());
this->textBox1 = (gcnew System::Windows::Forms::TextBox());
this->label1 = (gcnew System::Windows::Forms::Label());
this->pictureBox2 = (gcnew System::Windows::Forms::PictureBox());
this->button3 = (gcnew System::Windows::Forms::Button());
this->button4 = (gcnew System::Windows::Forms::Button());
(cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox1))->BeginInit();
(cli::safe_cast<System::ComponentModel::ISupportInitialize^>(this->pictureBox2))->BeginInit();
this->SuspendLayout();
//
// button2
//
this->button2->BackColor = System::Drawing::SystemColors::HotTrack;
this->button2->ForeColor = System::Drawing::SystemColors::ButtonFace;
this->button2->Location = System::Drawing::Point(225, 450);
this->button2->Name = L"button2";
this->button2->Size = System::Drawing::Size(133, 53);
this->button2->TabIndex = 11;
```

```
        this->button2->Text = L"Повернутися";
        this->button2->UseVisualStyleBackColor = false;
        this->button2->Click += gcnew System::EventHandler(this,
&MyForm::button2_Click);
        //
        // button1
        //
        this->button1->BackColor = System::Drawing::SystemColors::HotTrack;
        this->button1->ForeColor = System::Drawing::SystemColors::ButtonFace;
        this->button1->Location = System::Drawing::Point(428, 450);
        this->button1->Name = L"button1";
        this->button1->Size = System::Drawing::Size(116, 53);
        this->button1->TabIndex = 10;
        this->button1->Text = L"Продовжити";
        this->button1->UseVisualStyleBackColor = false;
        this->button1->Click += gcnew System::EventHandler(this,
&MyForm::button1_Click);
        //
        // label2
        //
        this->label2->AutoSize = true;
        this->label2->Location = System::Drawing::Point(237, 322);
        this->label2->Name = L"label2";
```

```
        this->label2->Size = System::Drawing::Size(0, 13);
        this->label2->TabIndex = 9;
        //
        // pictureBox1
        //
        this->pictureBox1->Image =
(cli::safe_cast<System::Drawing::Image^>(resources-
>GetObject(L"pictureBox1.Image")));
        this->pictureBox1->Location = System::Drawing::Point(-1, 231);
        this->pictureBox1->Name = L"pictureBox1";
        this->pictureBox1->Size = System::Drawing::Size(350, 92);
        this->pictureBox1->TabIndex = 8;
        this->pictureBox1->TabStop = false;
        //
        // textBox1
        //
        this->textBox1->Enabled = false;
        this->textBox1->Font = (gcnew System::Drawing::Font(L"Times New Roman",
12.25F));
        this->textBox1->Location = System::Drawing::Point(20, 61);
        this->textBox1->Multiline = true;
        this->textBox1->Name = L"textBox1";
        this->textBox1->Size = System::Drawing::Size(728, 164);
```

```
this->textBox1->TabIndex = 7;
}
```

Якщо тренінг не пройдено, користувач повернутися назад до початкового меню. Це реалізовано з допомогою .

```
private: System::Void button4_Click(System::Object^ sender,
System::EventArgs^ e) {
MessageBox::Show("Повернення до головного меню!");
this->Close();
}
```

Вихід з із тренажера відбувається з допомогою кнопки "Вихід", програма перед закриттям видає повідомлення: "Успіхів у навчанні! Програма закривається." Це блок реалізовано з допомогою функції

```
MessageBox::Show("Успіхів у навчанні! Програма закривається!");
        this->Close();
```

4.2 Опис роботи програми

При відкритті тренажеру з’являється початкове вікно програми, де він може побачити назву тренажеру, кнопки “Довідковий матеріал”, “Тестування”, “Інформація про тренажер”, “Вихід”, рис. (4.3).

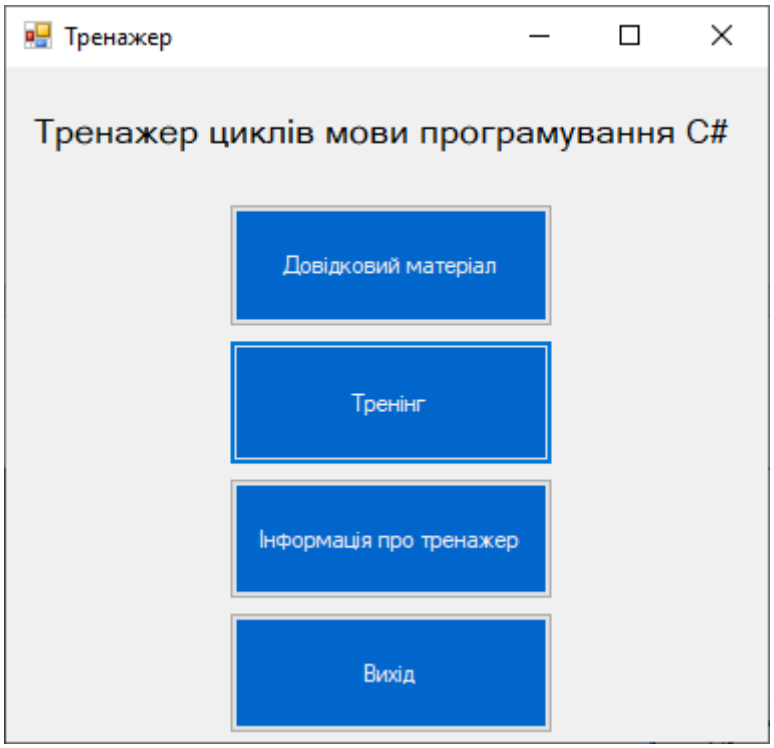


Рисунок 4.3 – На скриншоті зображено головне меню програми

Перед початком тренінгу , користувач може переглянути теоретичний матеріал за темою з допомогою кнопки “Довідковий матеріал”, після натискання відкриється вікно де знаходиться інформація про цикл мови C# спочатку по циклу while, рис. (4.4).

