ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ БІЗНЕСУ ТА СУЧАСНИХ ТЕХНОЛОГІЙ

ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ ІНФОРМАТИКИ

Допускається до захисту

Завідувач кафедри _____О.О. Ємець
(підпис)

«_____»_____2021 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО БАКАЛАВРСЬКОЇ РОБОТИ

на тему
ТРЕНАЖЕР З ТЕМИ «ПОБУДОВА БЛОК-СХЕМ АЛГОРИТМІВ ЦИКЛІЧНОЇ СТРУКТУРИ НА ПРИКЛАДІ ЦИКЛУ FOR» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ПРОГРАМУВАННЯ ІІ» ТА РОЗРОБКА ЙОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

зі спеціальності 122 «Комп'ютерні науки»

Виконавець роботи Крамаренко Олександр Володимирович

_____«__»_____2021р.
(підпис)

Науковий керівник канд. фіз.-мат. наук, Ємець Олександра Олегівна

_____ «__»_____2021р.
(підпис)

ПОЛТАВА 2021 р.

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ЗАТВЕРДЖУЮ

Завідувач кафедри_____О.О. Ємець
(підпис)

« _» _____20___р.

ЗАВДАННЯ ТА КАЛЕНДАРНИЙ ГРАФІК
ВИКОНАННЯ БАКАЛАВРСЬКОЇ РОБОТИ

Студент(ка) спеціальності 122 «Комп'ютерні науки»

Прізвище, ім'я, по батькові Крамаренко Олександр Володимирович

1. Тема **«Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення затверджена наказом ректора № 121-Н від « 1 » вересня 2020 р.**

Термін подання студентом бакалаврської роботи « 24 » травня 2021 р.

2. Вихідні дані до дипломної роботи: публікації з теми навчальні тренажери в дистанційних курсах з комп'ютерних наук.

Завдання розробки - програмна реалізація тренажеру для навчання на тему «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення»

3. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Вступ
1. ПОСТАНОВКА ЗАДАЧІ
1.1. Постановка задачі розробки тренажера.
2. ІНФОРМАЦІЙНИЙ ОГЛЯД
2.1. Огляд робіт зі схожим завданням.
2.2. Позитивні аспекти оглянутих робіт.
2.3. Вади розробок оглянутих робіт.
2.4. Актуальність розробки програмного забезпечення.
3. ТЕОРЕТИЧНА ЧАСТИНА
3.1. Загальні відомості.

3.2. Алгоритм роботи тренажера.

3.3. Блок-схема програми-тренажера.

4. ПРАКТИЧНА ЧАСТИНА

4.1. Опис програмного забезпечення.

Висновки

4. Перелік графічного матеріалу: 3-4 аркуші блок-схем, інші необхідні ілюстрації.

5. Консультанти розділів бакалаврської роботи

Розділ	Прізвище, ініціали, посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1. Постанова задачі	Ємець О. О.	05.09.20	05.09.20
2. Інформаційний огляд	Ємець О. О.	05.09.20	05.09.20
3. Теоретична частина	Ємець О. О.	05.09.20	05.09.20
4. Практична реалізація	Ємець О. О.	05.09.20	05.09.20

6. Календарний графік виконання бакалаврської роботи

Зміст роботи	Термін виконання	Фактичне виконання
1. Вступ	10.05.21	
2. Вивчення методичних рекомендацій та стандартів та звіт керівнику	1.10.20	
3. Постановка задачі	1.10.20	
4. Інформаційний огляд джерел бібліотек та інтернету	1.11.20	
5. Теоретична частина	1.02.21	
6. Практична частина	15.05.21	
7. Закінчення оформлення	20.05.21	
8. Доповідь студента на кафедрі	30.05.21	
9. Доробка (за необхідністю), рецензування	13.06.21	

Дата видачі завдання « 5 » вересня 2020 р.

Студент _____
(підпис)

Науковий керівник _____ канд. фіз.-мат. наук, Ємець Олександра Олегівна
(підпис)

Результати захисту бакалаврської роботи

Дипломна робота оцінена на _____
(балів, оцінка за національною шкалою, оцінка за ECTS)

Протокол засідання ЕК № ____ від «____» _____ 2021 р.

Секретар ЕК_____ _____
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Записка: 48 с., основна частина 42 с., джерел - 15.

Предмет розробки – програмна реалізація тренажеру з теми «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення».

Мета роботи – розробка програмного забезпечення з теми «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення».

Методи, які були використані для розв'язування задачі – тренажер розроблено в середовищі Microsoft Visual Studio з використанням Unity та мови програмування C#.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень, термінів

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, термінів
Алгоритм	Послідовність дій для розв’язування задачі
Блок-схема	алгоритм програми у вигляді геометричних фігур
Цикл	конструкція в програмуванні для багаторазового повторення

ВСТУП

Дистанційне, або віртуальне навчання майже нічим не відрізняється від традиційного. За допомогою передових технологій таке навчання в деякому сенсі навіть краще та зручніше за звичайне. Звичайно, технології не замінять звичного «очного» навчання, але дистанційне навчання може стати поштовхом, наприклад, для вивчення мови програмування С++. В наш час не обов'язково мати вдома бібліотеку підручників та наукових статей – достатньо мати доступ до мережі Інтернет. Віртуальні класи, вебінари чи онлайн-підручники далеко не весь перелік можливостей дистанційного навчання. Під час роботи з бакалаврською роботою було розглянуто такий вид дистанційного навчання як програми-вчителі, або, як їх ще називають – тренажери.

Основні функції подібних програм:

1) Самостійне навчання без допомоги викладача;
2) навчання у будь-який час та у будь-якому місці;
3) можливість навчання без постійного доступу в інтернет;
4) покращення навичок завдяки теоретичним та практичним матеріалам.

Мета роботи: розробка програмного забезпечення з теми «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення».

Об'єкт роботи: програмна реалізація тренажеру.

Предмет роботи: програмна реалізація тренажеру з теми «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення».

Методи роботи: написання програмного забезпечення відбувалося за допомогою середовища розробки Microsoft Visual Studio з використанням середовища розробки інтерактивних програм Unity та мови програмування C#.

Структура пояснювальної записки до бакалаврської роботи:

- титульний аркуш;

- завдання;

- реферат, що містить предмет, мету, методи, анотацію результатів, ключові слова, словосполучення;

- зміст;

- вступ;

- основна частина;

- висновки;

- рекомендації;

- список використаних джерел;

Обсяг пояснювальної записки: 48 стор., в т.ч. основна частина 42 стор., джерел - 15.

1 Постановка задачі

Основна задача бакалаврської роботи – розробка програмного забезпечення з теми «Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ» та розробка його програмного забезпечення.».

Задля нормальної роботи в системах дистанційного навчання описано наступні вимоги до програмного забезпечення:

1) Послідовна видача теоретичної інформації та практичних завдань;
2) покрокове пояснення побудови блок-схем;
3) розгляд побудови блок-схеми на певному прикладі.

Основні вимоги до бакалаврської роботи:

1) Пошук та оформлення теоретичної частини бакалаврської роботи;
2) складання алгоритму роботи програмного забезпечення;
3) складання блок-схеми алгоритму;
4) створення елементів програмного забезпечення у вигляді тренажеру.

2 Інформаційний огляд

2.1 Огляд робіт зі схожим завданням

Після проведення пошуку схожих робіт було обрано наступні:

1. Пояснювальна записка до дипломної на тему: Розробка програмного забезпечення тренажеру з теми: «Дефазифікація нечітких множин» дистанційного навчального курсу «Сучасні методи оптимізації та їх програмування»» [1]

Автори: Антоненко А. А.

2. Пояснювальна записка до бакалаврської роботи на тему: Розробка програмного забезпечення тренажеру з теми: «Злиття впорядкованих послідовностей» дистанційного навчального курсу «Алгоритми та структури даних» [2]

Автори: Кулинич Марина Костянтинівна

3. Пояснювальна записка до дипломної на тему: Розробка програмного забезпечення тренажеру з теми: «Побудова математичних моделей лінійних задач планування виробництва» дистанційного навчального курсу «Сучасні методи оптимізації та їх програмування» [3]

Автори: Григор’єв Владислав Віталійович

4. Пояснительная записка к бакалаврской работе на тему: Тренажер по теме: «Нахождение коэффициентов компетентности экспертов», Мамедов, Ахмед Али оглы» [4]

Автори: Мамедов Ахмед Али оглы

2.2 Позитивні аспекти розглянутих робіт

Перевагами першого обраного тренажеру, автор Антоненко, А. А. є:

1. Існують тренінги для п’яти різних методів дефазифікації;
2. Є перевірка введеної відповіді з відповідним повідомленням;
3. Почати тренінг можна з будь-якого прикладу та з будь-якої частини тренажеру;
4. Різний колір для кожного прикладу для кращого сприйняття.

Перевагами другого обраного тренажеру, автор Кулинич Марина Костянтинівна є:

1. Є повідомлення з підказкою після вибору неправильної відповіді.
2. Є перевірка введеної відповіді з відповідним повідомленням.
3. Є можливість перетягувати інформацію у поле з відповіддю.

Перевагами третього обраного тренажеру, автор Григор'єв Владислав Віталійович є:

1. При роботі з практичною частиною умова постійно перед очима.
2. Є перевірка введеної відповіді з відповідним повідомленням.

Перевагами четвертого обраного тренажеру, автор Мамедов Ахмед Али оглы є:

1. Є перевірка введеної відповіді з відповідним повідомленням.
2. Приємний дизайн.

2.3 Вади розглянутих робіт

Мінусами першого обраного тренажеру, автор Антоненко А. А. є:

1. Занадто яскравий дизайн;
2. Не має автоматичного виправлення після декількох помилок.

Мінусами другого обраного тренажеру, автор Кулинич Марина Костянтинівна є:

1. Не має можливості повторити роботу з тренажером не перезавантажуючи його.

Мінусами третього обраного тренажеру, автор Григор'єв Владислав Віталійович є:

1. Малий розмір вікна, погано працює на маленьких екранах.
2. Не має можливості повторити роботу з тренажером не перезавантажуючи його.

Мінусами четвертого обраного тренажеру, автор Мамедов Ахмед Али оглы є:

1. Не має можливості повторити роботу з тренажером не перезавантажуючи його.
2. Малий розмір вікна тренажеру.

2.4 Актуальність розробки програмного забезпечення

Виходячи з даних отриманих при дослідженні тренажерів зі схожим завданням було зроблено наступні висновки:

- В розробленому програмному забезпеченні необхідно додати можливість повтору роботи з тренажером;
- для усунення нагромадження інформації на одному вікні додати можливість показу прогресу на поточному кроці на окремому вікні;
- додати можливість бачити покрову побудову блок-схеми, що краще вплине на сприйняття та запам’ятовування.

3 Теоретична частина

3.1 Загальні відомості

Програмування – складна річ, яка вимагає чіткого плану дій. Будь-яке програмне забезпечення працює по заданому алгоритму, тому під час вивчення програмування необхідно чітко уявляти, що таке алгоритм, як він працює, як його створити та запрограмувати. Саме від спроектованого алгоритму залежить скільки часу у вас займе програмна реалізація програми, наскільки це буде складно та наскільки багато часу піде на пошук помилок. Інтуїтивно зрозумілий та простий алгоритм – першочергова задача програміста. Також алгоритм показує структуру виконання програми або частини коду, що важливо при подальшій модернізації програми.

Алгоритми, у яких використовується тільки структура «слідування», називаються **лінійними**. Дії виконуються одна за одною. Кожна команда є обов'язковою для виконання.

Алгоритми, в основі яких лежить структура «розгалуження», називаються **алгоритмами з розгалуженнями**. Передбачає виконання однієї із зазначених дій в залежності від справедливості заданої умови.

Алгоритми, в основі яких лежить структура «повторення», називають **циклічними**. Кожний циклічний алгоритм має умову, яка приймає значення істинності чи хибності. Якщо умова істинна, то алгоритм (зазвичай) закінчується, а якщо хибна - то продовжується до досягнення істинності.

Розглянемо алгоритми зі структурою повторення, тобто циклічні алгоритми.»

«У деяких алгоритмах передбачається можливість багаторазового виконання записаної сукупності дій. Такі алгоритми називають *циклічними* (циклом), а послідовність інструкцій, призначену для багаторазового виконання – *тілом циклу*. Ітерація – виконання тіла циклу один раз.

Для побудови циклічного алгоритму необхідно:

- визначити дії, які необхідно виконати до входу в цикл, тобто провести підготовку циклу;
- визначити операції, які ввійдуть до циклу;
- скласти умову виходу з циклу. [2]

Прості цикли з параметром.

Якщо в процесі перетворення інформації є змінна, значення якої змінюється за відомим правилом, або відомі межі її зміни, то можна визначити кількість повторень ітерацій циклу та організувати вихід із циклічного процесу. Такі цикли називають *циклами з параметром* (див. Рисунок 3.1), а відповідну змінну - *параметром циклу*. Використовуються у випадку, коли кількість повторень відома наперед. З кожним виконанням тіла циклу, значення параметру змінюється (збільшується або зменшується).

