

**ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»**

**ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ ТА
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**ФОРМА НАВЧАННЯ ЗАОЧНА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ
ІНФОРМАТИКИ**

Допускається до захисту

Завідувач кафедри _____ О.О. Ємець
(підпис)

« _____ » _____ 2021 р.

**ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО БАКАЛАВРСЬКОЇ РОБОТИ
на тему**

**РОЗРОБКА ТРЕНАЖЕРУ З ТЕМИ «ЗАДАННЯ МОВ РЕГУЛЯРНИМИ
ВИРАЗАМИ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ
ПРОГРАМУВАННЯ»**

зі спеціальності 122 «Комп'ютерні науки»

Виконавець роботи Ярош Андрій Віталійович

_____ « _____ » _____ 2021р.
(підпис)

Науковий керівник к.ф.-м.н., доц. Черненко Оксана Олексіївна

_____ « _____ » _____ 2021р.
(підпис)

ПОЛТАВА 2021р.

**ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»**

ЗАТВЕРДЖУЮ
Завідувач кафедри _____ **О.О. Ємець**
(підпис)
«_1_» вересня 2021 р.

**ЗАВДАННЯ ТА КАЛЕНДАРНИЙ ГРАФІК
ВИКОНАННЯ ДИПЛОМНОЇ РОБОТИ**

**Студент(ка) спеціальності 122 «Комп'ютерні науки»
Прізвище, ім'я, по батькові: Ярош Андрій Віталійович**

**1. Тема Розробка тренажеру з теми «Задання мов регулярними виразами»
дистанційного навчального курсу «Теорія програмування»**

затверджена наказом ректора № ___-Н від «___» _____ 2021 р.

Термін подання студентом дипломної роботи _____ «___» _____ 2021 р.

2. Вихідні дані до дипломної роботи: методичні матеріали за темою роботи; матеріали дистанційного навчального курсу «Теорія програмування» та стандарти.

Приклад 1. Записати регулярний вираз для мови {a, aa, aaa, ...};

Приклад 2. Записати регулярний вираз для мови {a, aa, aaa, ...};

Приклад 3. Записати регулярний вираз для мови {a, b, c};

Приклад 4. Записати регулярний вираз для мови {ac, abc, abbc, ...};

Приклад 5. Записати регулярний вираз для мови {abc, abbc, ...};

Приклад 6. Записати регулярний вираз для мови {abd, acd};

Приклад 7. Записати регулярний вираз для мови {abcd, abbc, ...}.

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

ВСТУП

1 ПОСТАНОВКА ЗАДАЧІ

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Позитивні сторони розглянутих тренажерів

2.2 Негативні сторони розглянутих тренажерів

3 ТЕОРЕТИЧНА ЧАСТИНА

3.1 Загальні відомості

3.2.1 Мови

3.2.2 Регулярні вирази

3.2 Алгоритм роботи тренажера

3.3 Блок-схема програми-тренажера

4 ПРАКТИЧНА ЧАСТИНА

4.1 Опис створення тренажера

4.2 Інструкція по використанню тренажера
ВИСНОВКИ
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ
ДОДАТОК А

4. Перелік графічного матеріалу (з точним визначенням кількості блок-схем, іншого графічного матеріалу)
Блок-схеми алгоритму, рисунки(в необхідній кількості)

5. Консультанти розділів дипломної роботи

Розділ	Прізвище, ініціали, посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ	Черненко О.О.		
1. Постановка задачі	Черненко О.О.		
2. Інформаційний огляд	Черненко О.О.		
3. Теоретична частина	Черненко О.О.		
4. Практична частина	Черненко О.О.		

6. Календарний графік виконання дипломної роботи

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

Дата видачі завдання «1» вересня 2020 р.

Студент(ка) _____
(підпис)

Ярош Андрій Віталійович

Науковий керівник _____
(підпис)

к.ф.-м.н., доц. Черненко О.О.

Результати захисту дипломної 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: 44 стор., в т.ч. основна частина 40 стор., джерел - 11.

Предмет розробки – тренажер з теми «Задання мов регулярними виразами».

Мета роботи – створення програмного забезпечення тренажеру з теми «Задання мов регулярними виразами» дистанційного навчального курсу «Теорія програмування».

Методи, які були використані для розв’язування задачі –

Тренажер було створено за допомогою MS Visual Studio як середовища для написання коду та платформи Unity для програмної реалізації.

Реалізовано перевірку валідності введеної відповіді, видачу підказки в повідомленні про неправильну відповідь та можливість повернутися в меню для повтору роботи після завершення роботи з тренажером.

Ключові слова: ТРЕНАЖЕР, РЕГУЛЯРНІ ВИРАЗИ, ТЕОРІЯ ПРОГРАМУВАННЯ.

ЗМІСТ

ВСТУП.....	5
1 ПОСТАНОВКА ЗАДАЧІ.....	7
2 ІНФОРМАЦІЙНИЙ ОГЛЯД.....	8
2.1 Позитивні сторони розглянутих тренажерів.....	8
2.2 Негативні сторони розглянутих тренажерів.....	11
3 ТЕОРЕТИЧНА ЧАСТИНА	12
3.1 Загальні відомості	12
3.2.1 Мови	12
3.2.2 Регулярні вирази	13
3.2 Алгоритм роботи тренажера.....	16
3.3 Блок-схема програми-тренажера.....	19
4 ПРАКТИЧНА ЧАСТИНА	21
4.1 Опис створення тренажера.....	21
4.2 Інструкція по використанню тренажера.....	24
ВИСНОВКИ.....	40
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	41
ДОДАТОК А.....	43

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, символів, скорочень
<i>ДК</i>	Дистанційний курс.
<i>ДН</i>	Дистанційне навчання.
<i>ТП</i>	Теорія програмування
<i>С#</i>	Мова програмування.
<i>Unity</i>	Платформа для програмної реалізації.
<i>UI</i>	(user interface) інтерфейс користувача

ВСТУП

В час розвитку комп'ютерних технологій дистанційне навчання займає важливу роль. Такий вид навчання дає можливість студентам отримувати якісні знання за допомогою сучасних інформаційних, що дозволяють навчатись на відстані без безпосереднього, особистого контакту між викладачем і студентом.

Електронні тренажери призначені для навчання, а також закріплення умінь та навичок розв'язування конкретних задач в відповідній предметній області. Вважається актуальним створення такого засобу дистанційного навчання.

Мета роботи – створення програмного забезпечення тренажеру з теми «Задання мов регулярними виразами» дистанційного навчального курсу «Теорія програмування».

Предмет розробки – тренажер з теми «Задання мов регулярними виразами».

Методи, які були використані для розв'язування задачі – Тренажер було створено за допомогою MS Visual Studio як середовища для написання коду та платформи Unity для програмної реалізації.

Дипломна робота складається з наступних складових частин:

- титульний аркуш;
- завдання до дипломної роботи;
- реферат;
- зміст;
- список умовних позначень та скорочень;
- вступ;
- теоретична частина;
- практична частина;
- висновки;
- список використаних джерел;

Обсяг пояснювальної записки: 44 стор., в т.ч. основна частина 40 стор., джерел - 11.