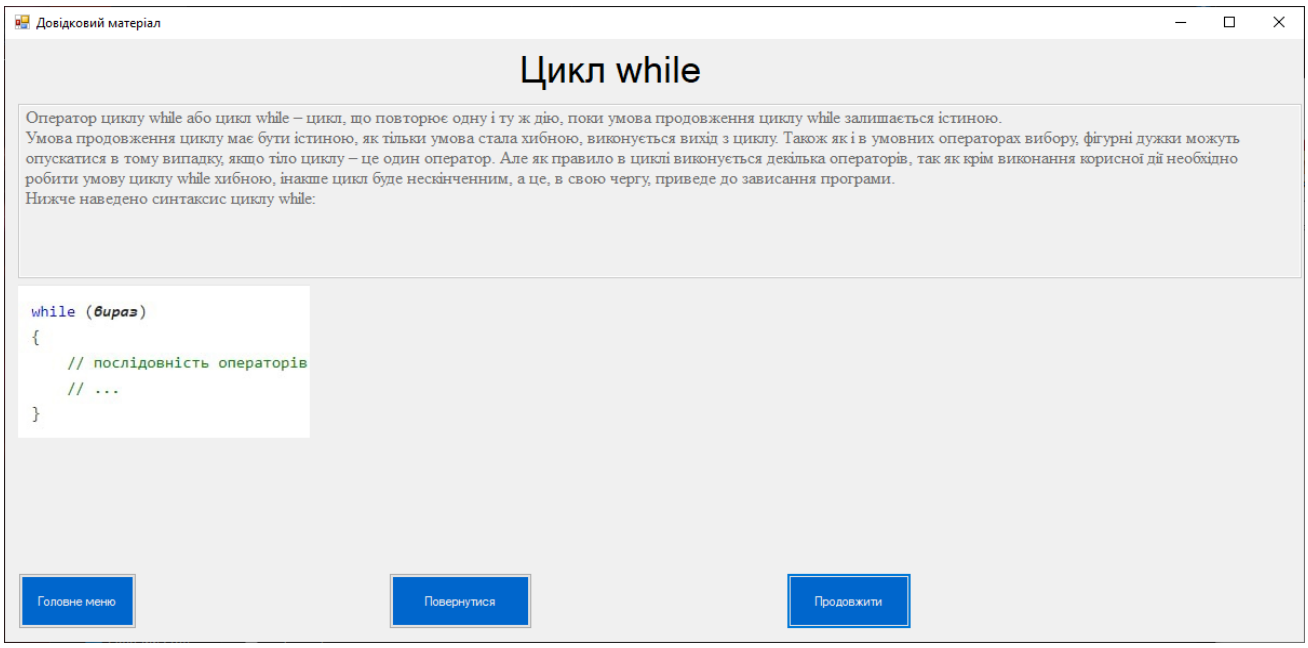


Рисунок 4.4 – В даному вікні бачимо інформацію про цикл while

Щоб перейти до наступних вікон з довідковою інформацією потрібно натиснути кнопку “Продовжити”, після чого відбудеться перехід до наступного вікна рис. (4.5-4.7).

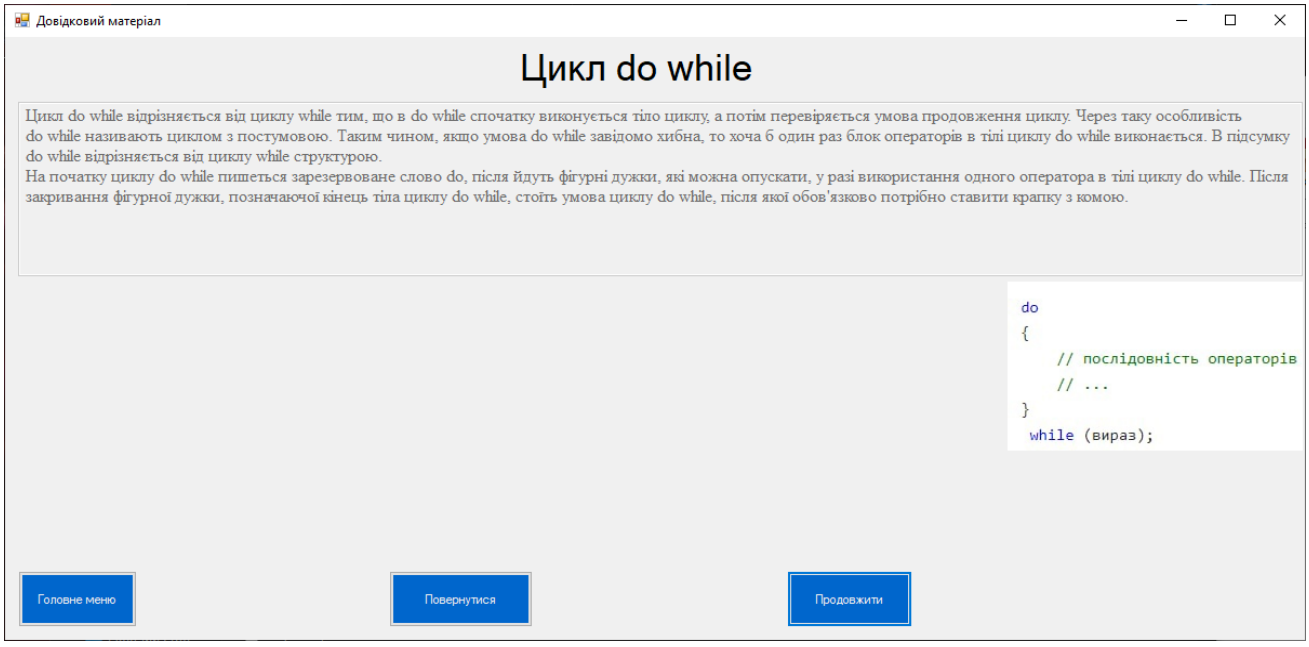


Рисунок 4.5 – Вікно з інформацією про цикл do while

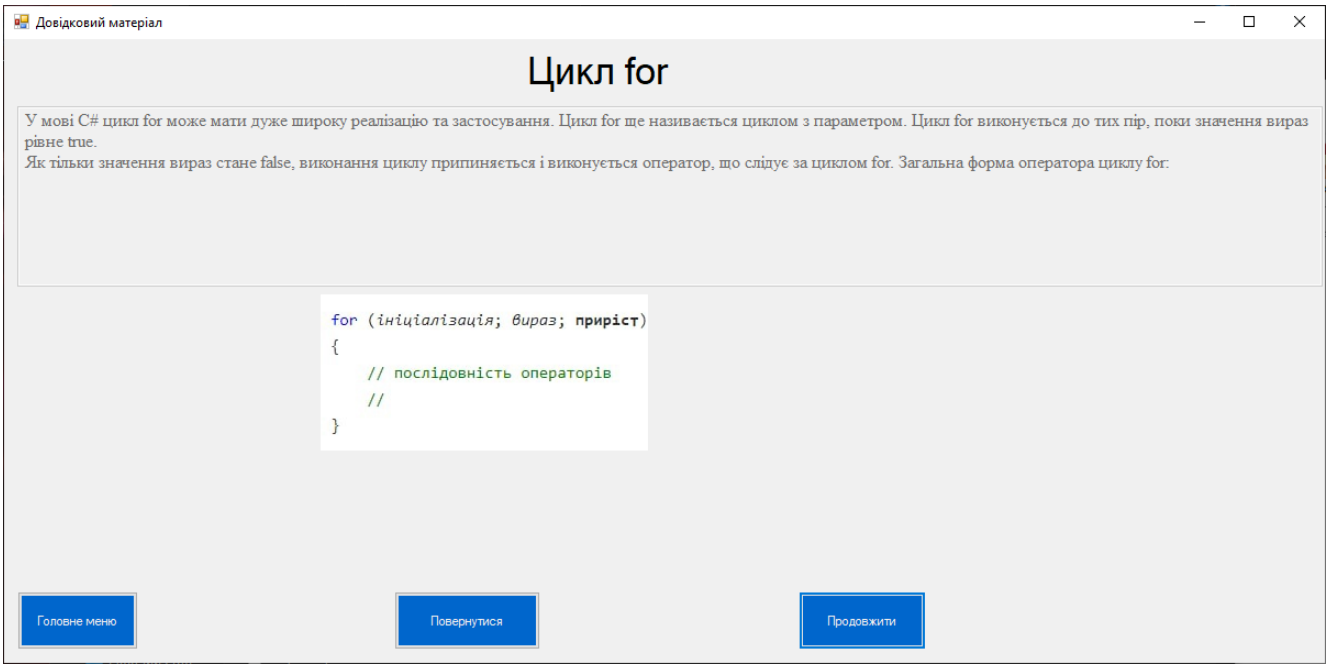


Рисунок 4.6 – Вікно з інформацією про цикл for

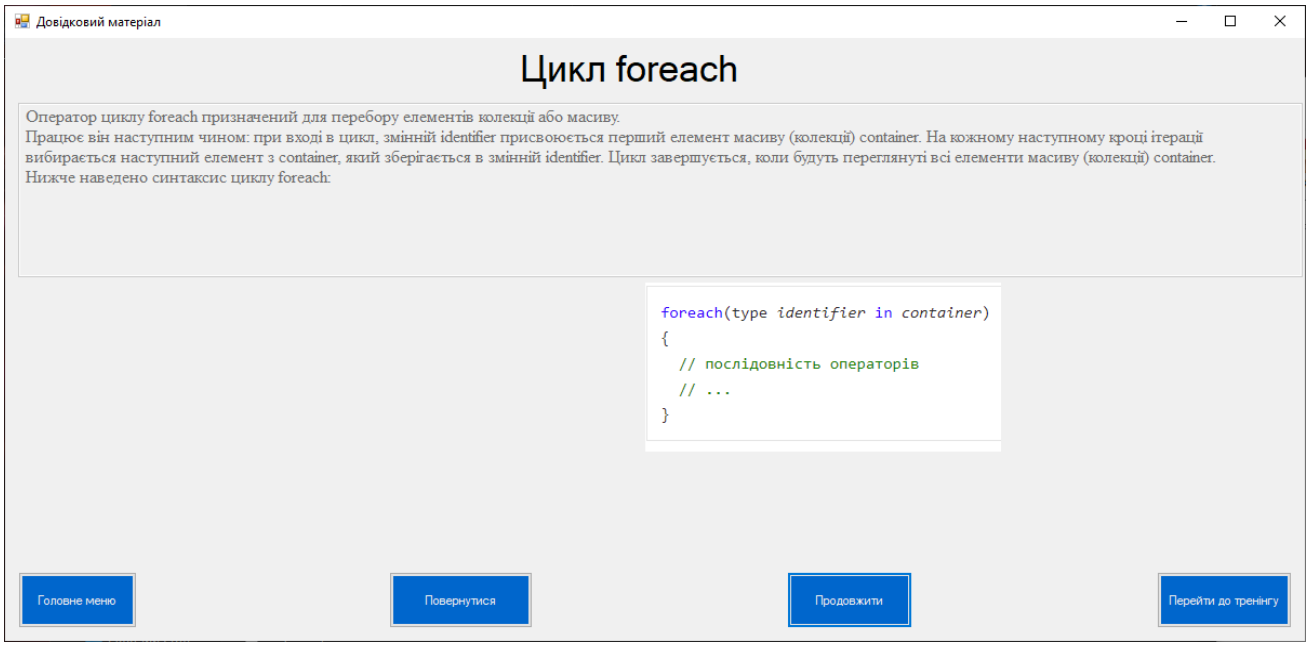


Рисунок 4.7 – Послідніє вікно в довідниковому матеріалі з інформацією про цикл foreach