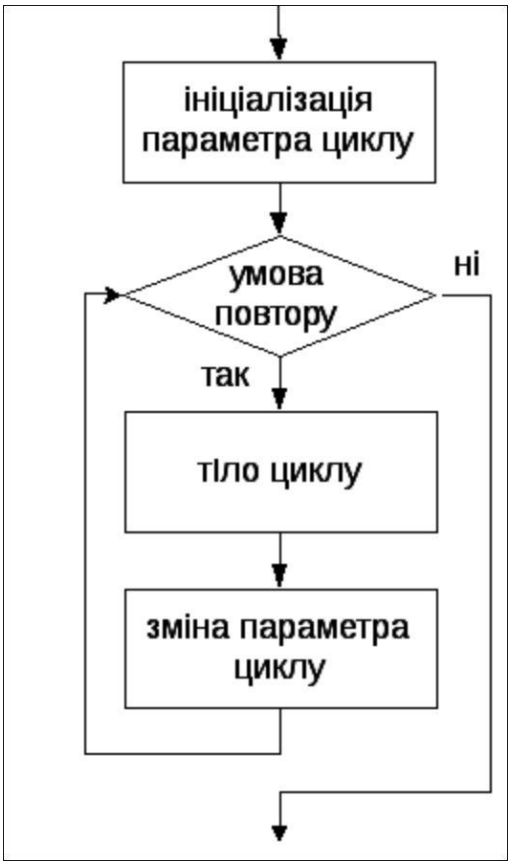

Рисунок 3.1 – Приклад блок-схеми циклу з параметром

Відповідним параметром циклу може бути:

- змінна, яка належить до оброблюваної інформації;
- індексна змінна, якщо оброблювана інформація є масивом;
- коефіцієнти, що змінюються за законом арифметичної прогресії. [2]

Ітераційні цикли.

Розв'язання систем лінійних алгебраїчних рівнянь з десятками та сотнями невідомих, пошук коренів алгебраїчних рівнянь високих степенів та коренів трансцендентних рівнянь, розв'язання систем диференційних рівнянь, інтерполяція та екстраполяція функцій, обчислення значень функції за допомогою рядів, інтегрування тощо — усі ці задачі розв'язуються за допомогою циклічних алгоритмів, що реалізують циклічні ітераційні процеси, для яких заздалегідь неможливо визначити кількість повторень циклу. Під час реалізації ітераційних процесів в алгоритмах повинне забезпечуватися виконання умови виходу з циклу. Тому необхідно сформулювати умову виходу з циклу з використанням особливостей самої задачі. [2]

Розрізняють два типи ітераційних циклів – з *передумовою* і з *постумовою* (див. Рисунок 3.2, 3.3).

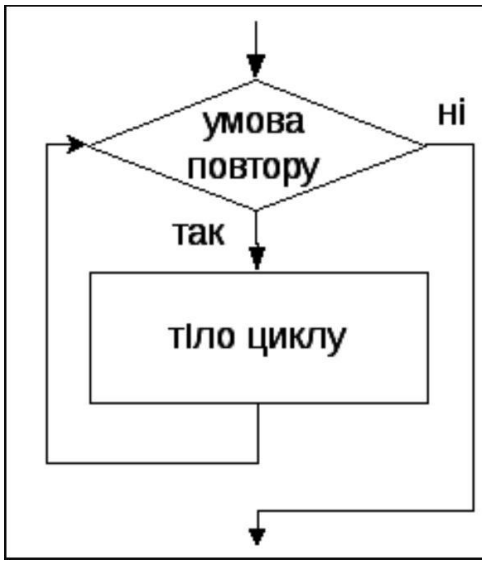

Рисунок 3.2 - Цикл з передумовою

Рисунок 3.3 - Цикл з постумовою

Цикл з передумовою раціонально використовувати під час реалізації процесів, де кількість повторень невизначена. Блок-схема реалізації зображена на Рисунок 3.2. Тіло циклу буде виконуватися до тих пір, поки умова повтору даватиме значення TRUE. Для нормальної роботи циклу і його логічного завершення потрібно, щоб усередині тіла циклу відбувалася зміна заданих значеннь змінних, з яких складається умова повтору. Можлива також ситуація, коли цикл з перезумовою не виконається жодного разу.

Цикл з постумовою відрізняється від циклу з передумовою тим, що умова перевіряється уже після виконання тіла циклу. Це забезпечує виконання циклу мінімум один раз, і може бути корисним в деяких ситуаціях. Порядок виконання циклу: виконуються оператори, з яких складається тіло циклу, після чого слідує перевірка умови. Якщо умова істинна – цикл завершується.

Складні циклічні процеси.

З алгоритмічних структур зазначених типів можна будувати складні циклічні процеси із вкладеними циклами. У цьому випадку виділяються *внутрішній* і *зовнішній* цикли. Для кожної зміни значення параметра у зовнішньому циклі відбувається багаторазове виконання дій у внутрішньому циклі, який називається *вкладеним*. Кількість вкладених циклів не обмежується. [2]

Треба зауважити, що як внутрішній, так і зовнішній цикли можуть бути параметричними та ітераційними, кожен, у свою чергу - бути складним (див. Рисунок 2.4). Але вони повинні цілком вкладатися один в одний і ніколи не перетинатися частково.

Оскільки особливістю базових алгоритмічних конструкцій є те, що будь-яка з них має лише один вхід і один вихід, то вони можуть вкладатися одна в одну довільним чином.[2]

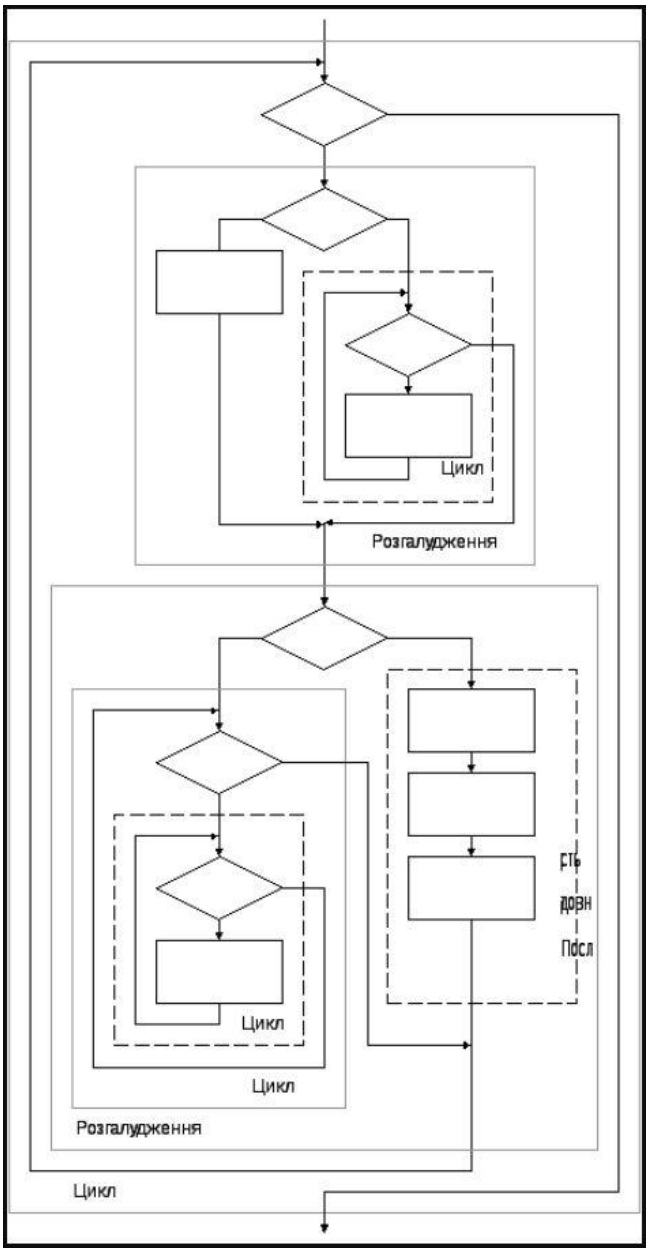

Рисунок 3.4 – Приклад блок-схеми з внутрішнім та зовнішнім циклами

Розглянемо циклічний алгоритм побудови блок-схем на прикладі циклу for.

Перед початком виконання тіла циклу перевіряється умова виконання циклу, що задається спочатку.

Потім задається умова виконування циклу; при невідповідності умові цикл припиняє свою роботу.

Потім виконується тіло циклу, наприклад додавання чи послідовний вивід вмісту змінної.

При закінченні відповідності умовам чи по завершенню виконання тілу циклу, він завершує свою роботу.

3.2 Алгоритм роботи тренажера

Розроблене програмне забезпечення у вигляді тренажеру має наступну структуру:

1. Головне меню;
2. Вивід теорії та основного завдання роботи;
3. Вивід практичних завдань у вигляді тестів;

Передбачена навігація між інформацією в тренажері та вивід повідомлення про правильність відповіді. Після закінчення роботи з тренажером стає активною кнопка повтору роботи.

Приклад практичного завдання в тренажері:

Завдання 1: Виберіть блок, що відповідає за початок алгоритму.

1 2 3

Для вводу даних необхідно натиснути на відповідне зображення в тренажері.

Правильна відповідь – 1.

Після вибору правильної відповіді виводиться відповідне повідомлення та з’являється кнопка «Далі».

Після вибору неправильної відповіді виводиться повідомлення про помилку.

3.3 Блок-схема програми-тренажера

Рисунок 3.5 – Блок-схема тренажера

4 Практична частина

4.1 Опис програмного забезпечення

Розроблене програмне забезпечення являє собою набір елементів перехід між якими виконується за допомогою скриптів для кнопок в Unity.

Розглянемо кнопки виходу та повернення в головне меню.

Кнопка виходу:

```
public void MenuPressed() {
    Application.Quit();
 }
```

Кнопка повернення в меню:

```
public void ExitPressed() {
    SceneManager.LoadScene("StartMenu");
}
```

Тоді, наприклад, для переходу з першого елементу тренажеру на елемент з блок-схемою необхідно застосувати наступний скрипт:

```
public void NextFirstElem() {
    SceneManager.LoadScene("FirstBlokSchm");
}
```

Під час запуску програмного забезпечення користувач переходить до елементу з головним меню (див. Рисунок 4.1), з якого за допомогою кнопки «Розпочати роботу» може перейти то теоретичної інформації (див. Рисунок 4.2).