1 ПОСТАНОВКА ЗАДАЧІ

Завданням та метою дипломної роботи є створення тренажеру дистанційного навчання з теми «Задання мов регулярними виразами».

Тренажер було створено за допомогою MS Visual Studio як середовища для написання коду та платформи Unity для програмної реалізації.

Під час роботи над постановкою задачі було виділено наступні завдання для роботи над бакалаврською роботою:

1. Надати можливість навчання без доступу до мережі інтернет;
2. Наявність в тренажері довідкового матеріалу з теми;
3. Наявність в тренажері практичних завдань у вигляді тестів;
4. Наявність перевірки валідності введеної відповіді та показ відповідного повідомлення;
5. Зручний та зрозумілий інтерфейс;
6. Можливість повторити роботу з тренажером через відповідну кнопку у кінці навчальних матеріалів.
7. Пошук теоретичної інформації;
8. Пошук та створення завдань для тренажеру;
9. Розробка алгоритму роботи тренажера;
10. Розробка блок-схем для роботи з тренажером;
11. Програмна реалізація програми-тренажера;
12. Дотримання вимог оформлення.

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

Для інформаційного огляду було відібрано два тренажери, а саме:

1. Тренажер Скоромінський М.В. з теми «Контекстовільні граматики» [2],
2. Тренажер Костромін І.І. з теми «Математичні основи теорії алгоритмів» [3].

2.1 Позитивні сторони розглянутих тренажерів

1. Після запуску тренажеру Скоромінський М.В. з теми «Контекстовільні граматики» користувач переходить до вікна з головною сторінкою тренажера на якій зображені тема тренажера, виконавець та науковий керівник.

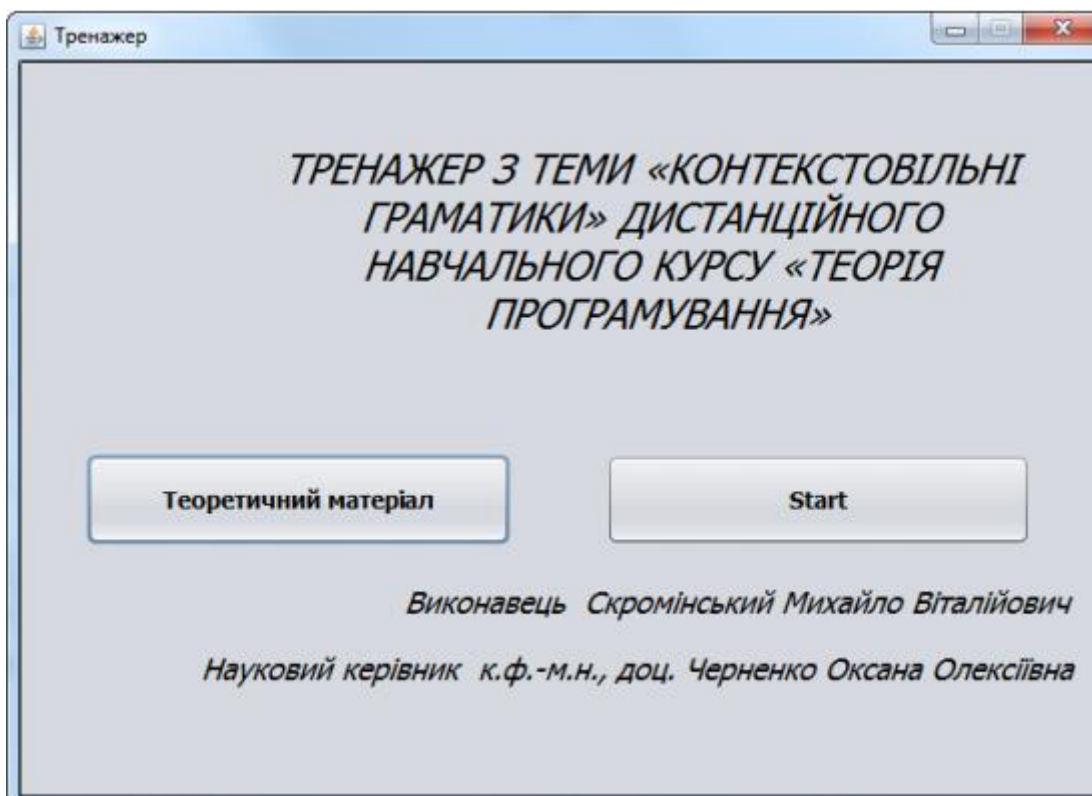


Рисунок 2.1 – Головне меню тренажера з теми «Контекстовільні граматики»

Після вибору теоретичного матеріалу та натиснення на відповідну кнопку користувач переходить до вікна з теоретичним матеріалом.

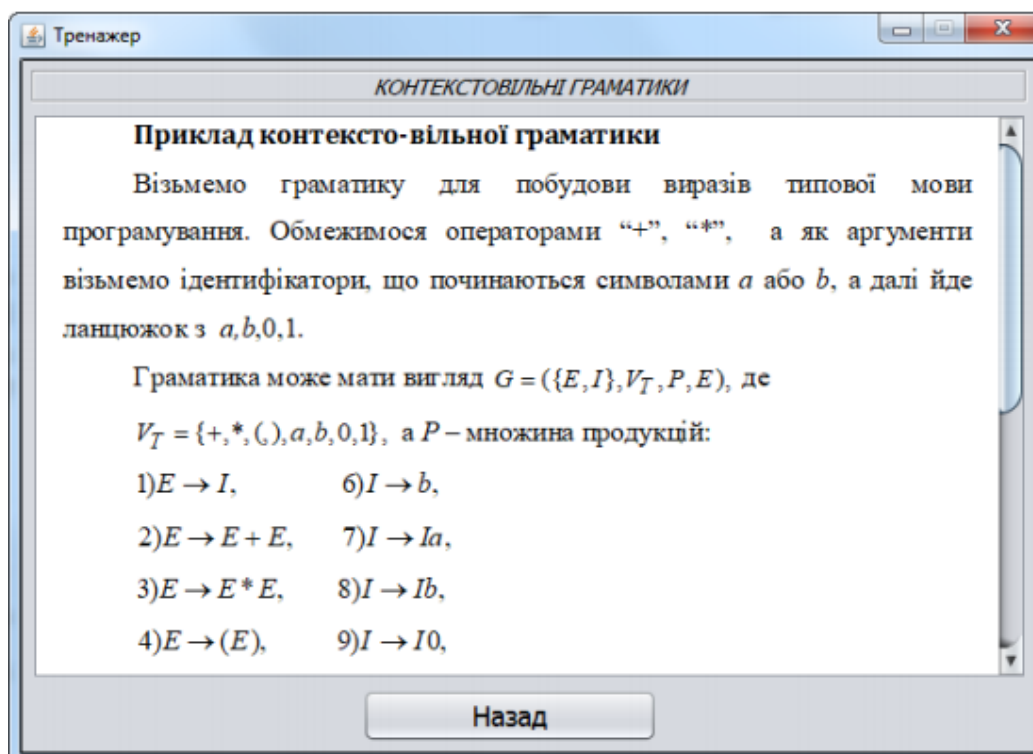


Рисунок 2.2 – Вікно з теоретичним матеріалом тренажера з теми «Контекстовільні граматики»

Після вибору практичної частини та натисканні на відповідну кнопку користувач переходить до практичних завдань у вигляді тестів.