Після перегляду довідкового матеріалу переходимо до тренінгу, натискаючи на кнопку “Перейти до тренінгу”, після чого з’явиться вікно з блоком питань по циклу while, рис. (4.8).

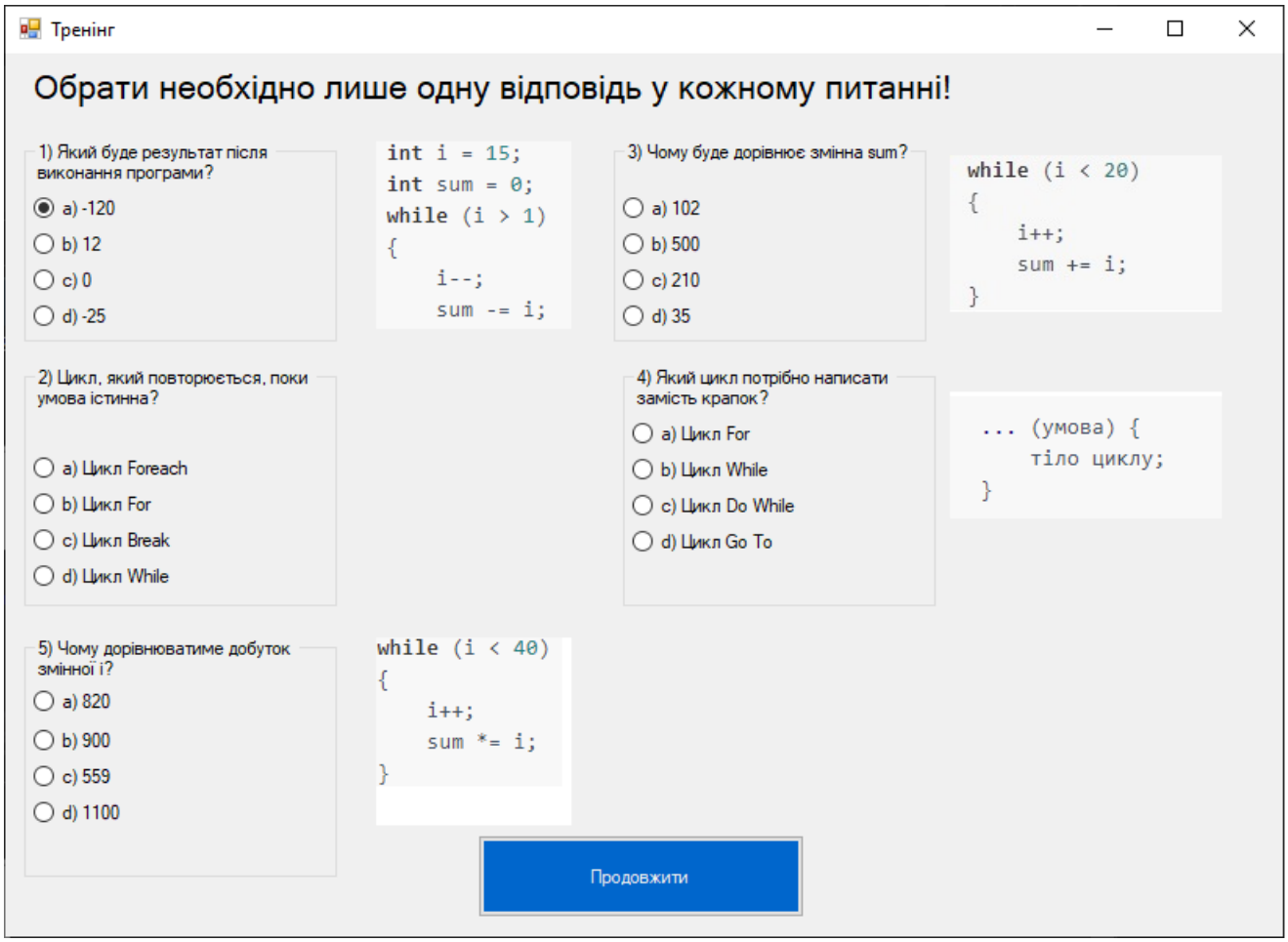


Рисунок 4.8 – Тренінг по циклу while

Вибравши одну із відповідей під кожним питанням і натиснувши кнопку “Продовжити” з’явиться вікно з інформацією про кількість правильних відповідей. Щоб перейти до наступного вікна потрібно правильно надати відповідь мінімум на чотири питання, рис. (4.9-4.10).