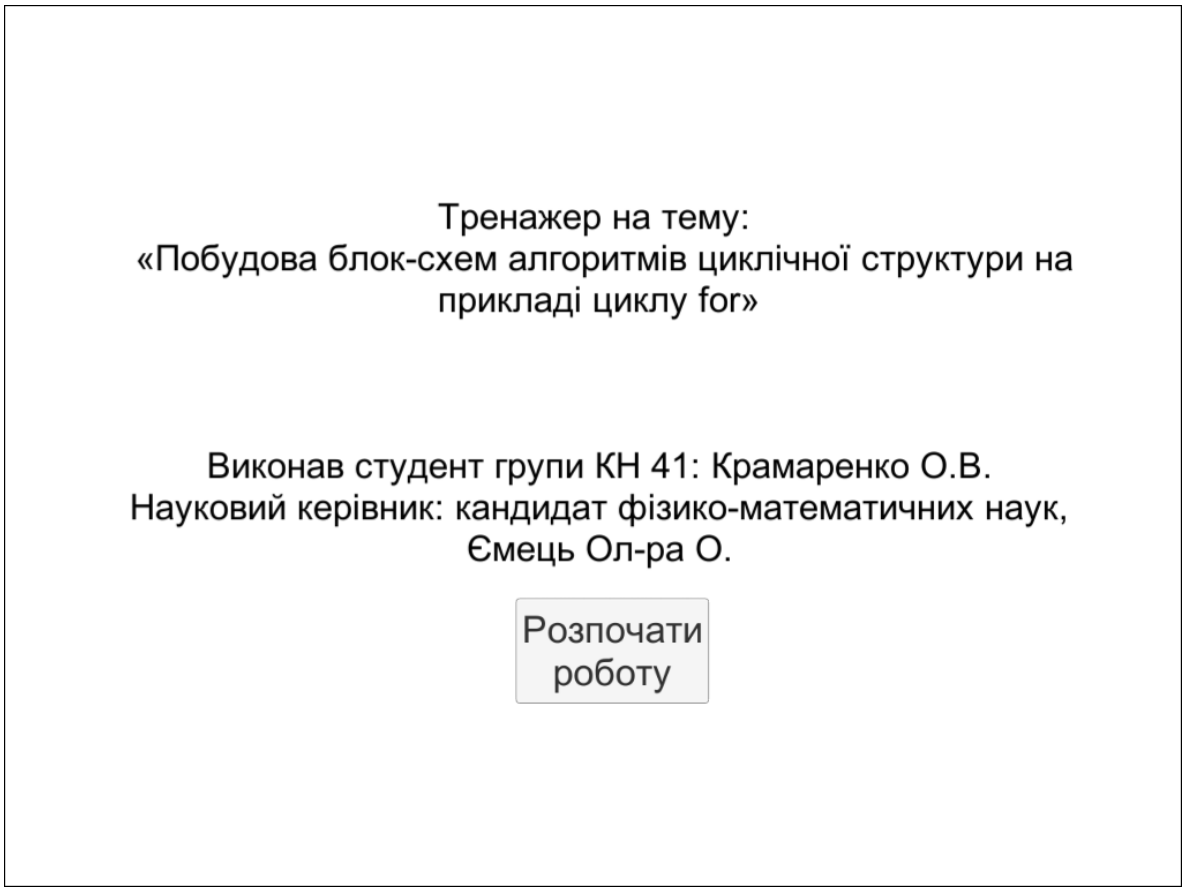

Рисунок 4.1 – Головне меню тренажеру

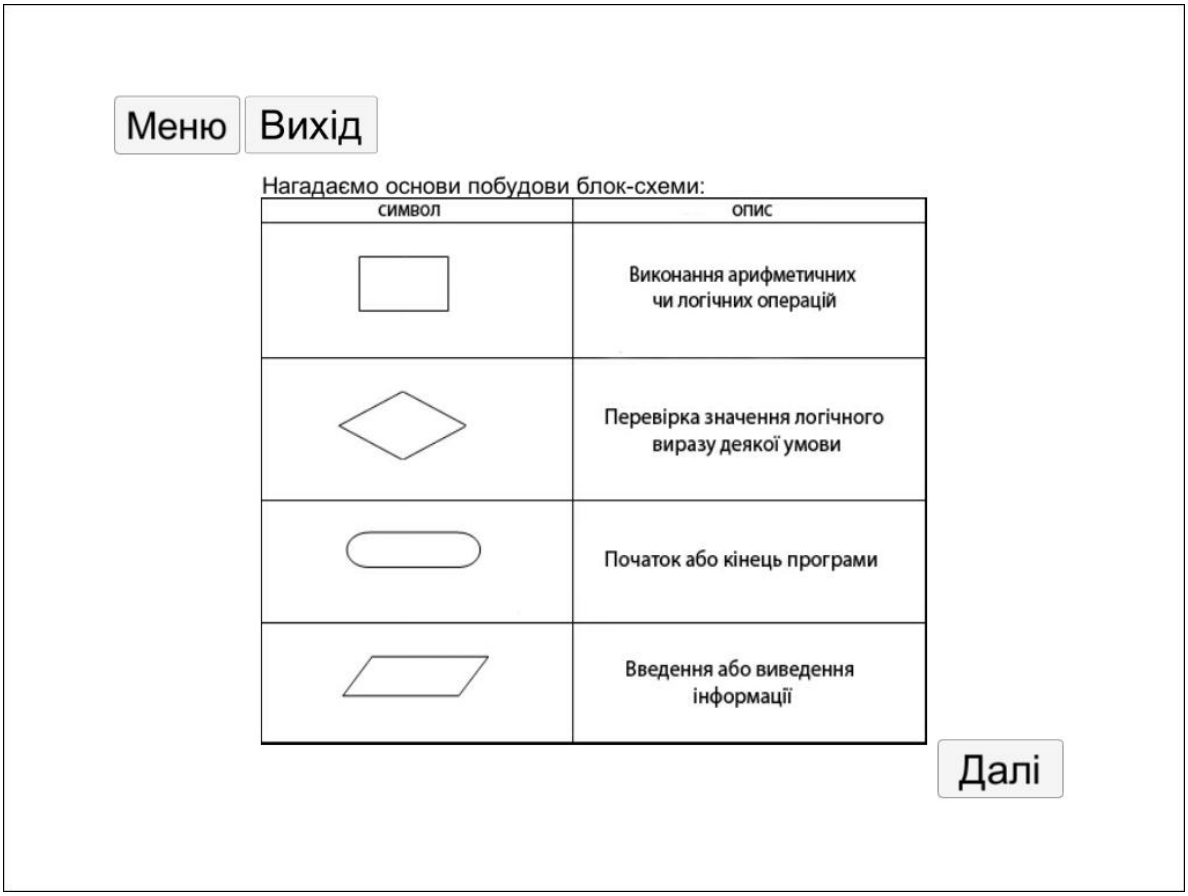

Рисунок 4.2 – Елемент з теоретичною інформацією

Після ознайомлення з теорією та натисненням на кнопку «Далі» користувач переходить до прикладу, по якому будується блок-схема. (див.Рисунок 4.3)

Рисунок 4.3 – Вікно прикладу

Далі починається частина з практичними завданнями. З’являється кнопка «Умова», де можна в будь-який момент побачити приклад (див. Рисунок 4.4).

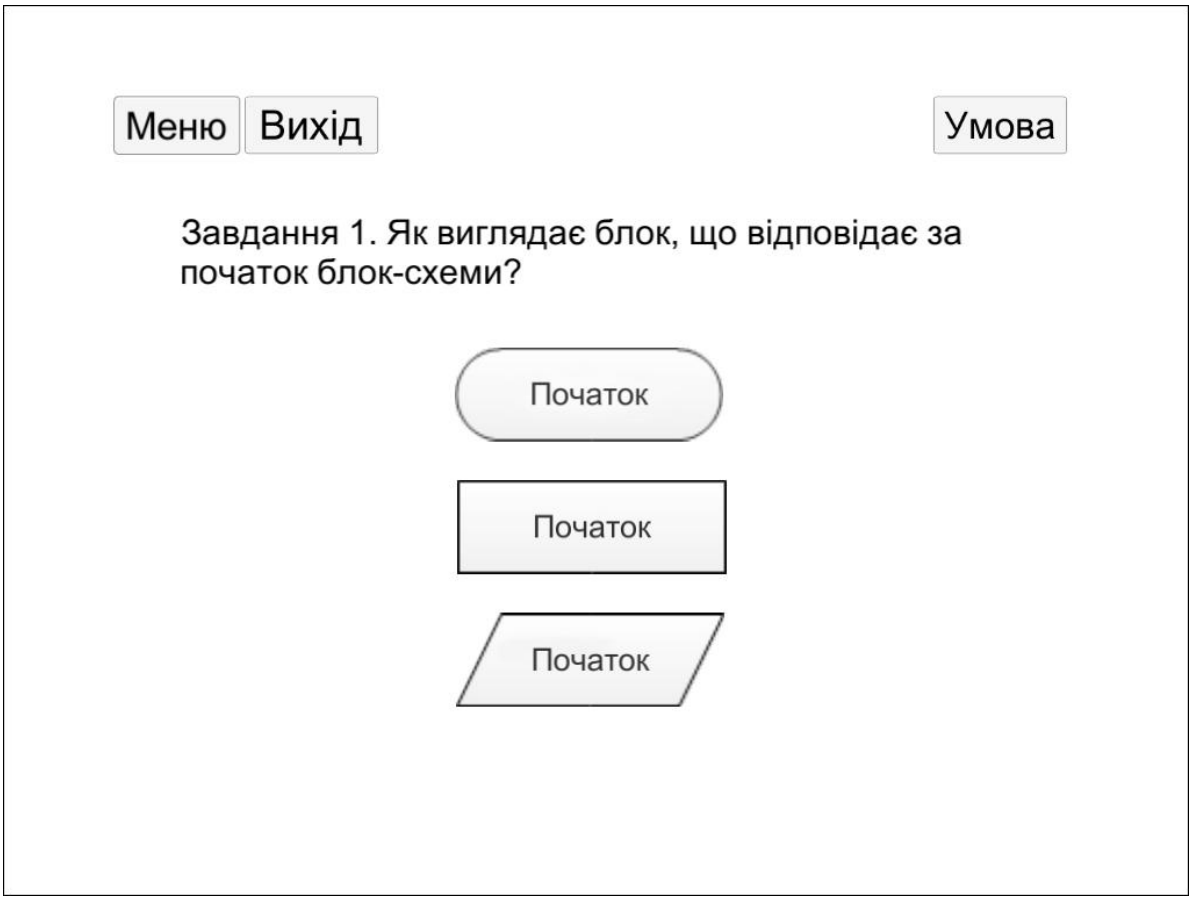

Рисунок 4.4 – Елемент з практичними завданнями

Після вибору правильної відповіді, виводиться відповідне повідомлення та активується кнопка «Далі» (див. Рисунок 4.5). При виборі неправильної відповіді виводиться лише повідомлення «Не правильно» (див. Рисунок 4.6).

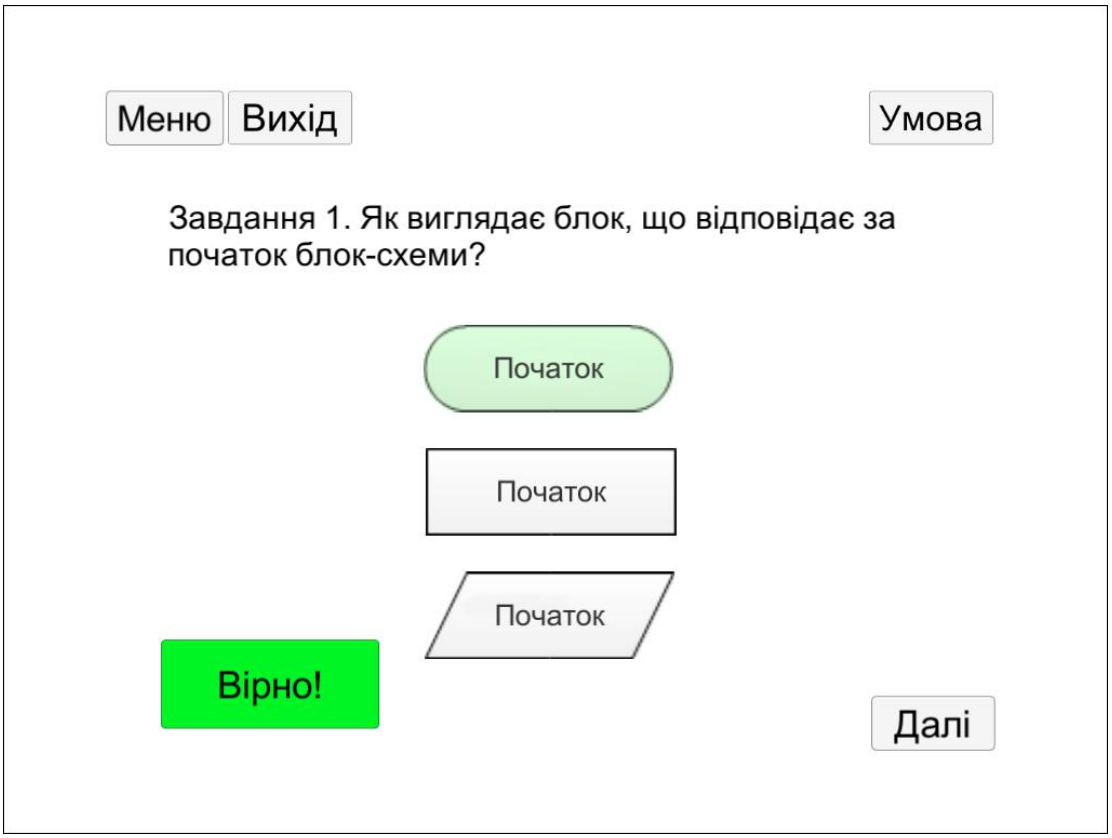

Рисунок 4.5 – Елемент з практичними завданнями після вибору правильної відповіді

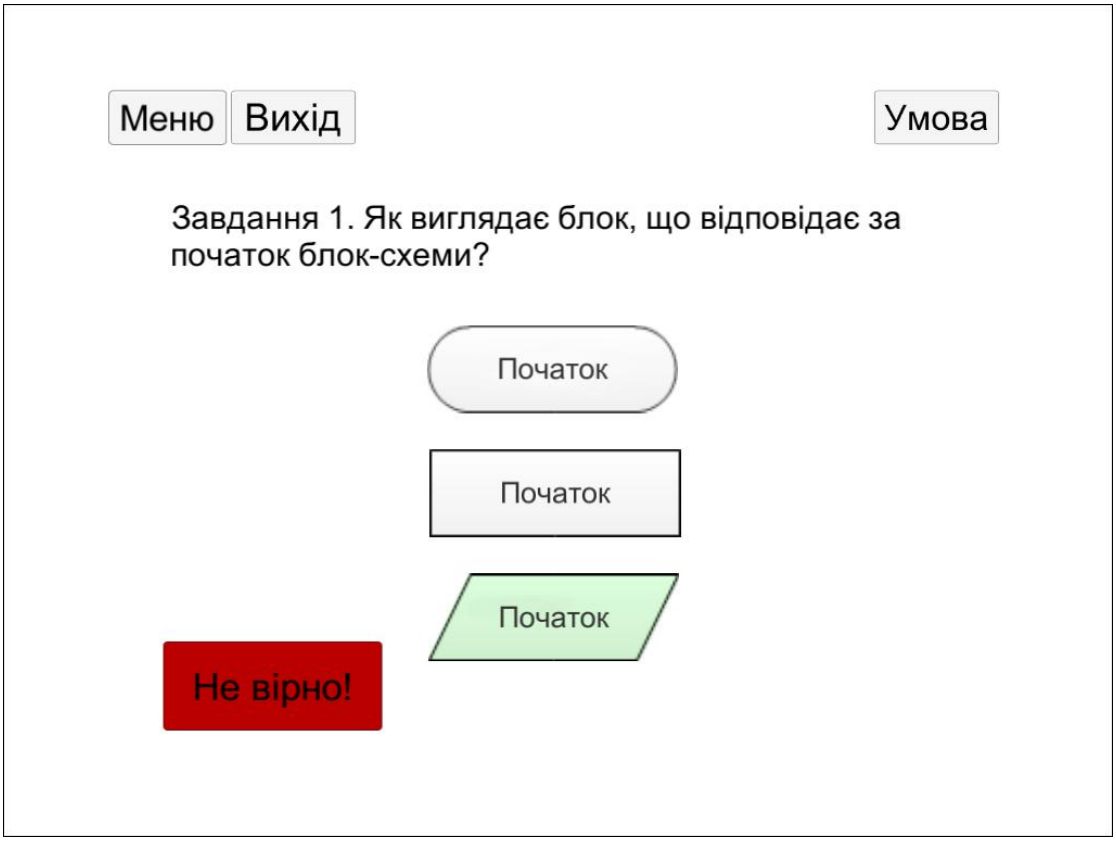

Рисунок 4.6 – Елемент з практичними завданнями після вибору неправильної відповіді

Після кожного практичного завдання та натиснення на кнопку «Далі» користувач може наглядно бачити покрокову побудову блок-схеми (див. Рисунок 4.7).