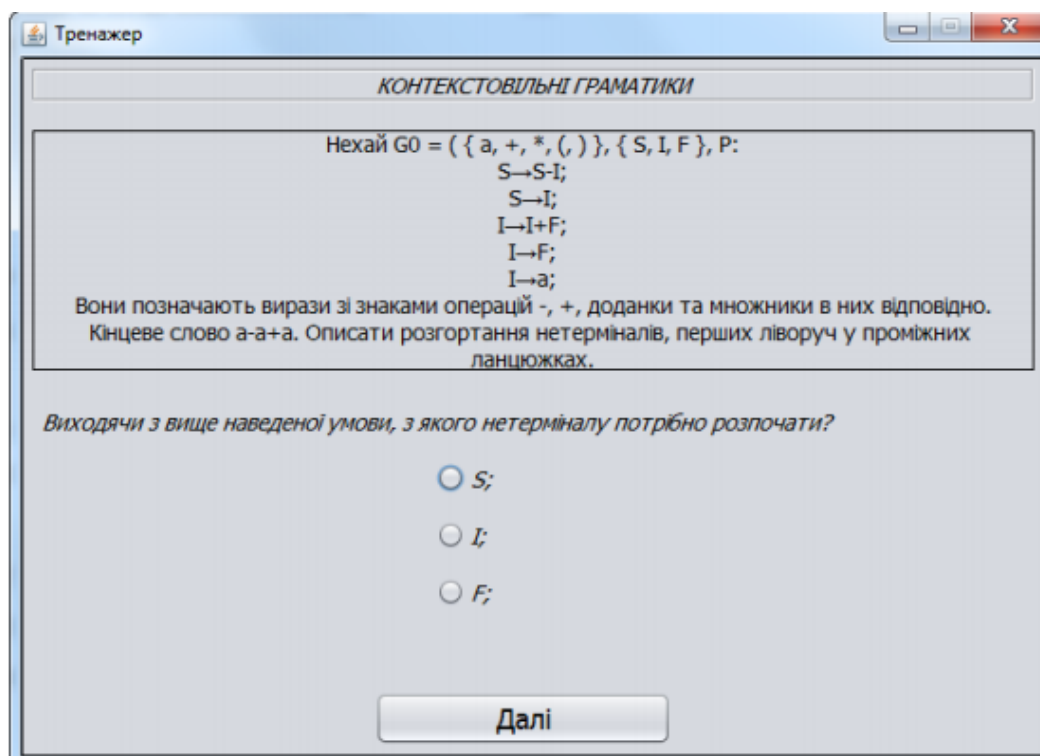


Рисунок 2.3 – Вікна з практичними завданнями тренажера з теми «Контекстовільні граматики»

При виборі неправильної відповіді користувач отримує відповідне повідомлення з підказкою, при виборі правильної відповіді відбувається продовження роботи з практичними завданнями в тренажері.

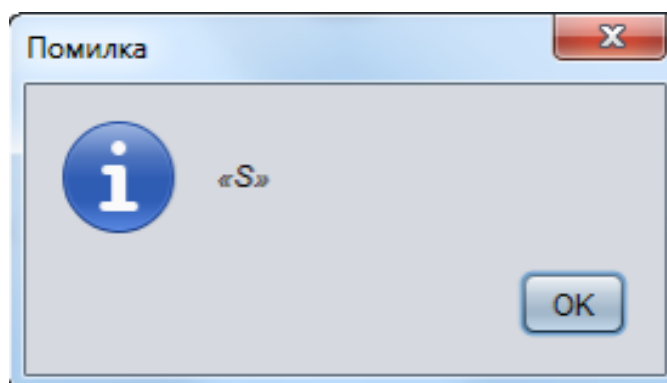


Рисунок 2.4 – Повідомлення про помилку тренажера з теми «Контекстовільні граматики»

Після закінчення роботи з практичними завданнями в тренажері користувач отримує доступ до вікна з повідомленням про завершення роботи та отримує можливість пройти роботу знову.