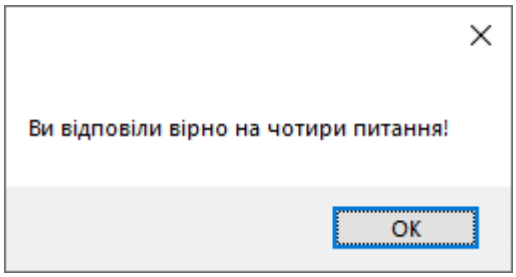


Рисунок 4.9 – Відображення повідомлення про кількість вірних відповідей

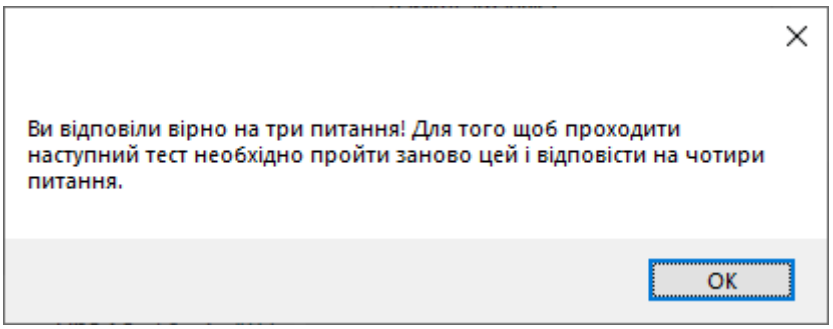


Рисунок 4.10 – Повідомлення про недостатню кількість вірних відповідей для переходу до наступного блоку питань

В цей же час у вікні питань правильні відповіді зафарбуються зеленим кольором, а неправильні червоним, рис. (4.11).

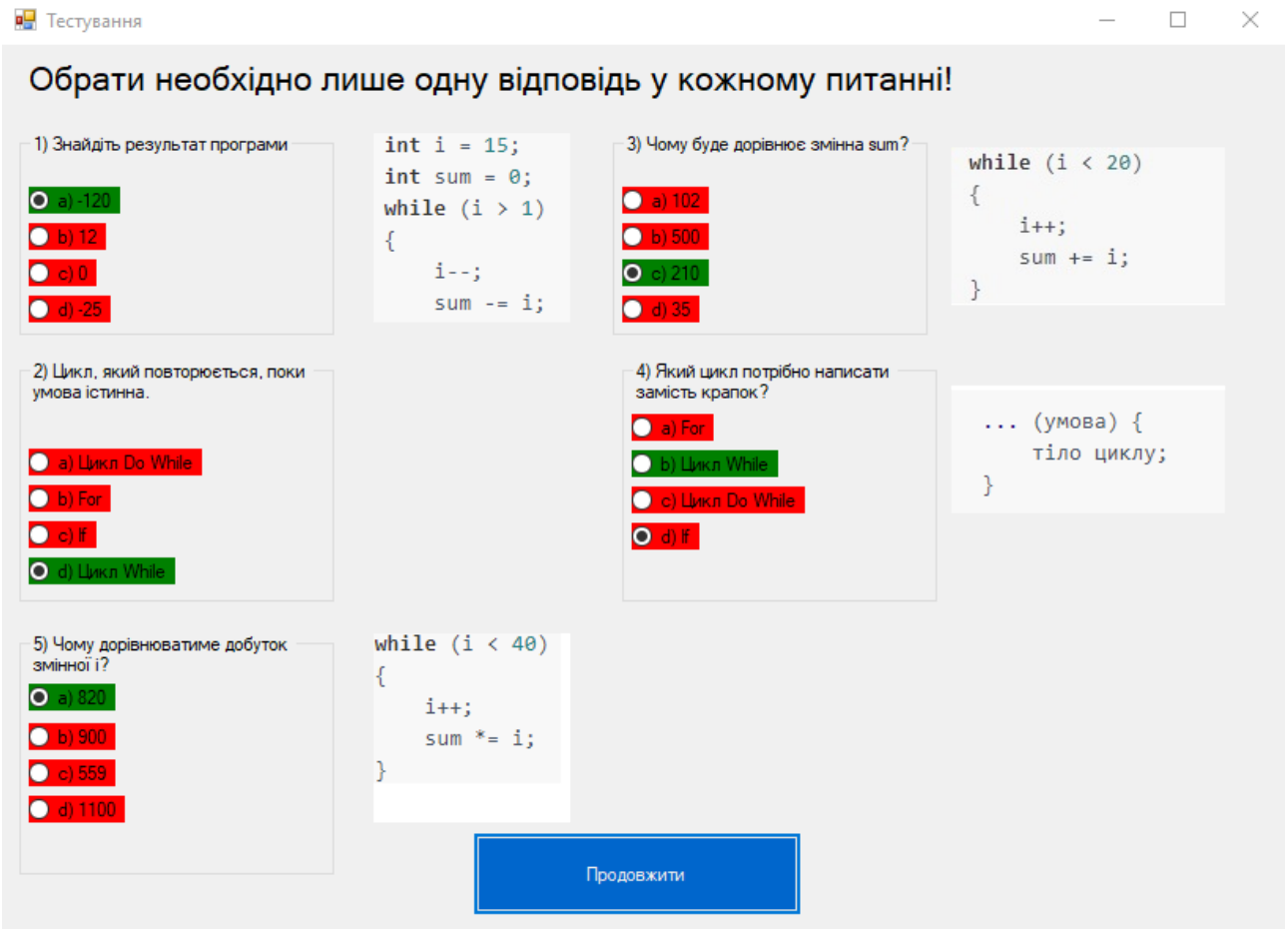


Рисунок 4.11 – Відображення правильних і неправильних відповідей кольором

Питання з інших циклів мови програмування C#, рис. (4.12-4.14).