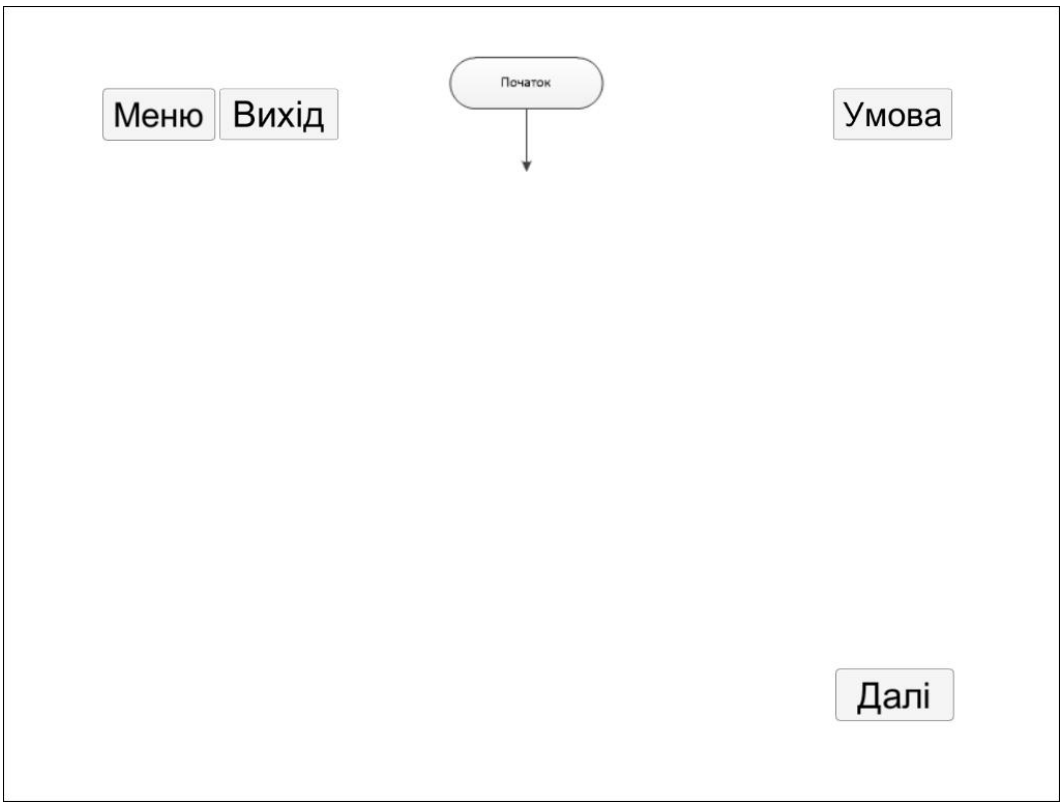

Рисунок 4.7 – Елемент з блок-схемою після виконання першого практичного завдання

В будь-який момент можна повернутися в головне меню чи вийти за допомогою відповідних кнопок (див. Рисунок 4.8).

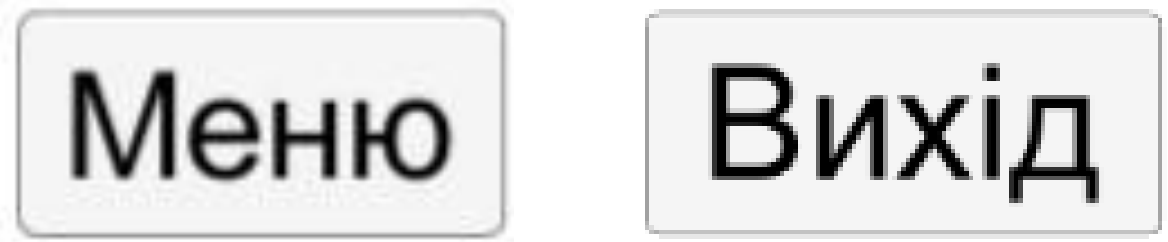

Рисунок 4.8 – Кнопки повернення до меню та виходу

4.2 Інструкція для роботи з тренажером

Для початку роботи необхідно запустити Trainer.exe з архіву. Після цього слід обрати розширення та якість картинки для показу навчальних матеріалів, або продовжити з налаштуваннями за замовчуванням (див. Рисунок 4.9).

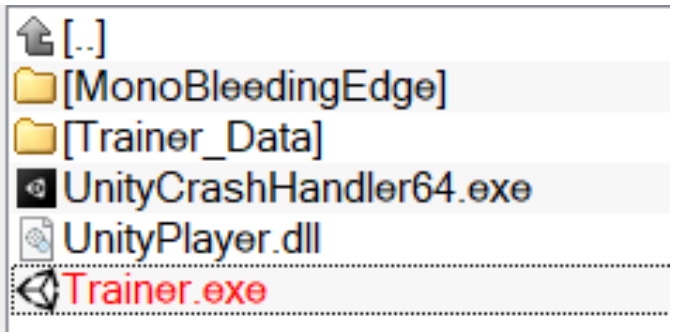

Рисунок 4.9 – Програмне забезпечення у вигляді тренажеру

Після вибору налаштувань користувач переходить до головного меню програми на якому зображено тему та виконавців роботи. (Рисунок 4.10)

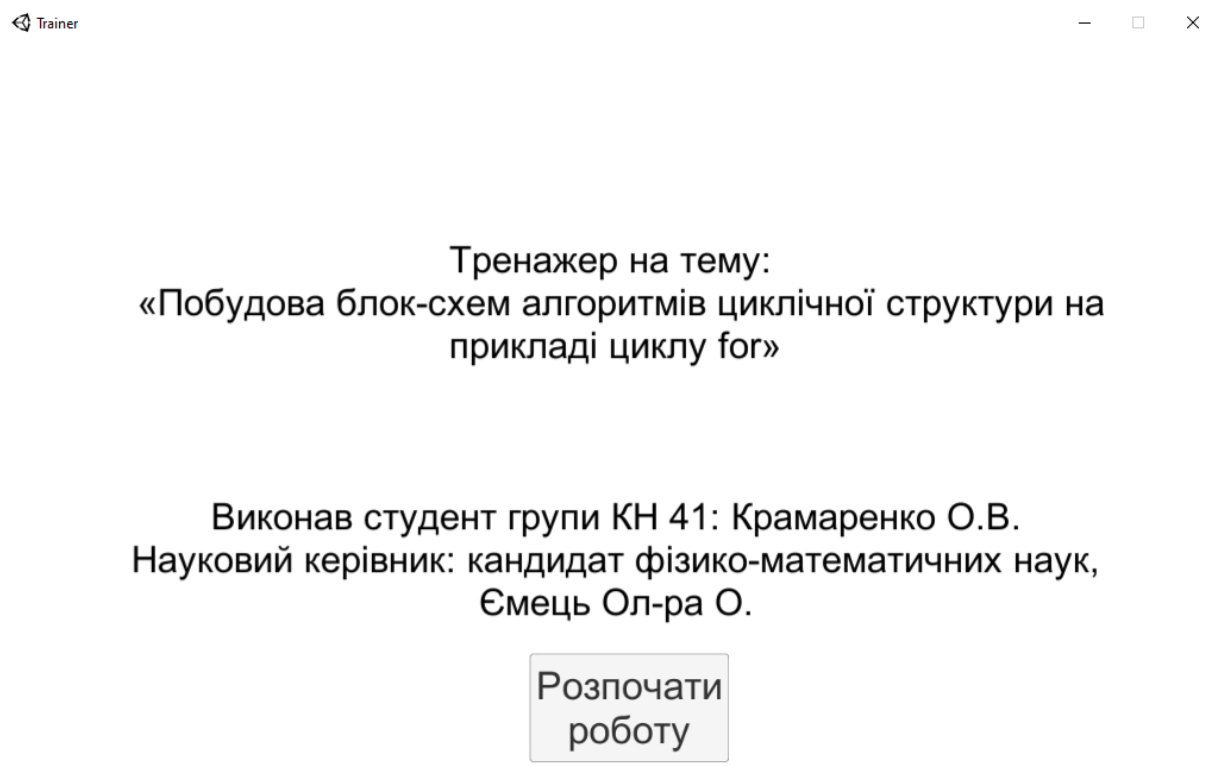

Рисунок 4.10 – Головне меню програми

По натисненні на кнопку «Розпочати роботу» користувач переходить до навчальних матеріалів тренажера. Першим елементом є коротка довідка до тренажеру (див. Рисунок 4.11).

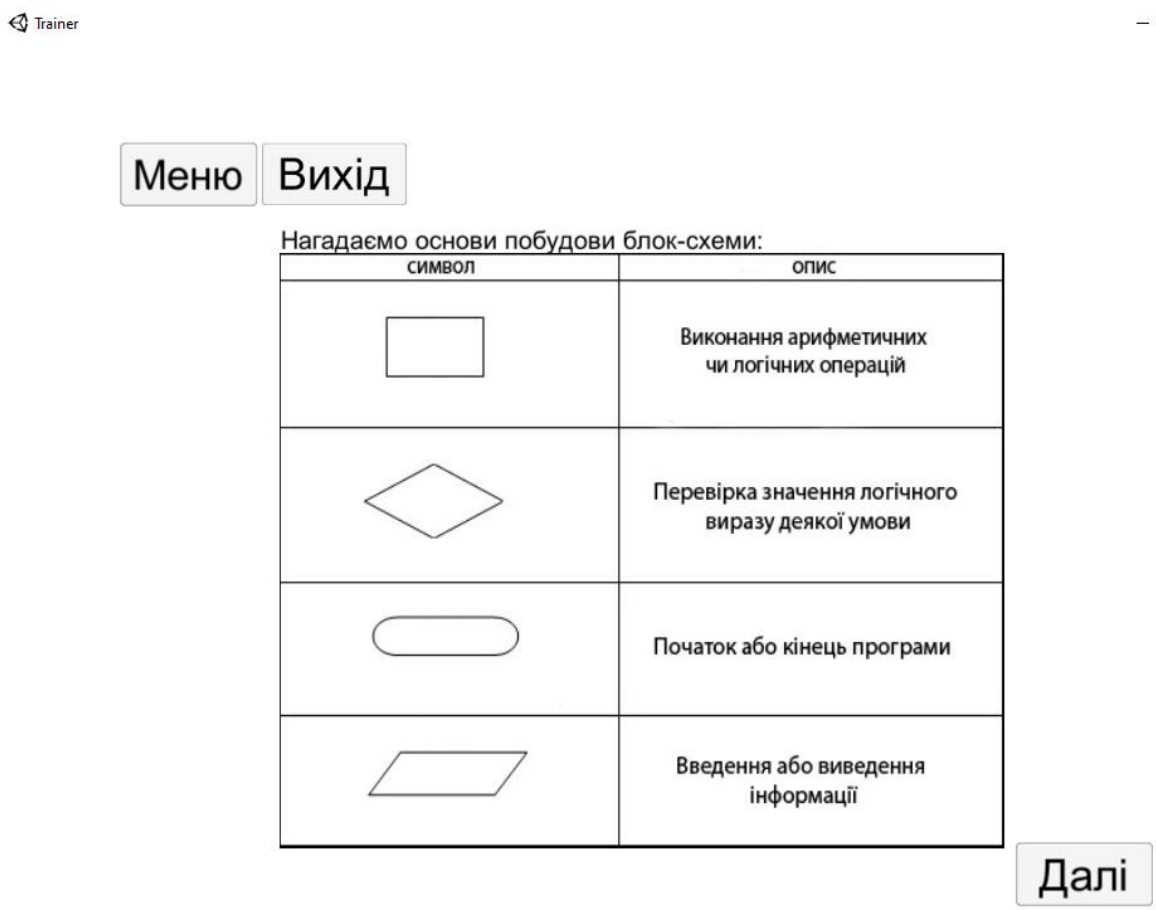

Рисунок 4.11 – Довідка до тренажеру

Після ознайомлення з основними правилами для побудови блок-схем користувач переходить до умови задачі, що буде розглядатися в тренажері. Пізніше під час роботи цю умову можна буде викликати через відповідну кнопку. (див. Рисунок 4.12).