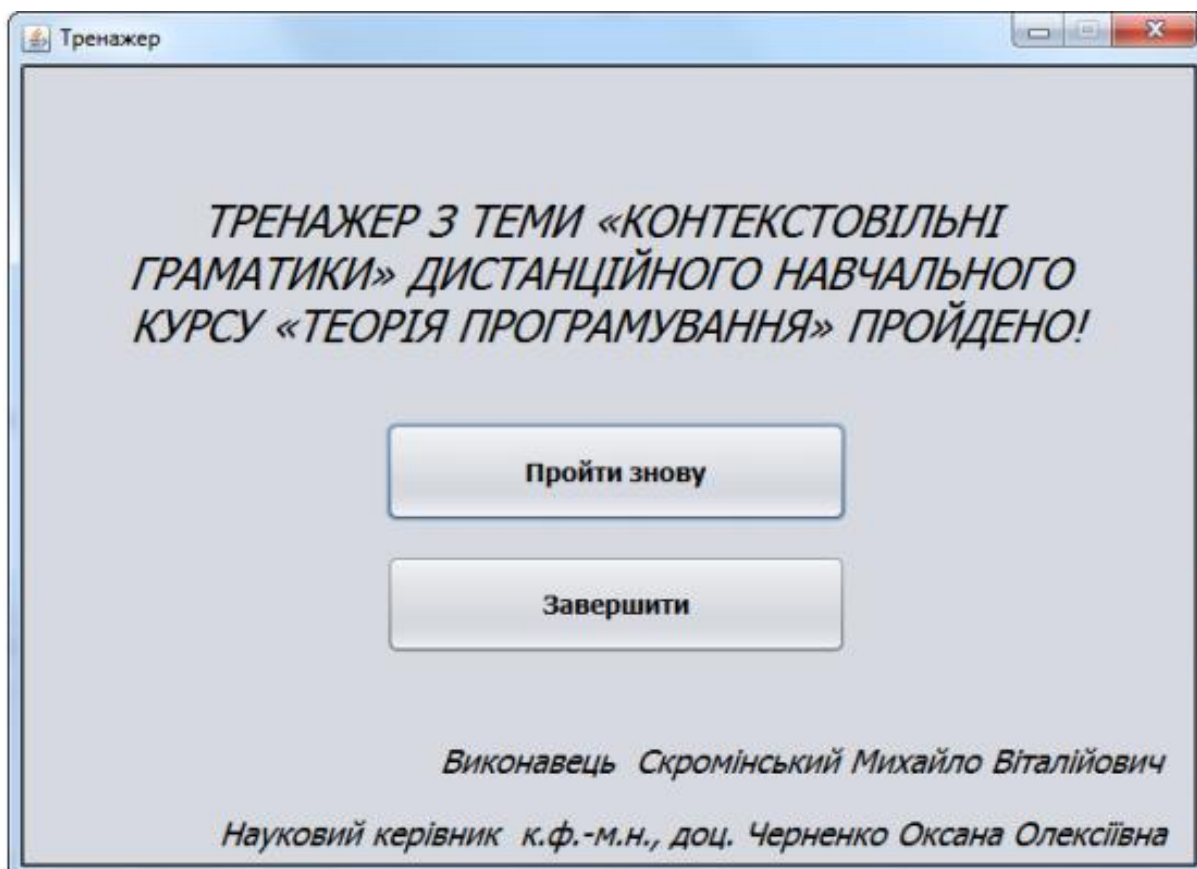


Рисунок 2.5 – Вікна з фіналом тренажера з теми «Контекстовільні граматики»

2. Наступний розглянутий тренажер має дуже схожу на попередній структуру. Після запуску тренажера Костромін І.І. з теми «Математичні основи теорії алгоритмів» користувач отримує доступ до стартового вікна з темою роботи де йому надається вибір відкрити теорію, або розпочати проходження тренажеру.

Під час роботи з практичними завданнями в тренажері після вибору правильної відповіді відбувається перехід на наступний крок, а після вибору неправильної відповіді видається повідомлення з правильною.

2.2 Негативні сторони розглянутих тренажерів

Негативними сторонами тренажеру Скромінський М.В. з теми «Контекстовільні граматики» є:

1. Кнопку Старт на стартовому вікні тренажера слід замінити на кнопку Перехід до навчальних матеріалів чи Практична частина;

2. Неможливо повернутися до стартового вікна тренажеру під час роботи з практичною частиною;

Негативними сторонами тренажеру Костромін І.І. з теми «Математичні основи теорії алгоритмів» є:

1. Тренажер запускається з помилкою;

2. Неможливо повторити роботу з тренажером після завершення роботи з ним.

3 ТЕОРЕТИЧНА ЧАСТИНА

3.1 Загальні відомості

В програмуванні, регулярний вираз — це рядок, що описує або збігається з множиною рядків, відповідно до набору спеціальних синтаксичних правил. Вони використовуються в багатьох текстових редакторах та допоміжних інструментах для пошуку та зміни тексту на основі заданих шаблонів. Багато мов програмування підтримують регулярні вирази для роботи з рядками. Наприклад, Perl та Tcl мають потужний механізм для роботи, вбудований безпосередньо в їхній синтаксис. Завдяки набору утиліт (разом з редактором sed та фільтром grep), що входили до складу дистрибутивів UNIX, регулярні вирази стали відомими та поширеними.[4]

3.2.1 Мови

Алфавіт – це скінченна множина символів. Позначатимемо його X .

Приклади

- Множина $\{0,1\}$ являє собою бінарний алфавіт.
- ASCII і Unicode є прикладами комп'ютерних алфавітів.

Рядком у даному алфавіті називають скінченну послідовність символів з цього алфавіту. У теорії мов рядок ще називають ланцюжком, реченням, словом.

Символ ϵ позначає порожній рядок. Порожній рядок має нульову довжину.

Терміни “підрядок” і “підпослідовність” будемо розрізняти. Підрядок одержують видаленням початку і (або) кінця з рядка, підпослідовність – видаленням декількох символів, не обов'язково послідовних.

Множина всіх скінченних слів у алфавіті X позначається X^* . Зауважимо, що вона нескінченна. Вона містить *порожнє слово* – послідовність довжиною 0, позначену буквою ϵ . Множину $X^* \setminus \{\epsilon\}$ позначимо X^+ , а слово вигляду $ww..w$, де слово w із X^+ записано n разів – wn . Вважатимемо, що $w^0 = \epsilon$.

Довільна підмножина множини X^* називається *формальною мовою*. Далі вона буде називатися просто *мовою*. Мова – це довільна множина рядків у деякому алфавіті.

Приклади.

1. Множина всіх слів у алфавіті $\{a\}$ позначається $\{a\}^* = \{\epsilon, a, aa, aaa, \dots\} = \{a^n | n \geq 0\}$. $\{a^n | n \text{—непарне}\}$ позначає множину, або мову слів непарної довжини в алфавіті $\{a\}$; обидві мови нескінченні.

2. Ідентифікатор є послідовністю букв і цифр, що починається буквою. Множина всіх ідентифікаторів у алфавіті $X = \{a, b, 1\}$ нескінченна. Якщо записати їх за зростанням довжини, то початок буде таким: $\{a, b, a1, aa, ab, b1, ba, bb, \dots\}$.

Задача перевірки, чи належить слово w мові L , називається *задачею належності*, або *проблемою слів*. Як правило, множина L задається певним *скінченим описом*, що визначає не тільки її саму, а й структуру її елементів.

Задача належності розв'язується найчастіше шляхом перевірки, чи має слово відповідну структуру, тобто шляхом *синтаксичного аналізу*, або *розпізнавання*. Наприклад, структура всіх можливих синтаксично правильних Паскаль-програм визначається скінченною та відносно невеликою сукупністю БНФ. Саме на її основі будуються *синтаксичні аналізатори* в трансляторах, тобто програми аналізу синтаксичної правильності вхідних програм.

Формальні мови розглядатимуться далі як мови, задані саме *скінченим описом*. Отже, головним у вивченні *формальних мов стає засіб їх задання*. [5]

3.2.2 Регулярні вирази

Модуль `re` з'явився в Python 1.5 і працював з регулярними виразами стилю Perl. Старіші версії Пайтона поставлялись з модулем `regex`, який реалізовував регулярні вирази в стилі Emacs. Модуль `regex` був вилучений з Python 2.5.

Регулярні вирази (часті скорочення: RE, regex) це спеціальна малесенька мова програмування вбудована в Python за допомогою модуля re. За допомогою цієї мови можна описувати множини слів (рядків), які відповідають шаблону (регулярному виразу). Множинами можуть бути всі правильні адреси e-mail, команди TeX, чи будь-що що ви захочете. Потім можна отримувати відповіді на запитання "Чи відповідає цей рядок шаблону?", або "Чи зустрічається у цьому рядку послідовність символів що відповідає шаблону?". Також можна використовувати регулярні вирази, щоб робити заміни в рядку, чи розділяти її на складові різними способами.

Регулярні вирази компілюються в послідовність байткодів, які потім виконуються підпрограмою написаною на C.

Через те, що регулярні вирази - доволі мала та обмежена мова, тому не всі завдання з обробки рядків можна виконати з їх допомогою. Також є задачі, які робляться з регулярними виразами, але вони виходять занадто складними, і ефективніше написати код обробки рядка на Python.

Вивчимо найпростіші регулярні вирази. Найпростіша задача, яку вони виконують – пошук відповідностей шаблону.