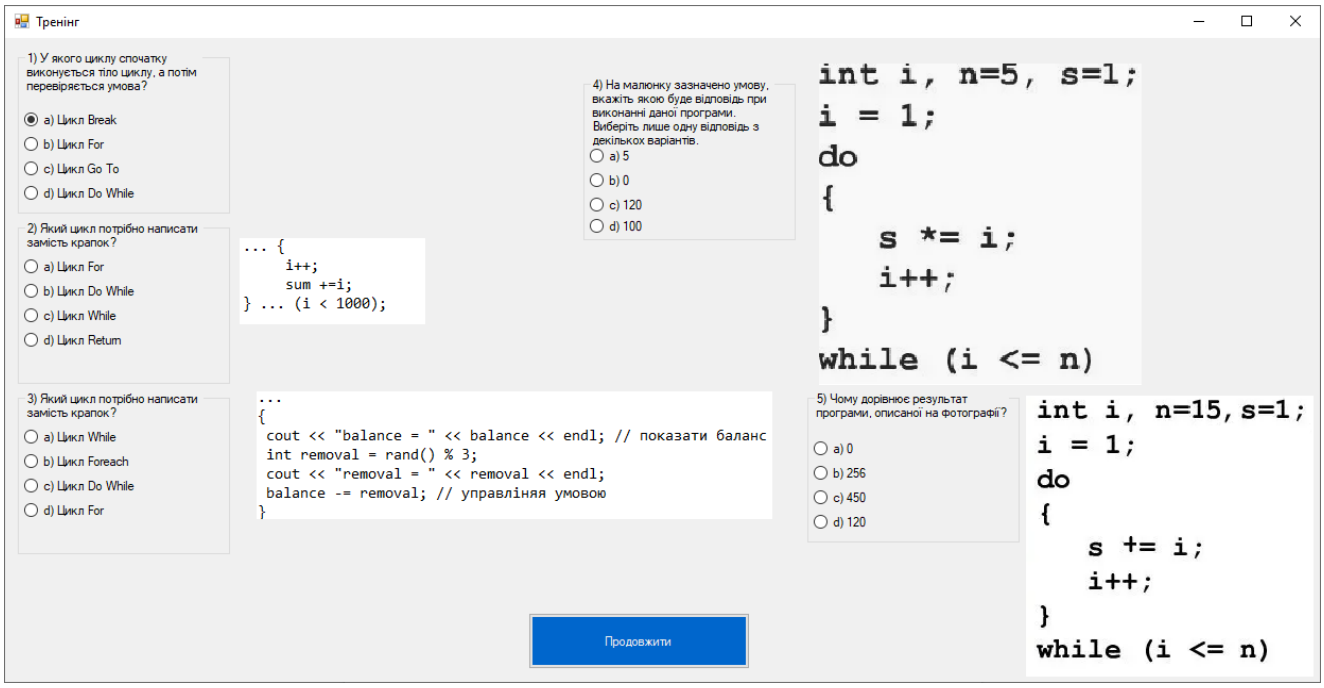


Рисунок 4.12 – Тренінг по циклу do while

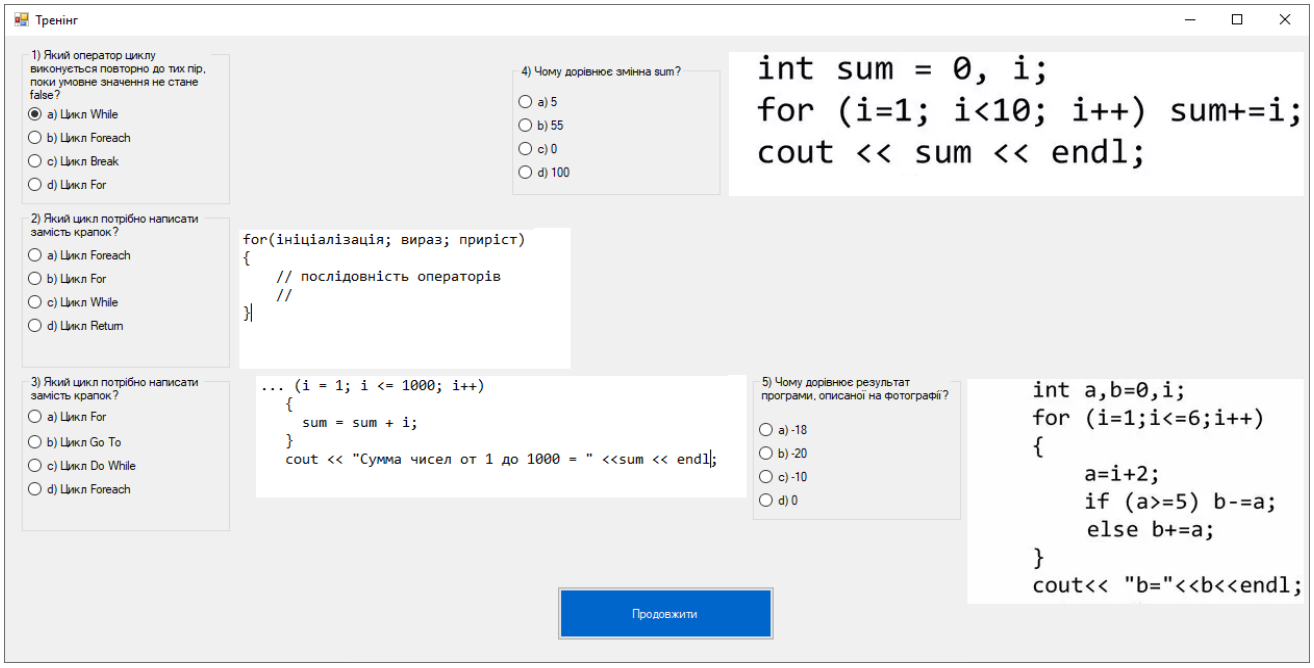


Рисунок 4.13 – Тренінг по циклу for

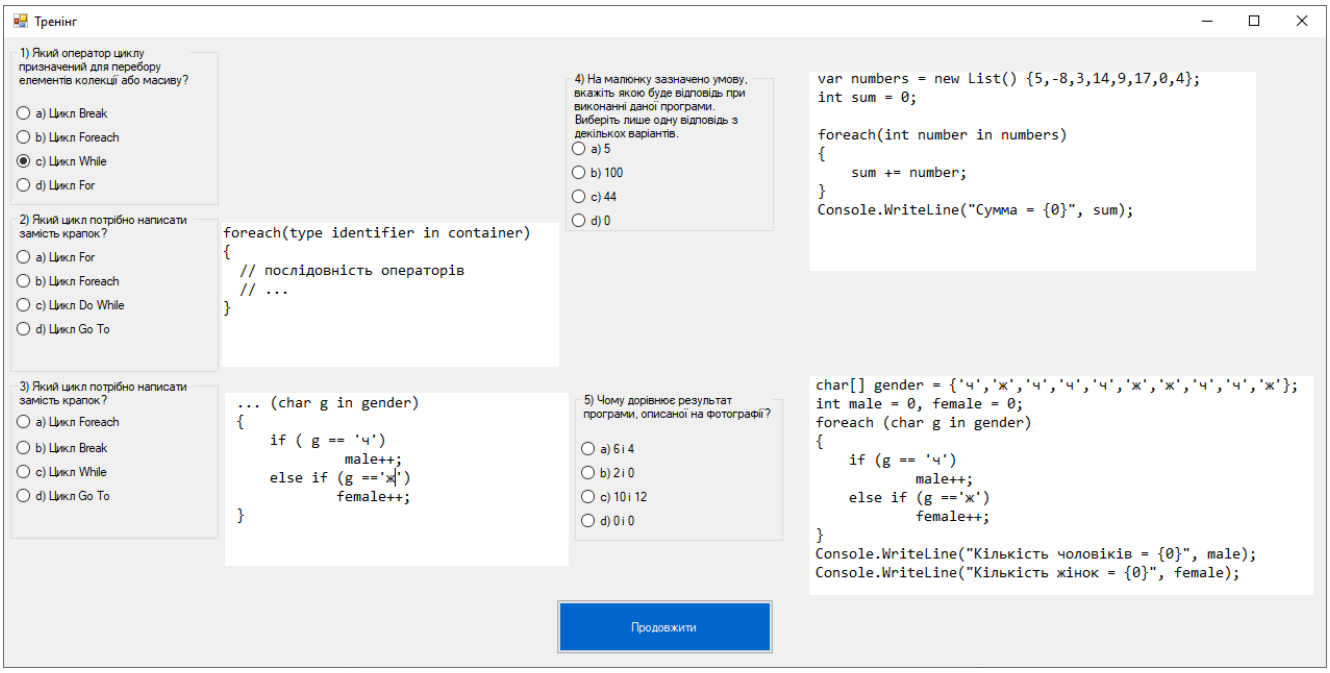


Рисунок 4.14 – Тренінг по циклу foreach

Якщо студент відповів вірно на 80% усіх питань по кожній частині тренінгу з’являється вікно, в якому буде повідомлення про рис. (4.14), натиснувши на кнопку “ОК”, , відбудеться повернення до головного меню тренажеру.

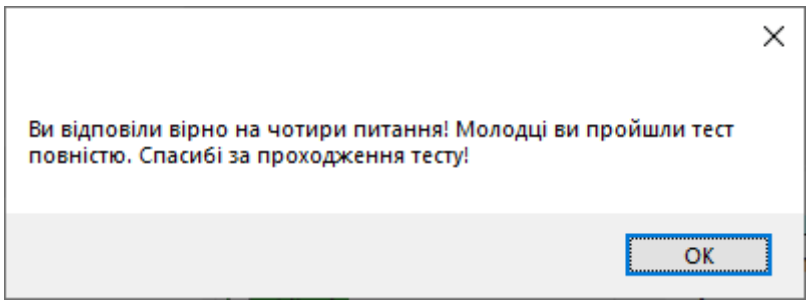