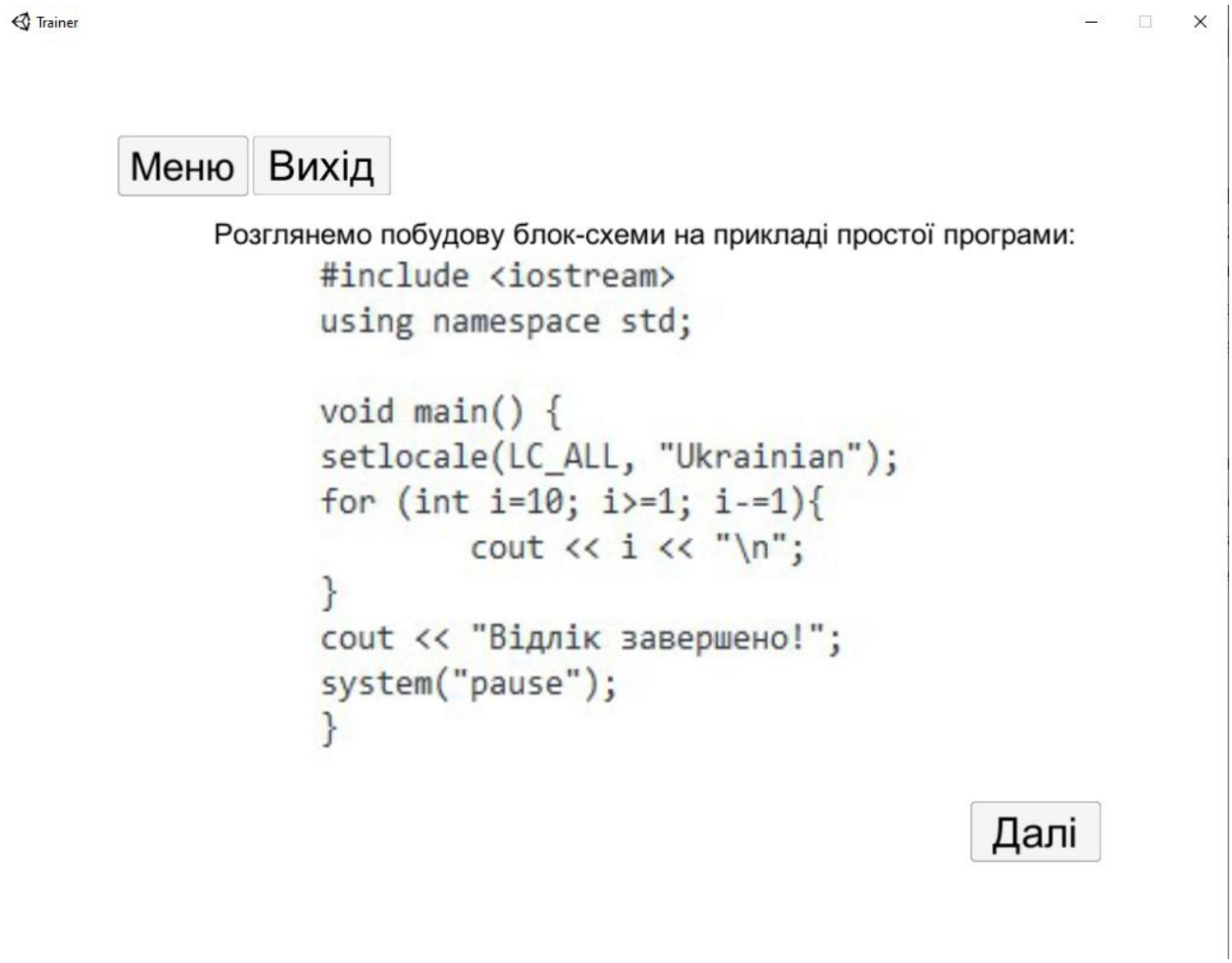

Рисунок 4.12 – Умова практичного завдання

Після ознайомлення з умовою користувач переходить до низки практичних завдань. Робота з практичними завданнями зводиться до вибору правильного варіанту відповіді. Всі варіанти відповіді реалізовано в якості кнопок задля зручної роботи з ними. У випадку вибору неправильного варіанту відповіді користувач отримує відповідне повідомлення – «Не вірно!» (див. Рисунок 4.14). У випадку вибору правильного варіанту користувач отримує повідомлення «Вірно!» та доступ до кнопки «Далі» (див. Рисунок 4.15).

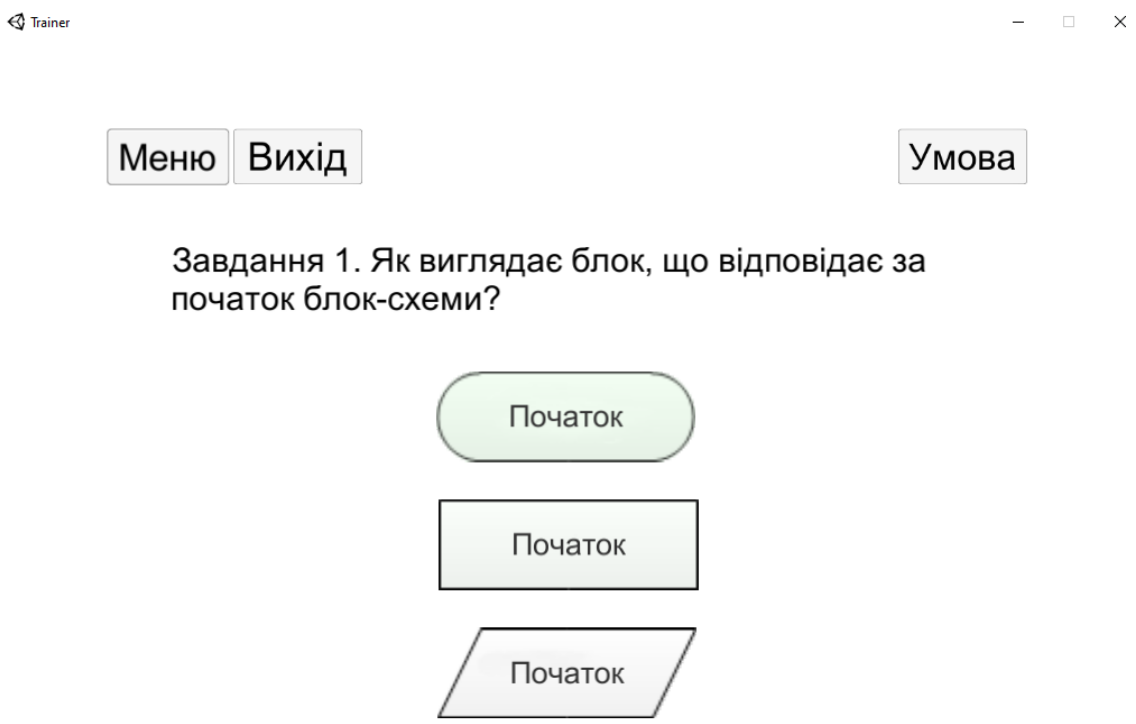

Рисунок 4.13 – Перше завдання в тренажері

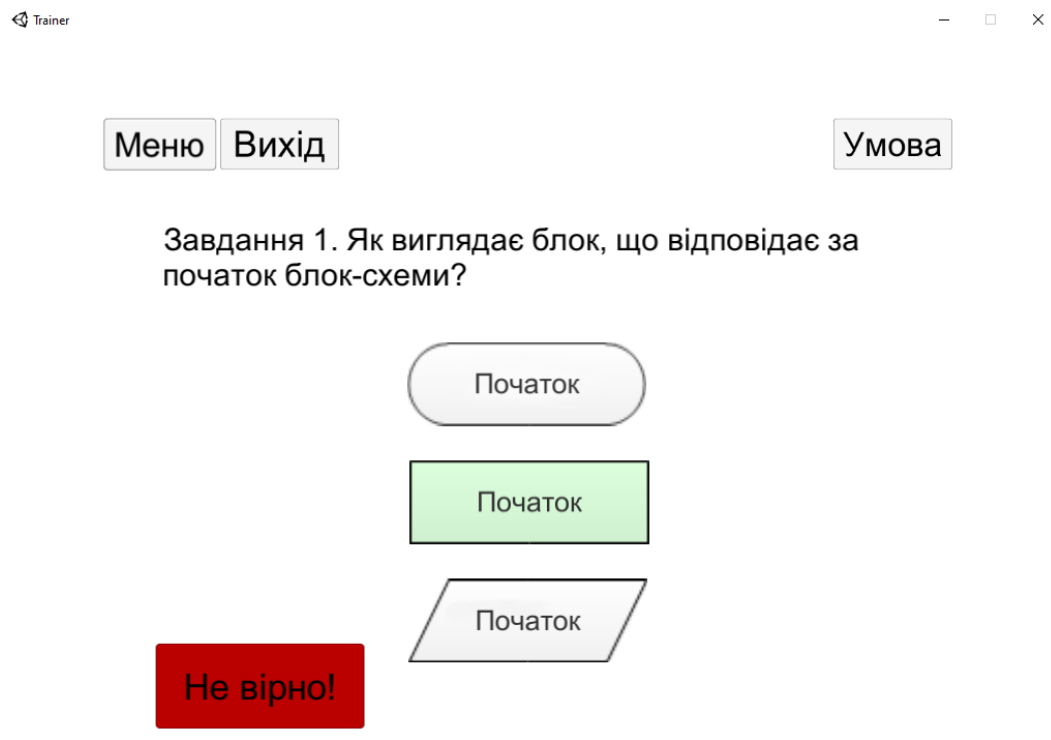

Рисунок 4.14 – Повідомлення у разі вибору неправильної відповіді

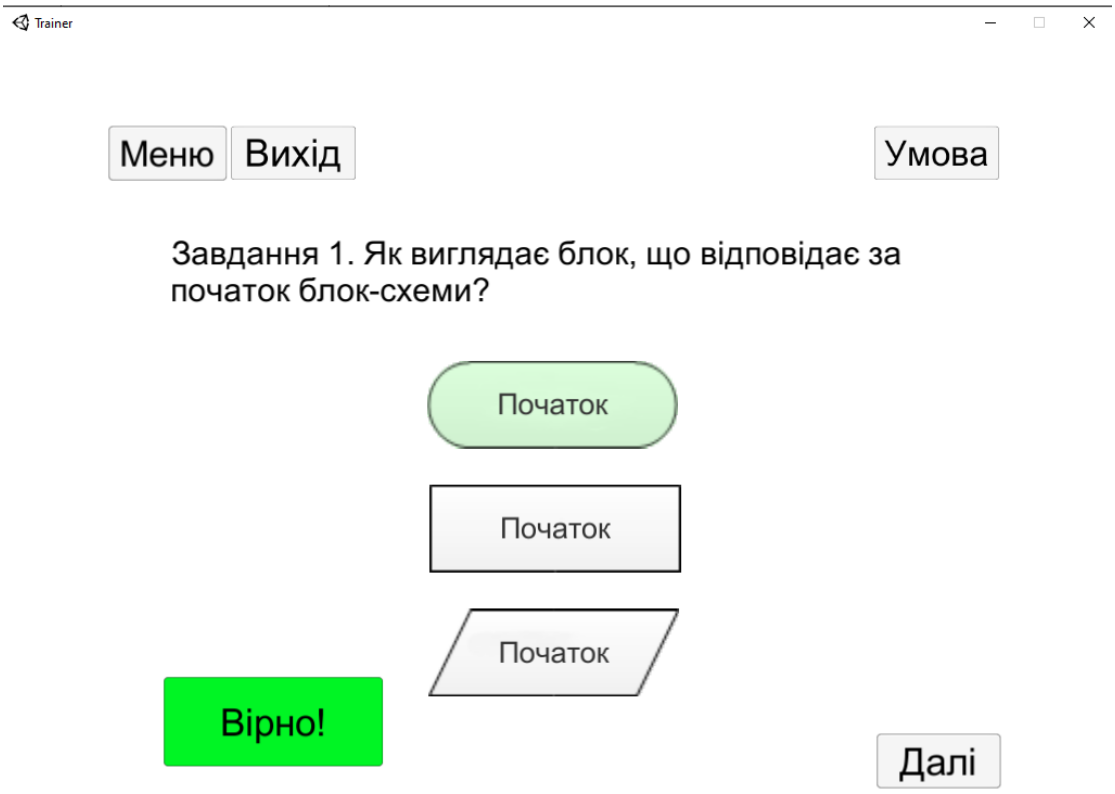

Рисунок 4.15 – Повідомлення у разі вибору правильної відповіді

Після вибору правильної відповіді та після натиснення кнопки «Далі» користувач переходить до зображення побудованої блок-схеми до циклу на даному етапі роботи.

З будь-якого місця роботи можливо подивитися на умову до завдання натиснувши кнопку «Умова», закрити умову можна повторно натиснувши на кнопку. (Рисунок 4.16)

Рисунок 4.16 – Кнопка «Умова»

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.17)

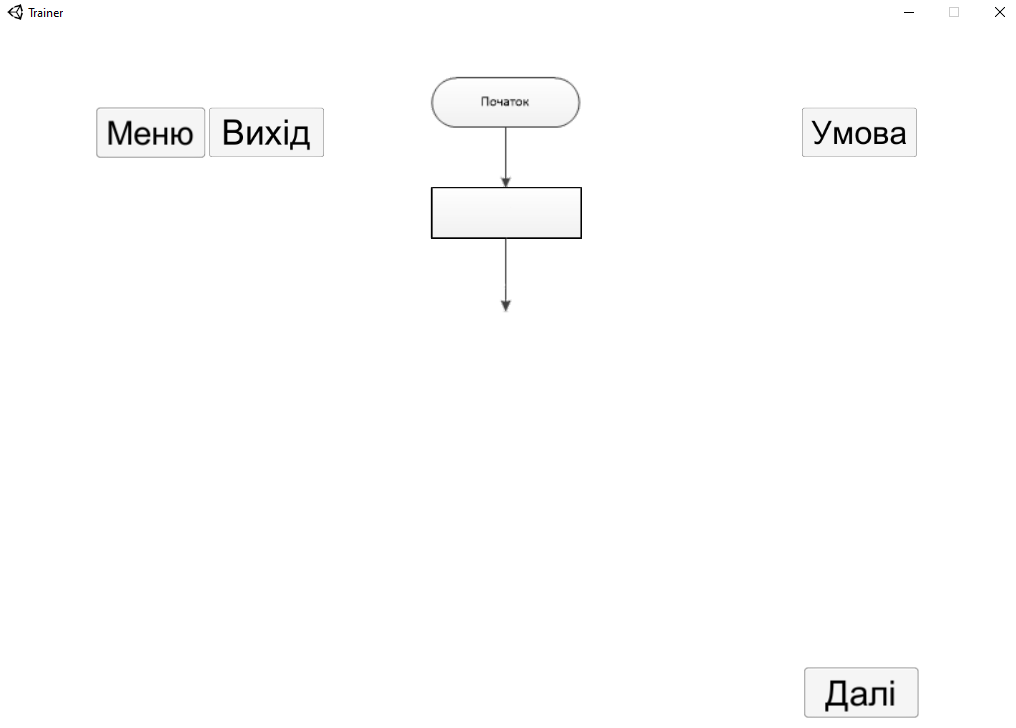

Рисунок 4.17 – Блок-схема алгоритму на даному етапі роботи

Після ознайомлення з блок-схемою користувач переходить до наступного практичного завдання. Починаючи з третього практичного завдання до кнопки «Умова» додано шматок коду біля тексту завдання для пришвидшення роботи. (Рисунок 4.18)