Нехай A – алфавіт.

Клас регулярних виразів визначається такими правилами:

1. ϵ є регулярним виразом;
2. якщо $a \in A$, то a – регулярний вираз;
3. якщо w – регулярний вираз, то w^* і (w) – регулярні вирази;

якщо w_1 і w_2 – регулярні вирази, то w_1w_2 , w_1/w_2 – регулярні вирази. У виразі w_1w_2 використовується операція катенації, у виразі w_1/w_2 – операція об'єднання.

Приклади.

$$L(a^*) = \{a, aa, aaa, \dots\}.$$

$$L(a/b/c) = \{a, b, c\}.$$

$$L(ab^*c) = \{ac, abc, abbc, \dots\}.$$

$$L(ab^+c) = \{abc, abbc, \dots\}.$$

$$L(a(b/c)d) = \{abd, acd\}.$$

Зайві дужки (необов'язково усі) можуть бути видалені, якщо прийняти такі співвідношення: операція ітерація “*” має вищий пріоритет, катенація – другий за значенням і “|” – нижчий.

Приклад виконання завдання. Написати регулярний вираз для мови, яка являє собою множину ланцюжків у алфавіті $\{a,b,c\}$, що містять хоча б один символ a і хоча б один символ b .

Потрібний вираз може бути представлений, наприклад, у такій формі:

$$((a/b/c)^*a(a/b/c)^*b(a/b/c)^*) / ((a/b/c)^*b(a/b/c)^*a(a/b/c)^*) [5].$$

3.2 Алгоритм роботи тренажера

Крок 1. Користувач запускає тренажер через виконуваний файл з назвою теми.

Крок 2. Після запуску тренажера користувач переходить до вікна з вибором якості графіки та розширення екрану.

Крок 3. Користувач отримує доступ до вікна з темою та виконавцем роботи. Після натиснення на кнопку «Почати» користувач переходить до наступного вікна.

Крок 4. На наступному вікні користувач бачить перед собою інструкцію для використання тренажера:

«Під час роботи з практичними завданнями необхідно обрати одну правильну відповідь для продовження роботи. У разі вибору неправильної відповіді ви отримаєте відповідне повідомлення та повинні будете змінити свою відповідь. Навігація в тренажері реалізована через кнопки «Далі» та «Назад», що дозволяють орієнтуватися в навчальних матеріалах. Після закінчення роботи з тренажером ви зможете повторити роботу з ним через відповідну кнопку. Успіхів!»

Крок 5. Після ознайомлення з інструкцією користувач переходить до практичних завдань.

Крок 6. Отримання практичних завдань у вигляді тестів.

«Завдання 1. Регулярний вираз (в програмуванні) це?

1. Стандартна мова для задання зразків пошуку;
2. Події, представимі в скінченних автоматах;
3. Мова програмування;

Правильна відповідь – 1.»

Крок 7. Отримання практичних завдань у вигляді тестів.

«Завдання 2. Скільки разів повинен виконатися вислів A при записі регулярного виразу $L(A^+)$?

1. Один раз;
2. Один або декілька разів;

3. Більше одного разу;

Правильна відповідь – 2.»

Крок 8. Отримання практичних завдань у вигляді тестів.

«Завдання 3. Скільки разів повинен виконатися вислів A при записі регулярного виразу $L(A|)$?

1. Вислів A є обов'язковим та повинен виконатися принаймні один раз;

2. Вислів A є необов'язковим та може виконатися принаймні один раз;

3. Вислів A є необов'язковим та може виконатися максимум один раз;

Правильна відповідь – 1.»

Крок 9. Отримання практичних завдань у вигляді тестів.

«Завдання 4. Скільки разів повинен виконатися вислів A при записі регулярного виразу $L(A^*)$?

1. Вираз повинен збігтися нуль разів;

2. Вираз повинен збігтися нуль чи більше разів;

3. Вираз повинен збігтися один чи більше разів;

Правильна відповідь – 2.»

Крок 10. Отримання практичних завдань у вигляді тестів.

«Завдання 5. Що є результатом виконання наступного виразу - $L(a^*b)$?

1. $\{ab, aab, aaab, \dots\}$;

2. $\{ab, \dots\}$;

3. $\{ab, aa, \dots\}$;

Правильна відповідь – 1.»

Крок 11. Отримання практичних завдань у вигляді тестів.

«Завдання 6. Що є результатом виконання наступного виразу - $L(a|b|c)$?

1. $\{ \}$;

2. $\{aa, bbb, cccc\}$;

3. $\{a, b, c\}$;

Правильна відповідь – 3.»

Крок 12. Отримання практичних завдань у вигляді тестів.

«Завдання 7. Що є результатом виконання наступного виразу - $L(ab^*c)$?

1. {a, ab, abc, ...};
2. {ac, abc, abbc, ...};
3. {abc, abbc, ...};

Правильна відповідь – 2.»

Крок 13. Отримання практичних завдань у вигляді тестів.

«Завдання 8. Що є результатом виконання наступного виразу - $L(ab+c)$?

1. { abc, abbc, ...};
2. { abc, abbbc, ...};
3. { b, bb, ...};

Правильна відповідь – 1.»

Крок 14. Отримання практичних завдань у вигляді тестів.

«Завдання 9. Що є результатом виконання наступного виразу - $L(a(b|c)d)$?

1. {abc, abd};
2. {abd, acd};
3. {ad, ab, ac};

Правильна відповідь – 2.»

Крок 15. Отримання практичних завдань у вигляді тестів.

«Завдання 10. Що є результатом виконання наступного виразу - $L(a(b+c)d)$?

1. {ab, abb, ...};
2. {abcd, abbcd, ...};
3. {abcd, abbc, ...};

Правильна відповідь – 3.»

Крок 16. Користувач отримує доступ до вікна з повідомленням «Ви успішно завершили роботу з тренажером! Для повтору роботи натисніть на кнопку «Повтор» нижче.»

Крок 17. Після натиснення на кнопку «Повтор» користувач переходить до початкового меню тренажеру та може повторити роботу з тренажером.