Рисунок 4.15 – Повідомлення про успішне завершення тренінгу

Натиснувши кнопку “Інформація про тренажер”, користувач може ознайомитись з інформацією, коли було створено тренажер, його версією і іменем автора. Натиснувши кнопку “Повернутися” програма повернеться до головного меню, рис. (4.16).

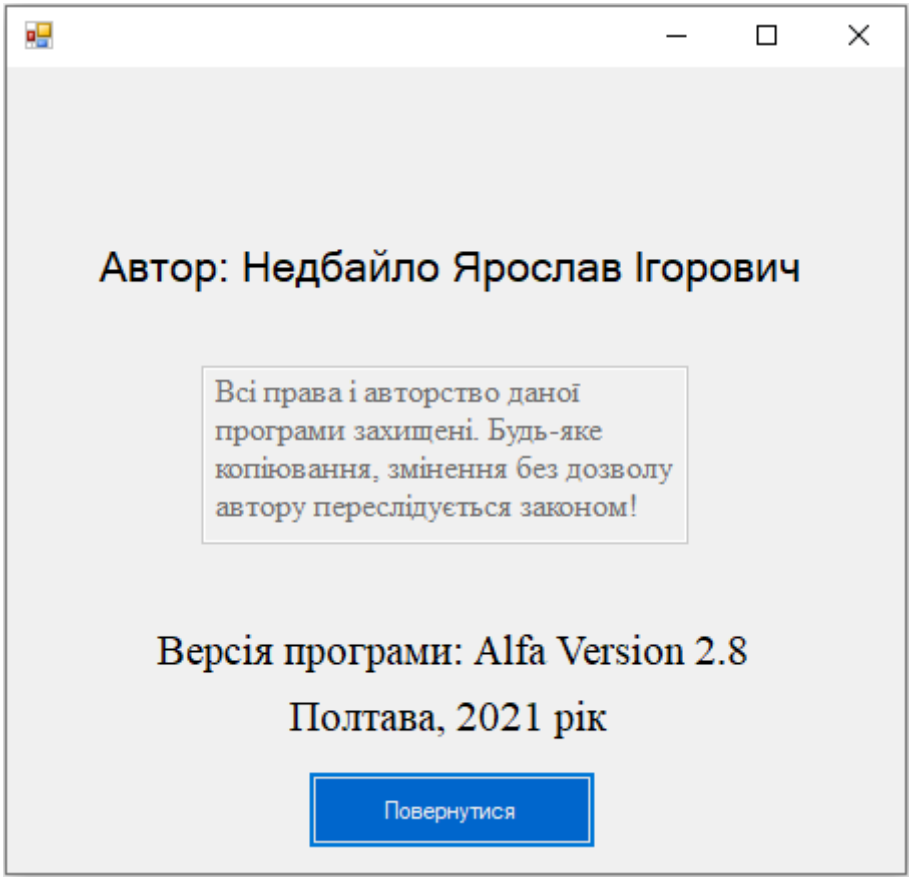


Рисунок 4.16 – Вигляд вікна пункту меню “Інформація про тренажер”

ВИСНОВКИ

При виконанні магістерської роботи було розроблено тренажер для проходження тестів.