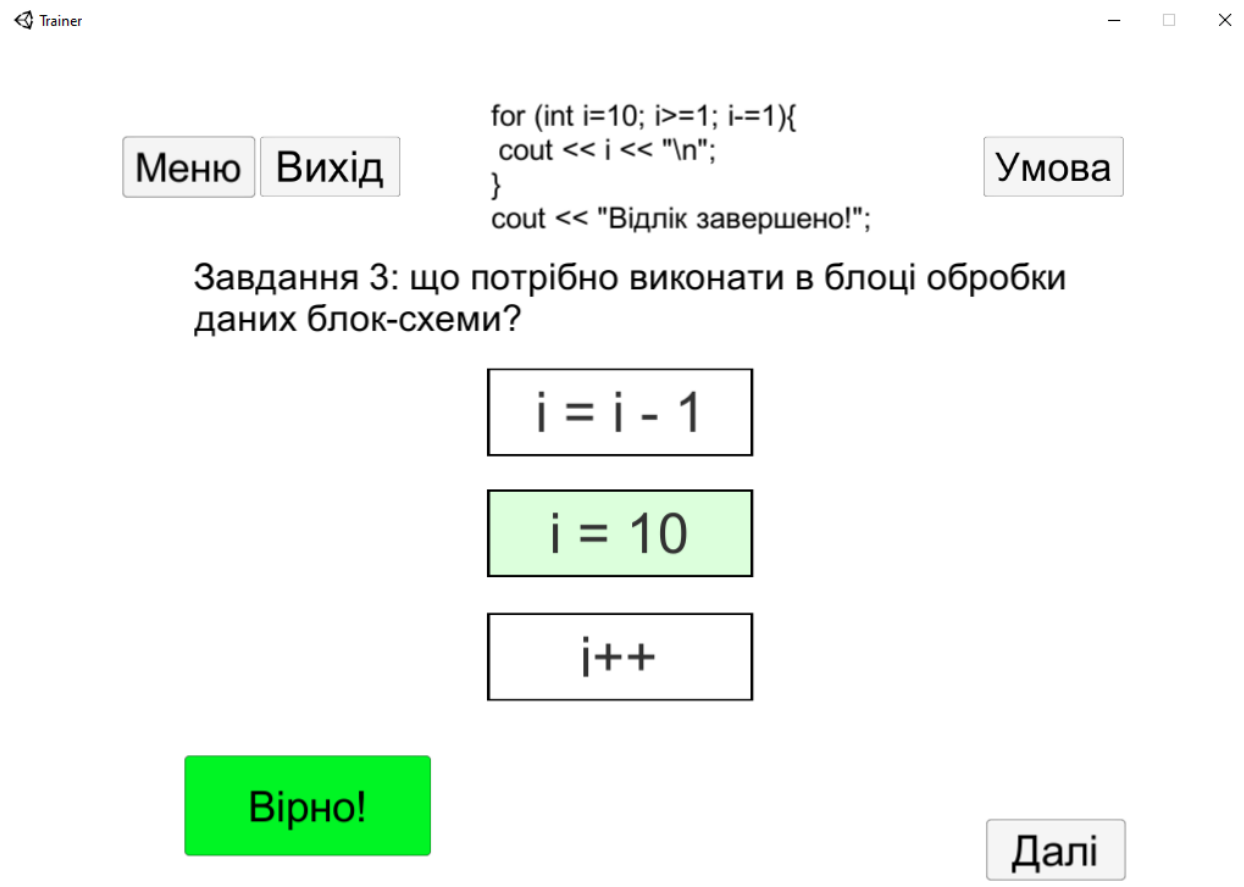

Рисунок 4.18 – Вибір правильного варіанту відповіді на третє завдання

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.19)

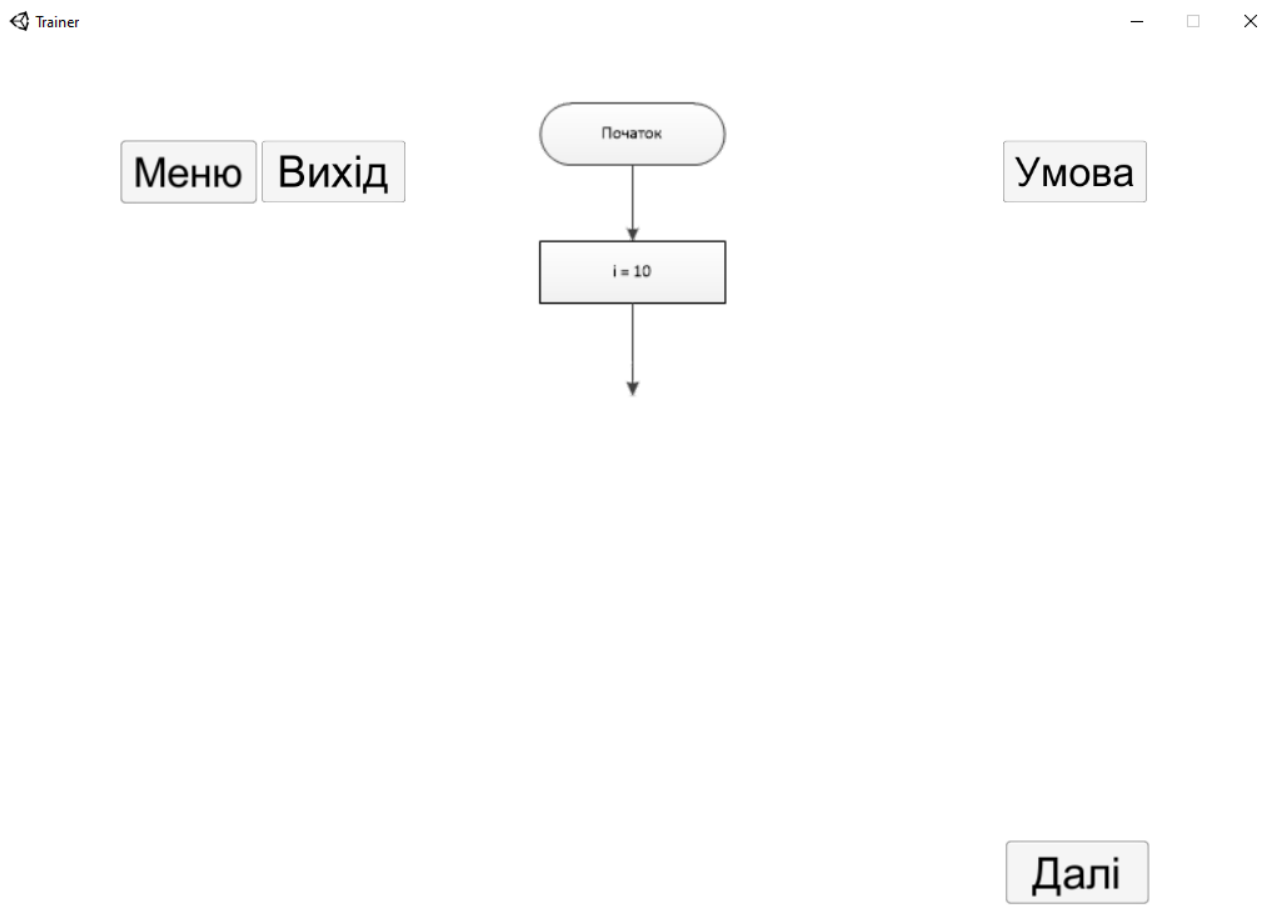

Рисунок 4.19 - Блок-схема алгоритму на даному етапі роботи

Після ознайомлення з блок-схемою користувач переходить до наступного практичного завдання. (Рисунок 4.20)

Рисунок 4.20 - Вибір правильного варіанту відповіді на четверте завдання

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.21)

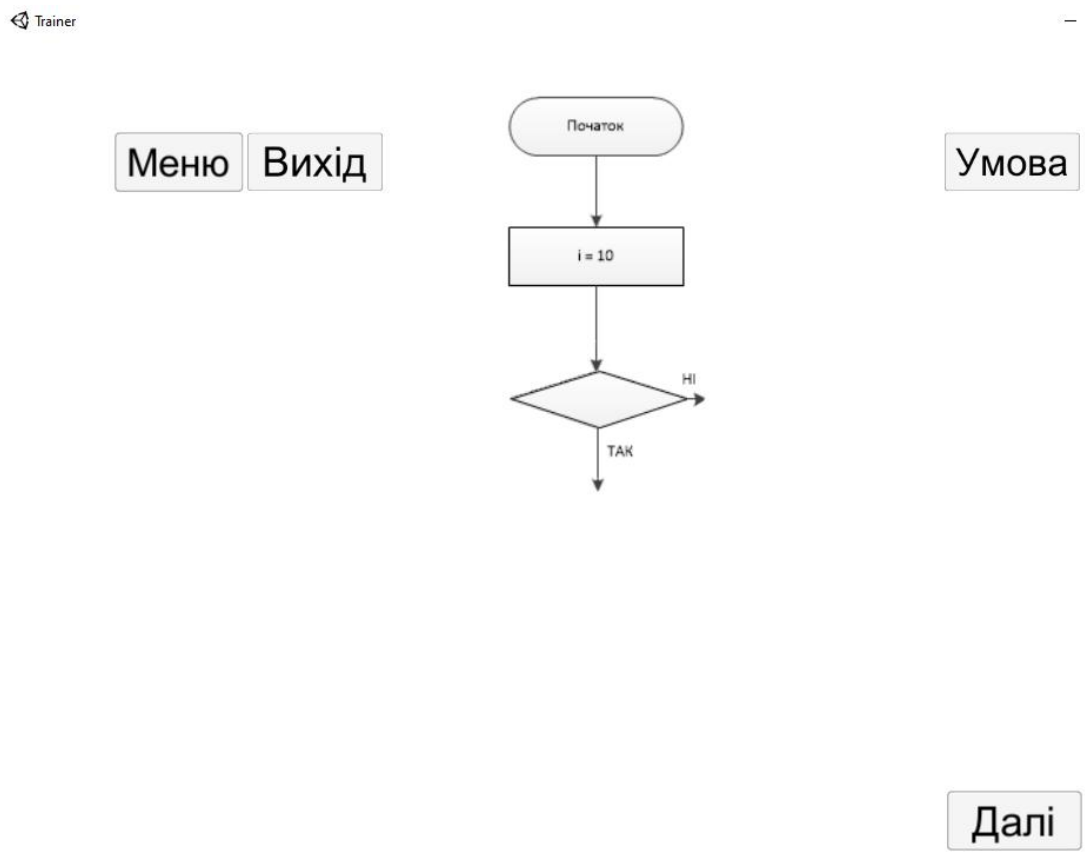

Рисунок 4.21 - Блок-схема алгоритму на даному етапі роботи

Після ознайомлення з блок-схемою користувач переходить до наступного практичного завдання. (Рисунок 4.22)

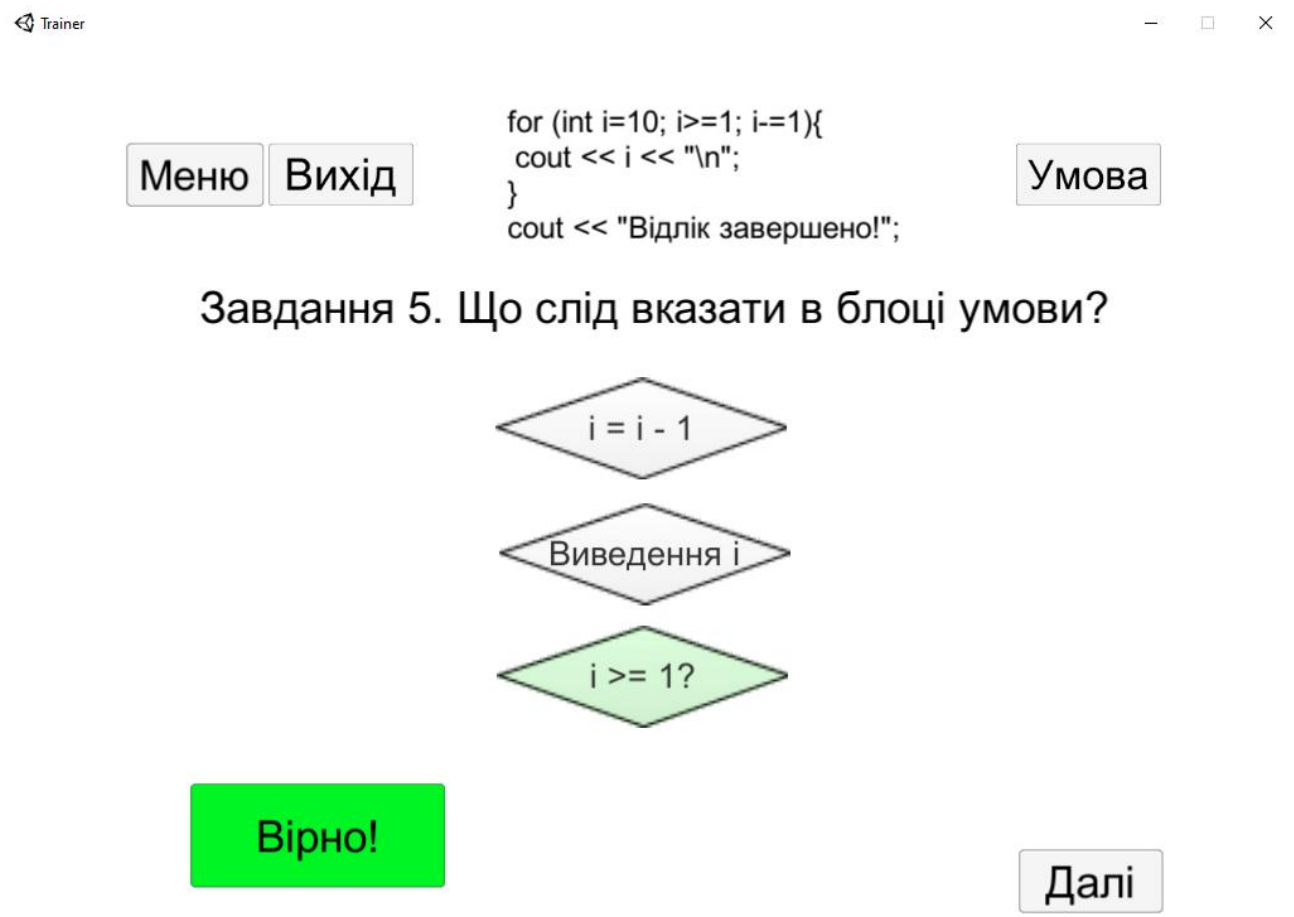

Рисунок 4.22 - Вибір правильного варіанту відповіді на п’яте завдання

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.23)