3.3 Блок-схема програми-тренажера

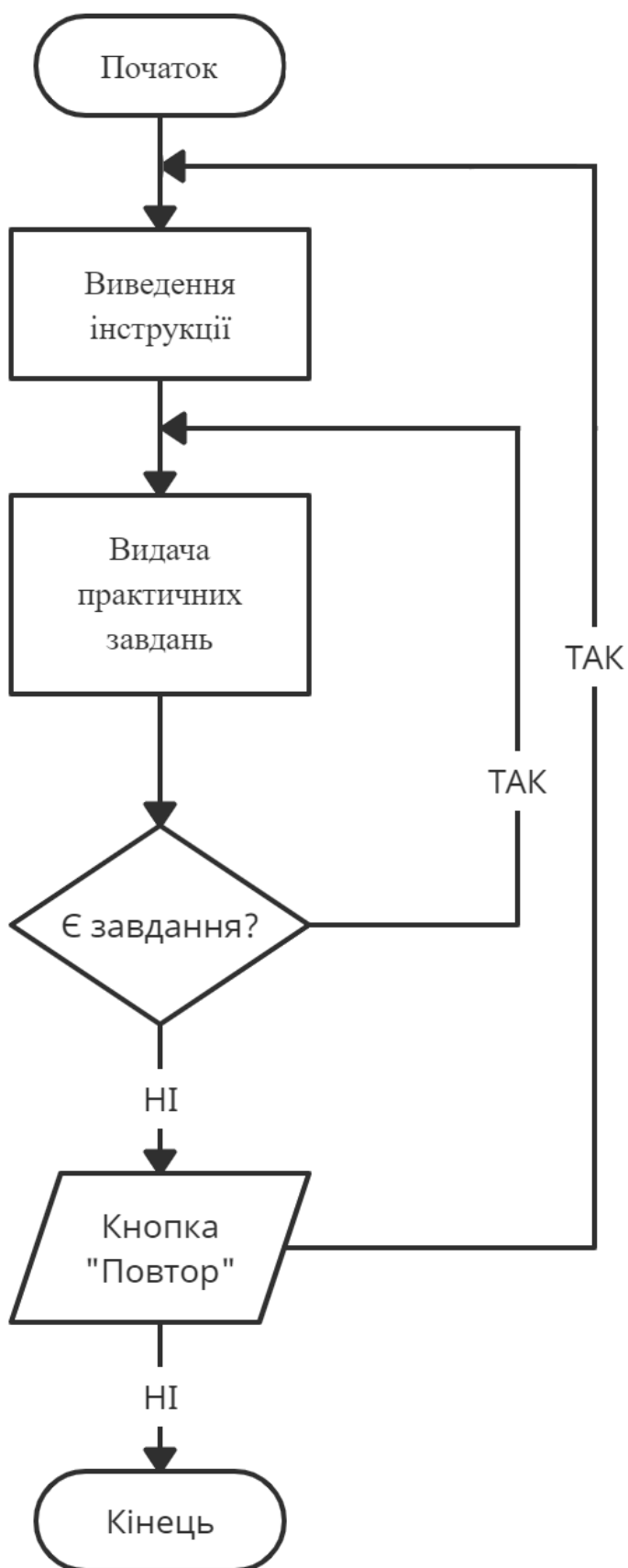


Рисунок 3.4 – Блок-схема програми-тренажера

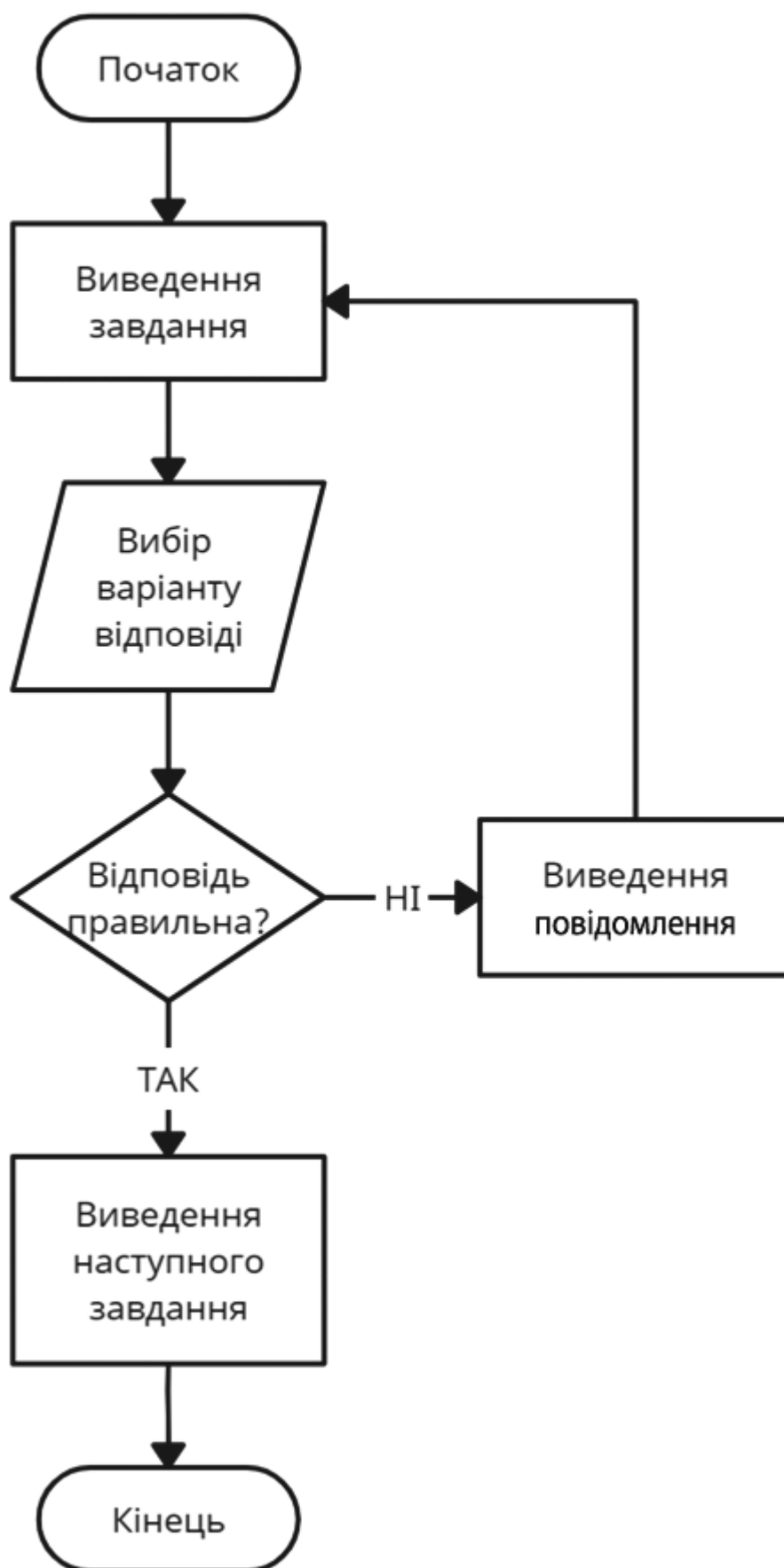


Рисунок 3.5 – Блок-схема роботи з практичними завданнями в тренажері

4 ПРАКТИЧНА ЧАСТИНА

4.1 Опис створення тренажера

В новому проєкті Unity необхідно створити сцену та задати ієрархію в ній.