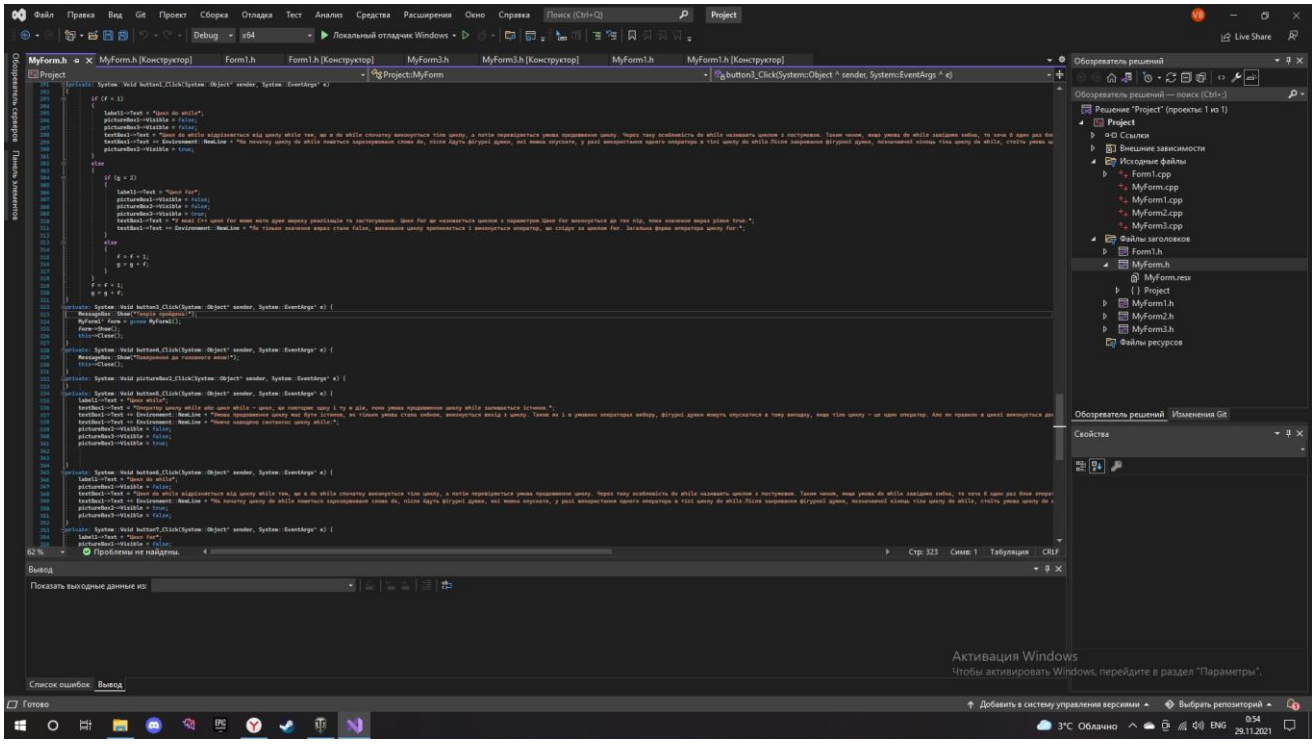
Виконані основні умови для створення тренажеру, а саме:

- розглянуто цикли у мові програмування C#;
- розглянуто теоретичні відомості про цикли;
- розроблено алгоритм тренажера, що дозволить закріпити знання з теми «Цикли в C#»;
- обрано програмне забезпечення;
- описано процес програмної реалізації;
- описано роботу тренажера.

Тренажер дотримав усіх вимог щодо розробки та роботи і це показує його надійність та ефективність у роботі.

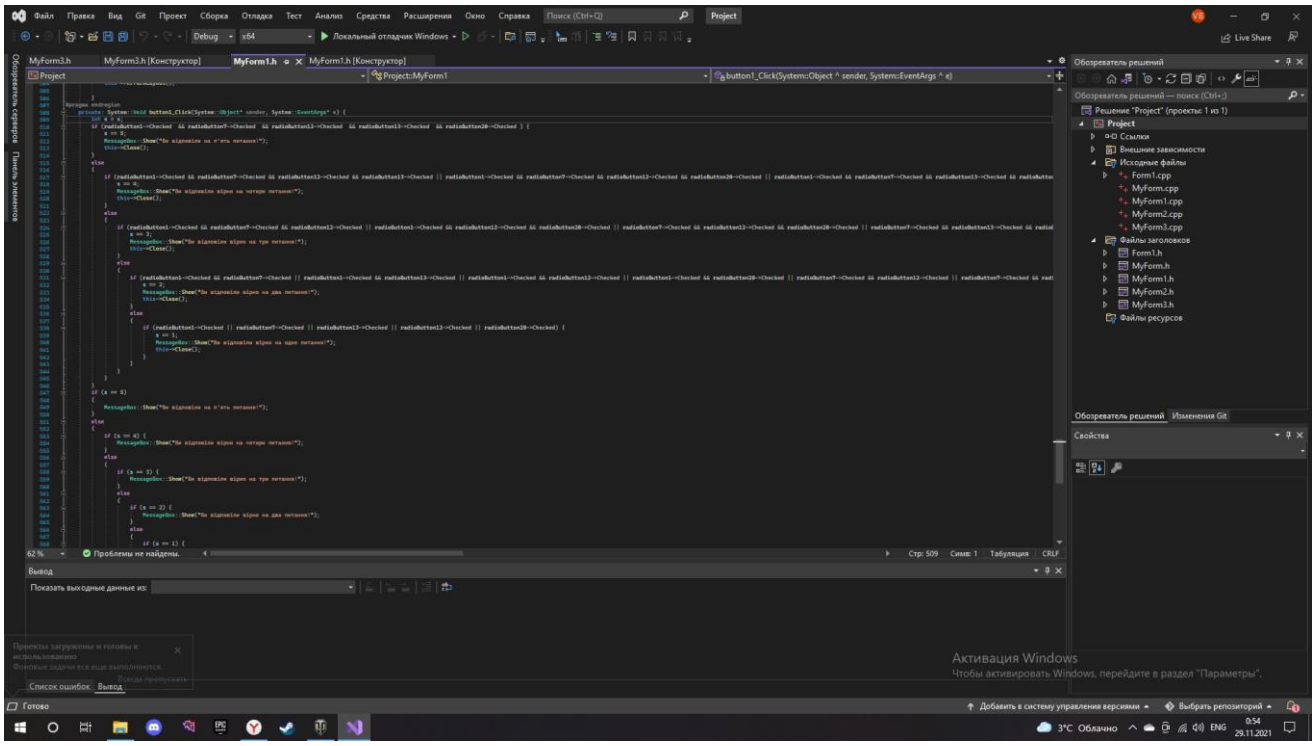
ДОДАТКИ

Довідковий матеріал



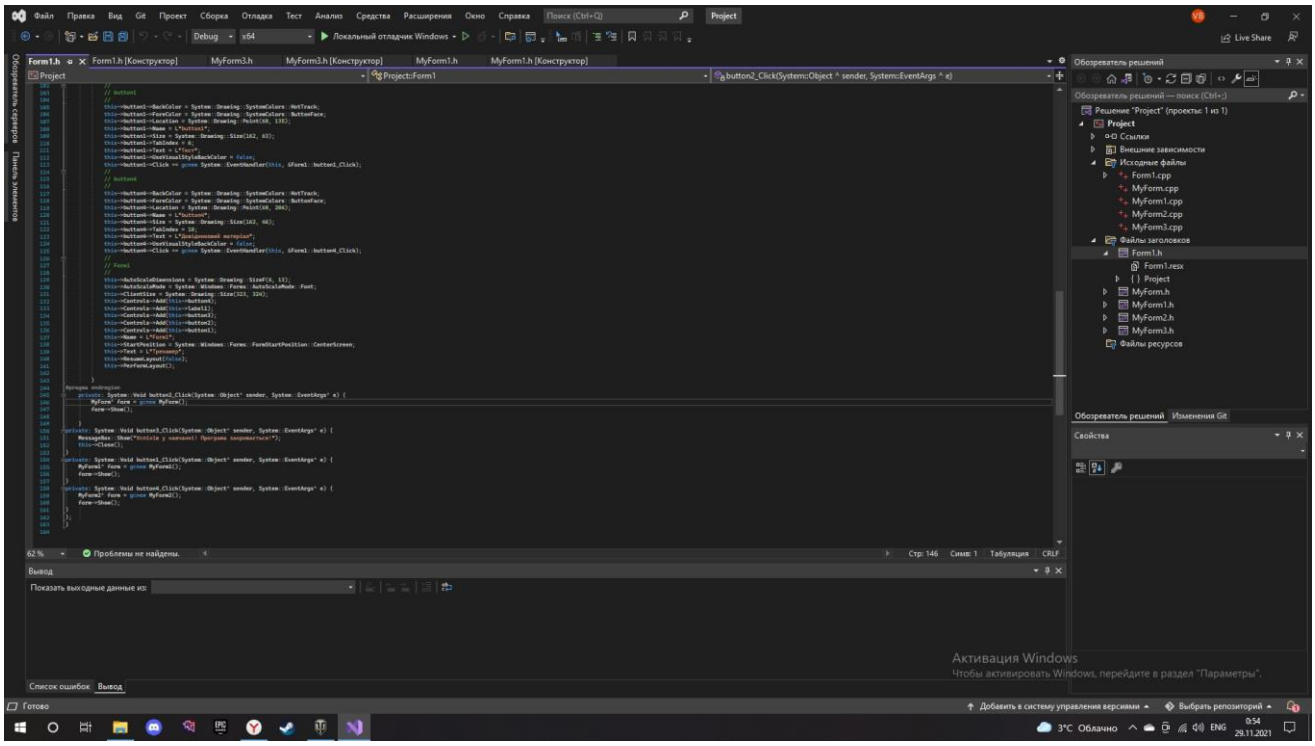
На цьому скриншоті коду, взятого з програми ми бачимо декілька функцій описуючих роботу кнопок. Наприклад при натисканні на кнопку з назвою циклу “do while”, змінюється опис даного циклу циклу, і його фото.

Тренінг



На наступному скриншоті ми бачимо реалізацію виводу вікна з результатом кількості правильних відповідей наданих в тренінгу.

Завершення роботи тренажеру



На останньому скриншоті ми бачимо функції, описуючі роботу головного вікна програми. При натисканні на кнопку “Вихід” з’являється вікно з повідомленням про завершення роботи програми і побажання від автора. Також є три кнопки для переходу по різним вікнам які показують різну інформацію про цикли.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. C# .Net: Цикли [Електронний ресурс]. – Режим доступу: https://programm.top/uk/c-sharp/tutorial/loops/