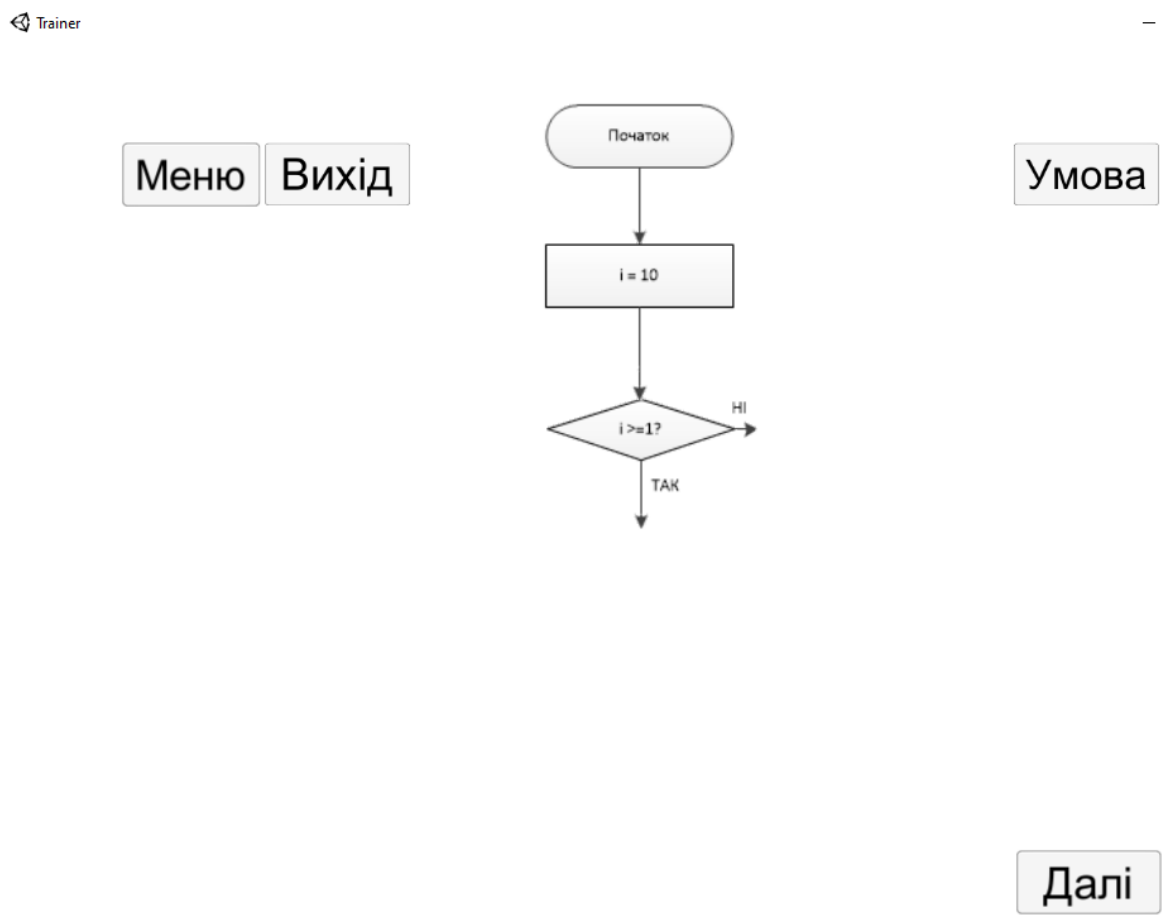

Рисунок 4.23 - Блок-схема алгоритму на даному етапі роботи

Після ознайомлення з блок-схемою користувач переходить до наступного практичного завдання. (Рисунок 4.24)

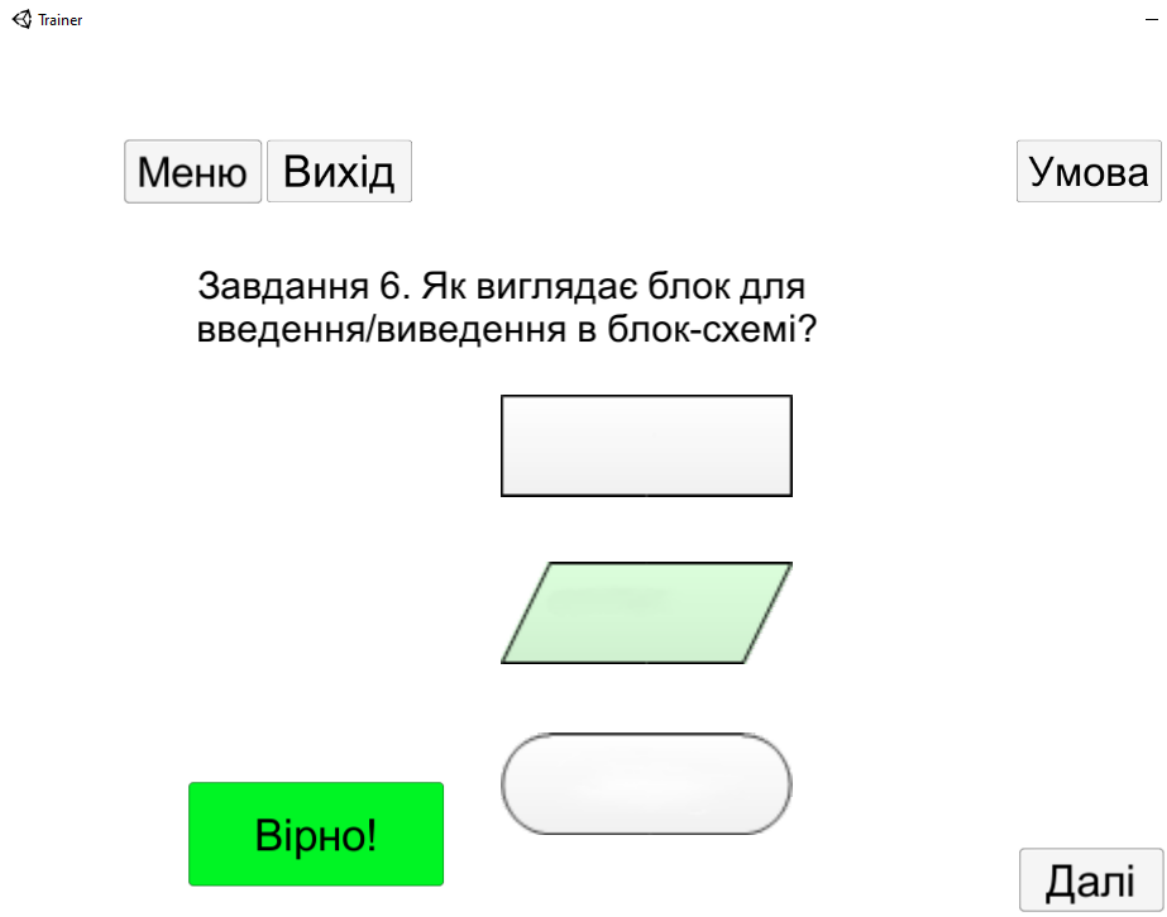

Рисунок 4.24 - Вибір правильного варіанту відповіді на шосте завдання

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.25)

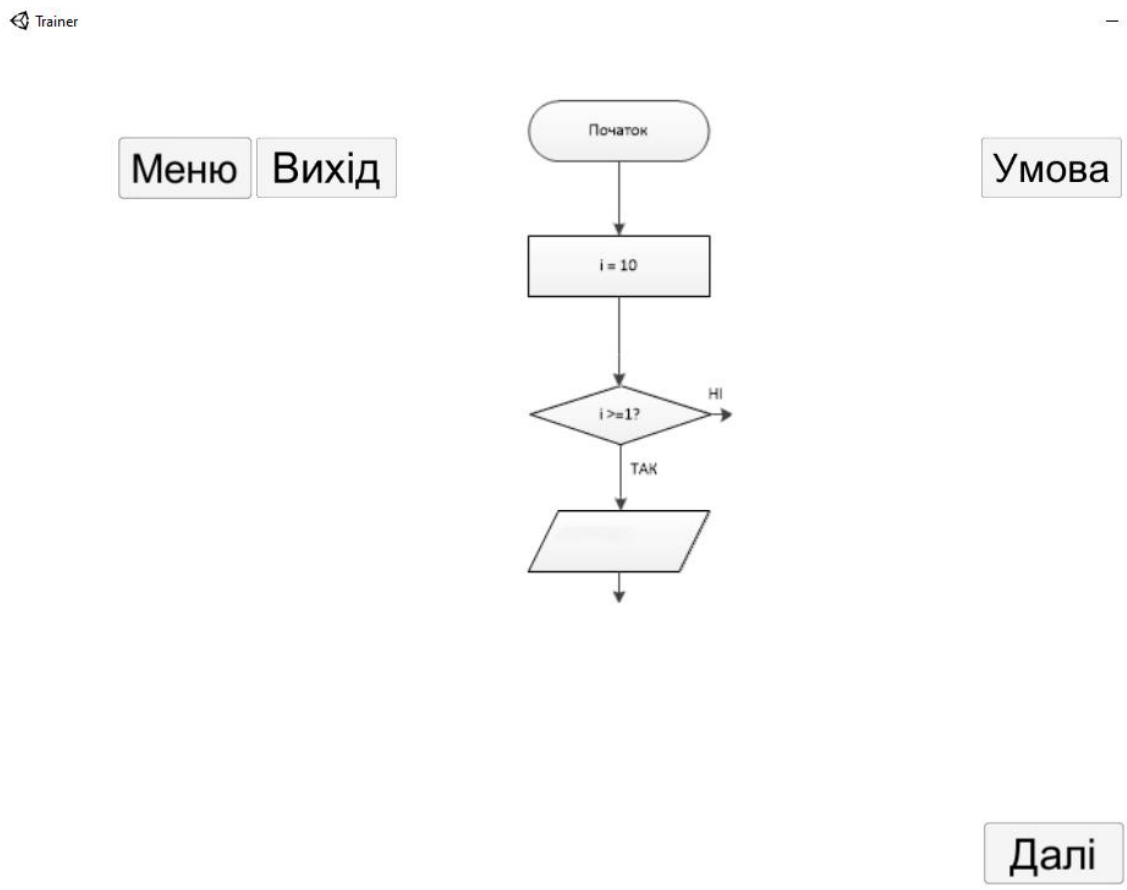

Рисунок 4.25 - Блок-схема алгоритму на даному етапі роботи

Після ознайомлення з блок-схемою користувач переходить до наступного практичного завдання. (Рисунок 4.26)

Рисунок 4.26 - Вибір правильного варіанту відповіді на сьоме завдання

Після вибору правильного варіанту відповіді відбувається перехід до блок-схеми на даному етапі роботи. (Рисунок 4.27)