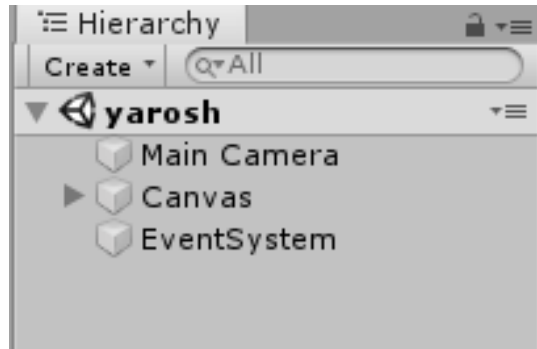


Рисунок 4.1 – Створення сцени та ієрархії

Всі елементи тренажера було створено через UI елементи створені за допомогою інтерфейсу в Unity.

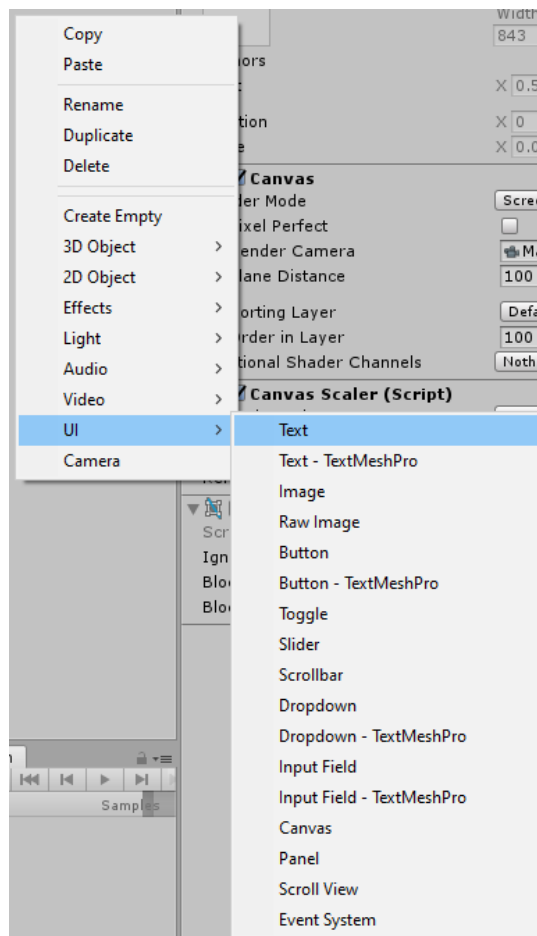


Рисунок 4.2 – Додання в проєкт UI елементів

Всі дії в проєкті виконуються за допомогою скриптів для кнопок. Скрипти для кнопок працюють на булевих функціях, що вмикають чи вмикають певні елементи інтерфейсу користувача.

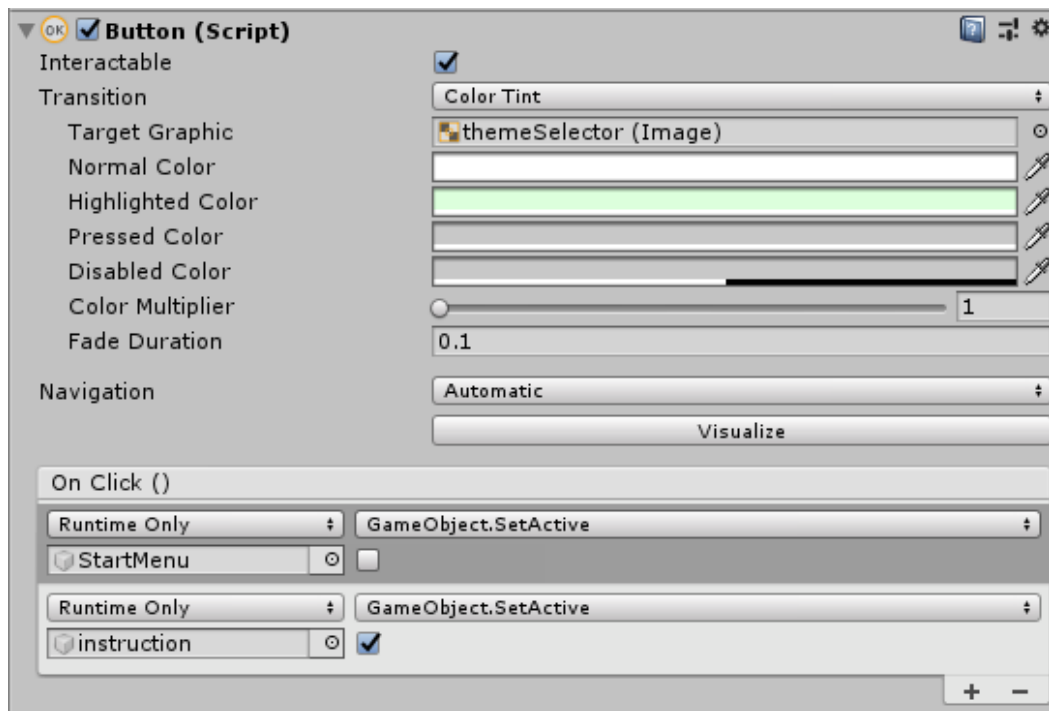


Рисунок 4.3 – Скрипт для кнопки «Почати» в головному меню

Всі кнопки навігації працюють таким самим чином. Кнопки з варіантами відповіді працюють так, як зображено на Рисунку 4.4 та 4.5.