2. Операторы итераций (справочник по C#) [Електронний ресурс]. – Режим доступу: https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/statements/iteration-statements
3. Оператори C#. Циклічні структури [Електронний ресурс]. – Режим доступу: https://dl.sumdu.edu.ua/textbooks/108989/459185/index.html
4. Яку мову програмування краще вивчати першою [Електронний ресурс]. – Режим доступу: https://issoft.com.ua/blog/yaku-movu-programuvannya-krashhe-vivchati/
5. Порівняння мов програмування [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Порівняння_мов_програмування
6. Чому варто вивчати мову програмування C# [Електронний ресурс]. – Режим доступу: https://www.quality-assurance-group.com/chomu-varto-vyvchaty-movu-programuvannya-c-c-sharp/
7. C# — Преимущества и недостатки [Електронний ресурс]. – Режим доступу: https://shwanoff.ru/plus-minus-c-sharp/
8. C Sharp [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/C_Sharp
9. Для чего нужен язык C# [Електронний ресурс]. – Режим доступу: https://thecode.media/c-sharp/
10. 10 фактов в пользу C# [Електронний ресурс]. – Режим доступу: https://itvdn.com/ru/blog/article/10facts-csharp18
11. Есть ли смысл изучать C# в 2021 году: карьерные перспективы новичка [Електронний ресурс]. – Режим доступу: https://proglib.io/p/est-li-smysl-izuchat-c-v-2021-godu-karernye-perspektivy-novichka-2021-07-29
12. С днем программиста, или 10 фактов о C++++ [Електронний ресурс]. – Режим доступу: https://habr.com/ru/company/microsoft/blog/309774/