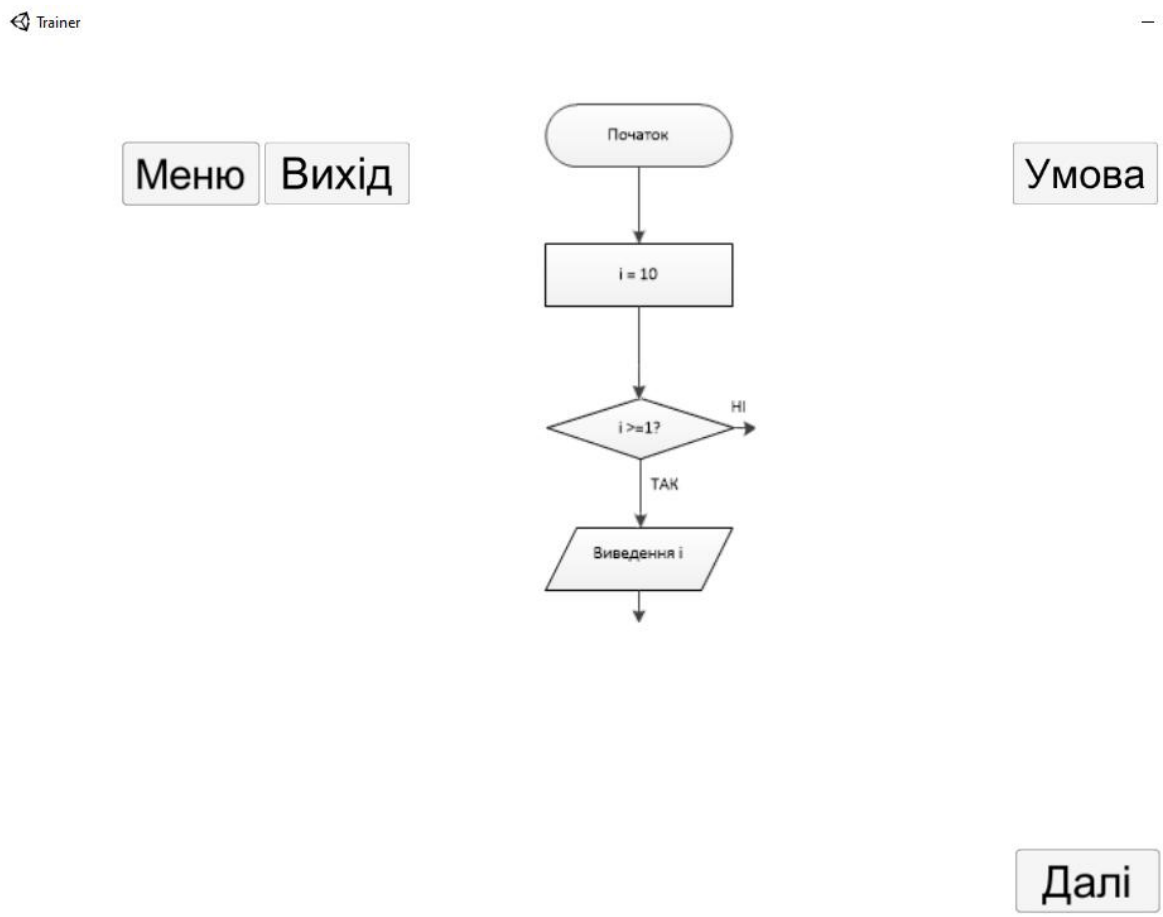

Рисунок 4.27 - Блок-схема алгоритму на даному етапі роботи

Після відповіді на останнє запитання виводиться кінцевий результат блок-схеми. Тут же можна порівняти її з прикладом, заданим на початку роботи.

Після ознайомлення з кінцевим варіантом блок-схеми користувач отримує можливість повторити роботу з тренажером через натиснення кнопки «Повтор?» (Рисунок 4.28)

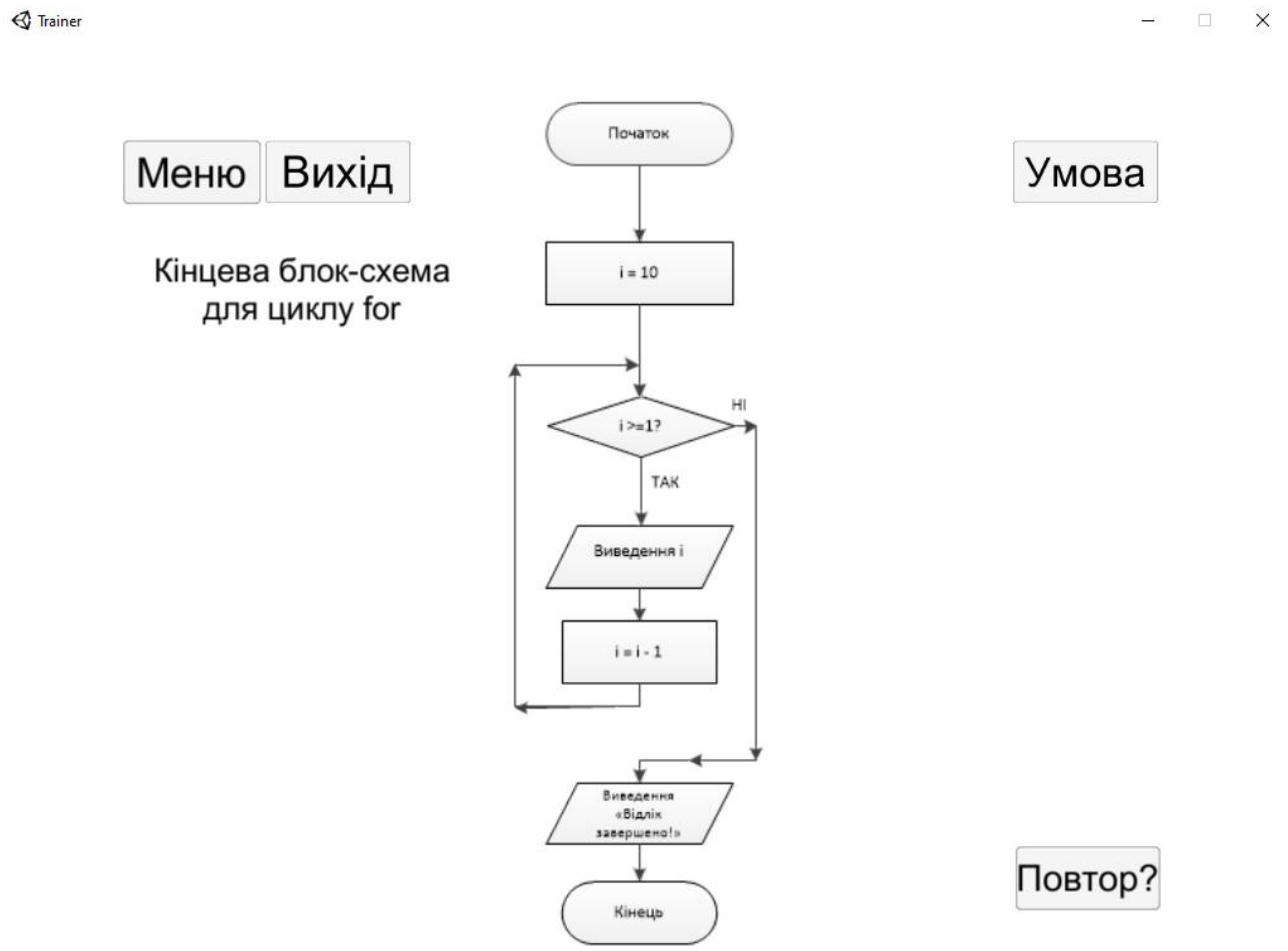

Рисунок 4.28 - Блок-схема алгоритму по завершенню роботи

ВИСНОВКИ

По результатам виконання бакалаврської роботи було виконано мету та завдання бакалаврської роботи, створено програмне забезпечення з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Програмування ІІ».

Плюсами розробленого програмного забезпечення є:

1. Послідовна видача теоретичної інформації та практичних завдань;
2. Покрокове пояснення побудови блок-схем;
3. Розгляд побудови блок-схеми на певному прикладі.

Основні вимоги до бакалаврської роботи виконано, створено алгоритм роботи програмного забезпечення, складено блок-схеми роботи алгоритму та розроблено елементи програмного забезпечення за допомогою середовища розробки Microsoft Visual Studio, Unity та мови програмування C#.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Пояснювальна записка до дипломної на тему: Розробка програмного забезпечення тренажеру з теми: «Дефазифікація нечітких множин» дистанційного навчального курсу «Сучасні методи оптимізації та їх програмування», Антоненко А. А. [Електронний ресурс]

Режим доступу до ресурсу: http://dspace.puet.edu.ua/handle/123456789/8991

2. Пояснювальна записка до бакалаврської роботи на тему: Розробка програмного забезпечення тренажеру з теми: «Злиття впорядкованих послідовностей» дистанційного навчального курсу «Алгоритми та структури даних», Кулинич Марина Костянтинівна [Електронний ресурс]

Режим доступу до ресурсу: http://dspace.puet.edu.ua/handle/123456789/9006

3. Пояснювальна записка до дипломної на тему: Розробка програмного забезпечення тренажеру з теми: «Побудова математичних моделей лінійних задач планування виробництва» дистанційного навчального курсу «Сучасні методи оптимізації та їх програмування», Григор'єв, Владислав Віталійович [Електронний ресурс]

Режим доступу до ресурсу: http://dspace.puet.edu.ua/handle/123456789/8992

4. Пояснительная записка к бакалаврской работе на тему: Тренажер по теме: «Нахождение коэффициентов компетентности экспертов», Мамедов, Ахмед Али оглы [Електронний ресурс]

Режим доступу до ресурсу: http://dspace.puet.edu.ua/handle/123456789/8997

5. Гмиза Б. Ю. Тренажер з теми «Побудова блок-схем алгоритмів циклічної структури на прикладі циклу for» дистанційного навчального курсу «Інформатика» та розробка його програмного забезпечення / Б. Ю. Гмиза, Ол-ра О. Ємець // Комп'ютерні науки і прикладна математика (КНіПМ-2019):

матеріали наук.-практ. семінару. Випуск 3. / За ред. Ємця О. О. – Полтава: Кафедра ММСІ ПУЕТ, 2019. – С. 38-39. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/7036.

6. Мордасова І. В. Тренажер з теми «Побудова блок-схем алгоритмів розгалуженої структури» дистанційного навчального курсу «Інформатика» та розробка його програмного забезпечення / І. В. Мордасова, Ол-ра О. Ємець // Комп'ютерні науки і прикладна математика (КНіПМ-2019): матеріали наук.-практ. семінару. Випуск 3. / За ред. Ємця О. О. – Полтава: Кафедра ММСІ ПУЕТ, 2019. – С. 35-37. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/7037.

7. Шакуро В. Є. Розробка програмного забезпечення з теми «Побудова блок-схема алгоритмів лінійної структури» дистанційного курсу «Інформатика»/ В. Є. Шакуро, Ол-ра О. Ємець // Комп'ютерні науки і прикладна математика (КНіПМ-2019): матеріали наук.-практ. семінару. Випуск 3. / За ред. Ємця О. О. – Полтава: Кафедра ММСІ ПУЕТ, 2019. – С. 40-42. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/7038.

8. Сузанська А. О. Тренажер «Побудова блок-схем алгоритмів розгалуженої структури» / А. О. Сузанська, Є. М. Ємець, О. О. Ємець // Комп'ютерні науки і прикладна математика (КНіПМ-2020): матеріали наук.-практ. семінару. Випуск 5. / За ред. Ємця О. О. – Полтава: Кафедра ММСІ ПУЕТ, 2020. – С. 56-62. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/8906.

9. Ємець О.О. Дистанційний курс Полтавського університету економіки і торгівлі «Інформатика. Частина 1» для студентів спеціальності «Комп'ютерні науки» / О.О. Ємець. – [Електронний ресурс].

10. Ємець О.О. Дистанційний курс Полтавського університету економіки і торгівлі «Програмування ІІ. Частина 1» для студентів спеціальності «Комп'ютерні науки» / О.О. Ємець. – [Електронний ресурс].

11. Ємець О. О. Методичні рекомендації до виконання бакалаврської роботи для студентів спеціальності 122 «Комп`ютерні науки та інформаційні

технології» освітня програма «Комп`ютерні науки» галузь знань – 12 «Інформаційні технології» / О. О. Ємець. – Полтава: ПУЕТ, 2017. – 71 с.

12. Побудова блок-схем [Електронний ресурс]

Режим доступу до ресурсу: https://nmetau.edu.ua/file/011.pdf.- nmetau.edu.ua

13. Циклічні алгоритми [Електронний ресурс]

Режим доступу до ресурсу: https://studfile.net/preview/3740818/page:5/.- studfile.net ua

14. Блок-схеми алгоритмів [Електронний ресурс]

Режим доступу до ресурсу: http://programming.in.ua/programming/basisprogramming/141-graph-algorithm.html.- programming.in.ua

15. Цикл з параметром for [Електронний ресурс]

Режим доступу до ресурсу: http://www.tvd-home.ru/prog/3_1.- tvd-home.ru ua

ДОДАТОК А

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.UI;
using UnityEngine;

public class Inputs : MonoBehaviour
{
    public GameObject Input;
    public GameObject AfterInput;
    public InputField inputedTxt;
    public Text ShowInpTxt;



    public void Next()
    {
        Input.SetActive(false);
        AfterInput.SetActive(true);

        ShowInpTxt.text = inputedTxt.text;
    }
}

public class QuitButton : MonoBehaviour
{
    public void QuitGame()
    {
        Debug.Log ("QUIT!!");
        Application.Quit();
```

```
    }

void Start()
    {
        Button btn = nxtButton.GetComponent<Button>();
        btn.onClick.AddListener(TaskOnClick);
    }

void TaskOnClick()
    {
        Theme1.SetActive(false);
        Theme2.SetActive(true);

        Debug.Log("You touched this button.");
    }

void Start()
    {
        Button btn = menuButton.GetComponent<Button>();
        btn.onClick.AddListener(TaskOnClick);
    }

void TaskOnClick()
    {
        Theme6.SetActive(false);
        MainMenu.SetActive(true);

        Debug.Log("You touched this button.");
    }
public void Next()
```

```
    {
        Input.SetActive(false);
        AfterInput.SetActive(true);

        ShowInpTxt.text = inputedTxt.text;
    }
}
```