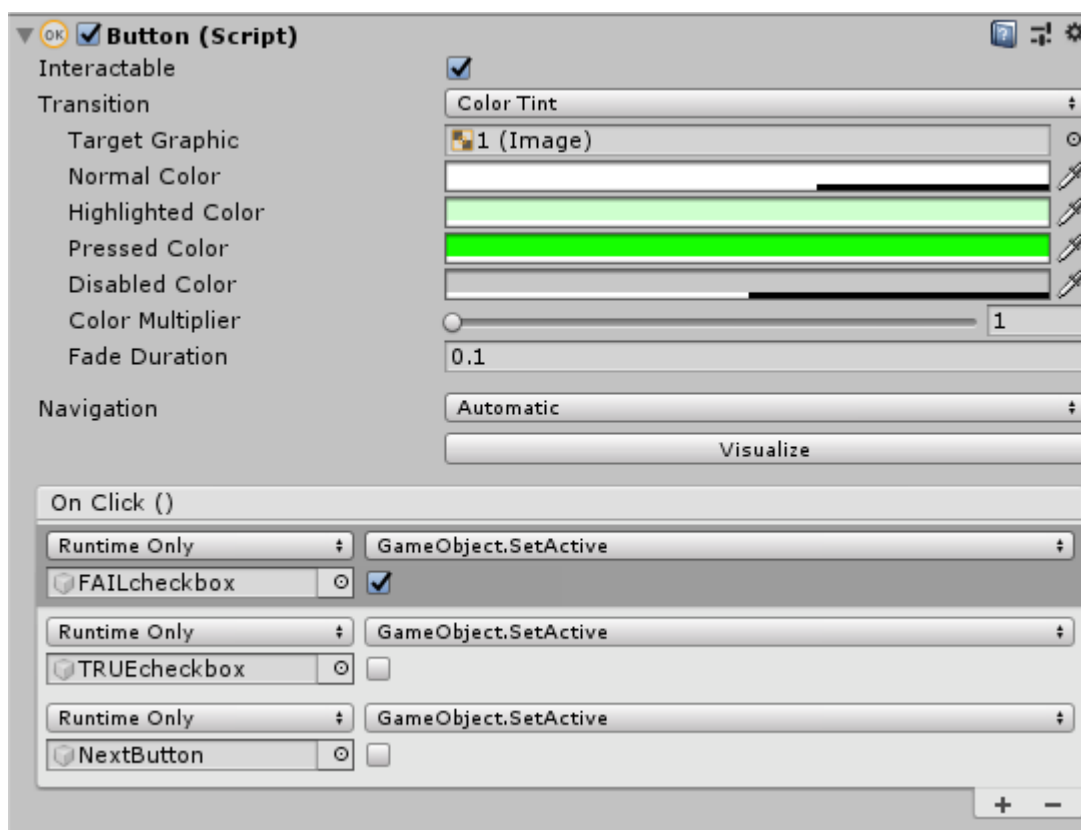


Рисунок 4.4 – Скрипт для кнопки з неправильною відповіддю

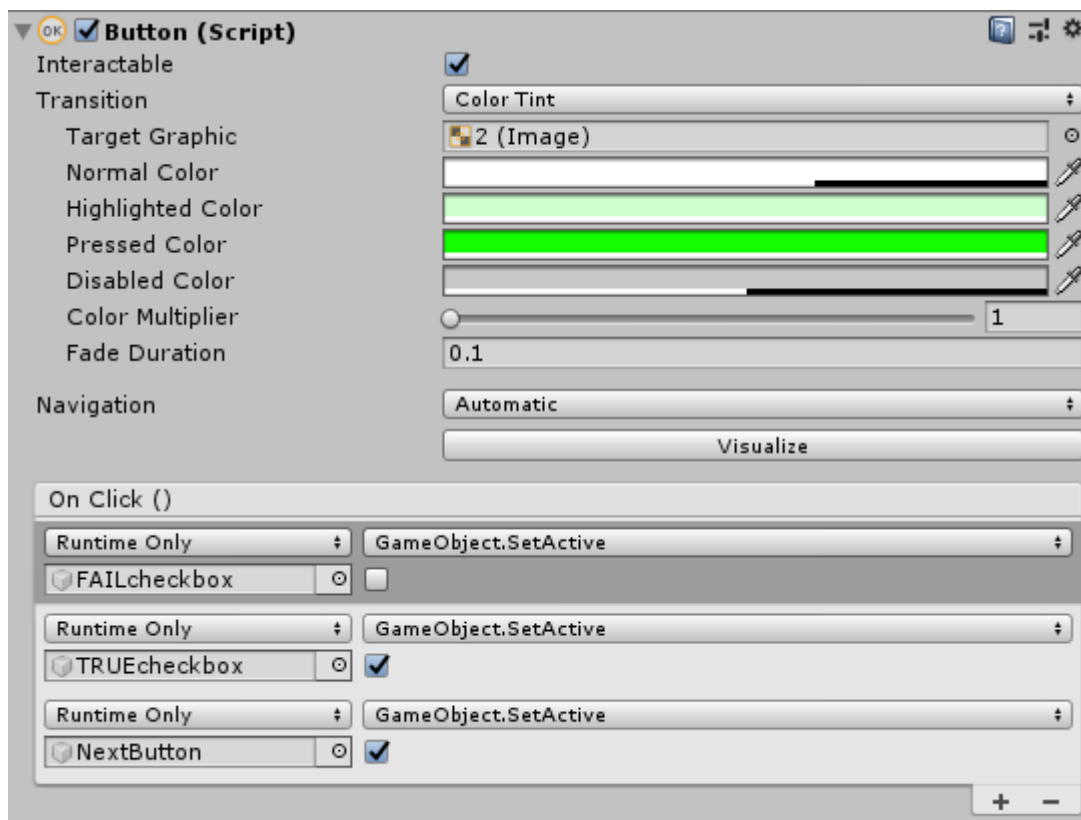


Рисунок 4.5 – Скрипт для кнопки з правильною відповіддю

4.2 Інструкція по використанню тренажера

Після запуску тренажера користувача переходить до вікна з вибором конфігурації тренажеру. На цьому вікні доступний вибір розширення екрану, якості зображення та монітору для виводу зображення.

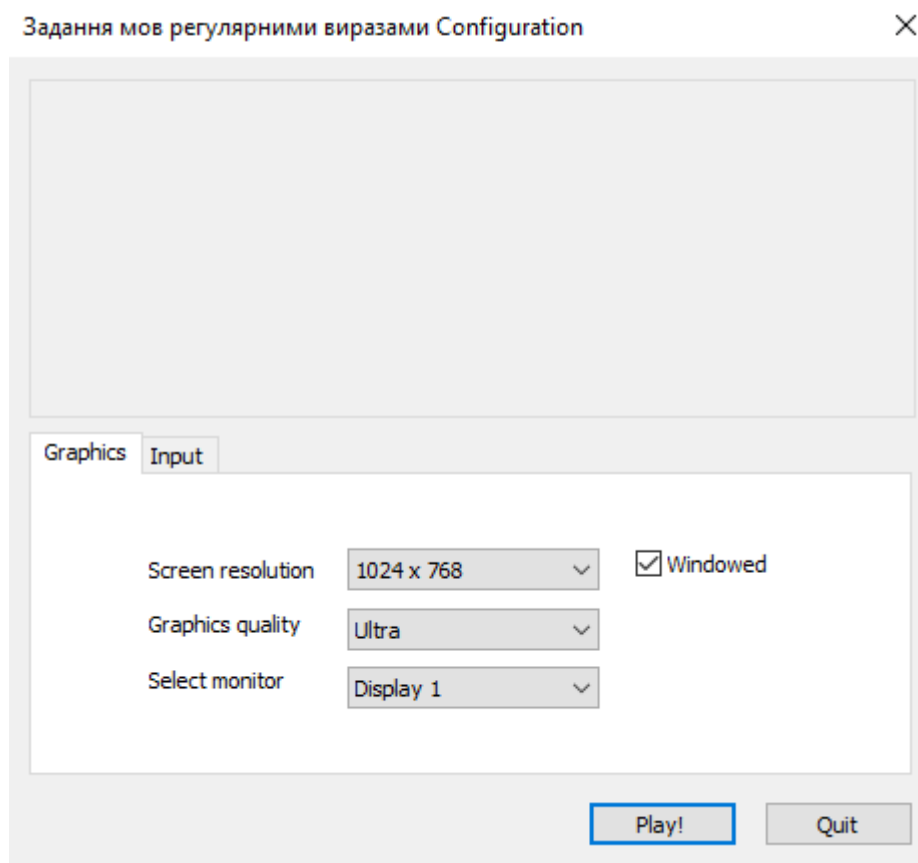


Рисунок 4.6 – Налаштування конфігурації

Після вибору конфігурації користувач переходить до головного меню тренажеру.

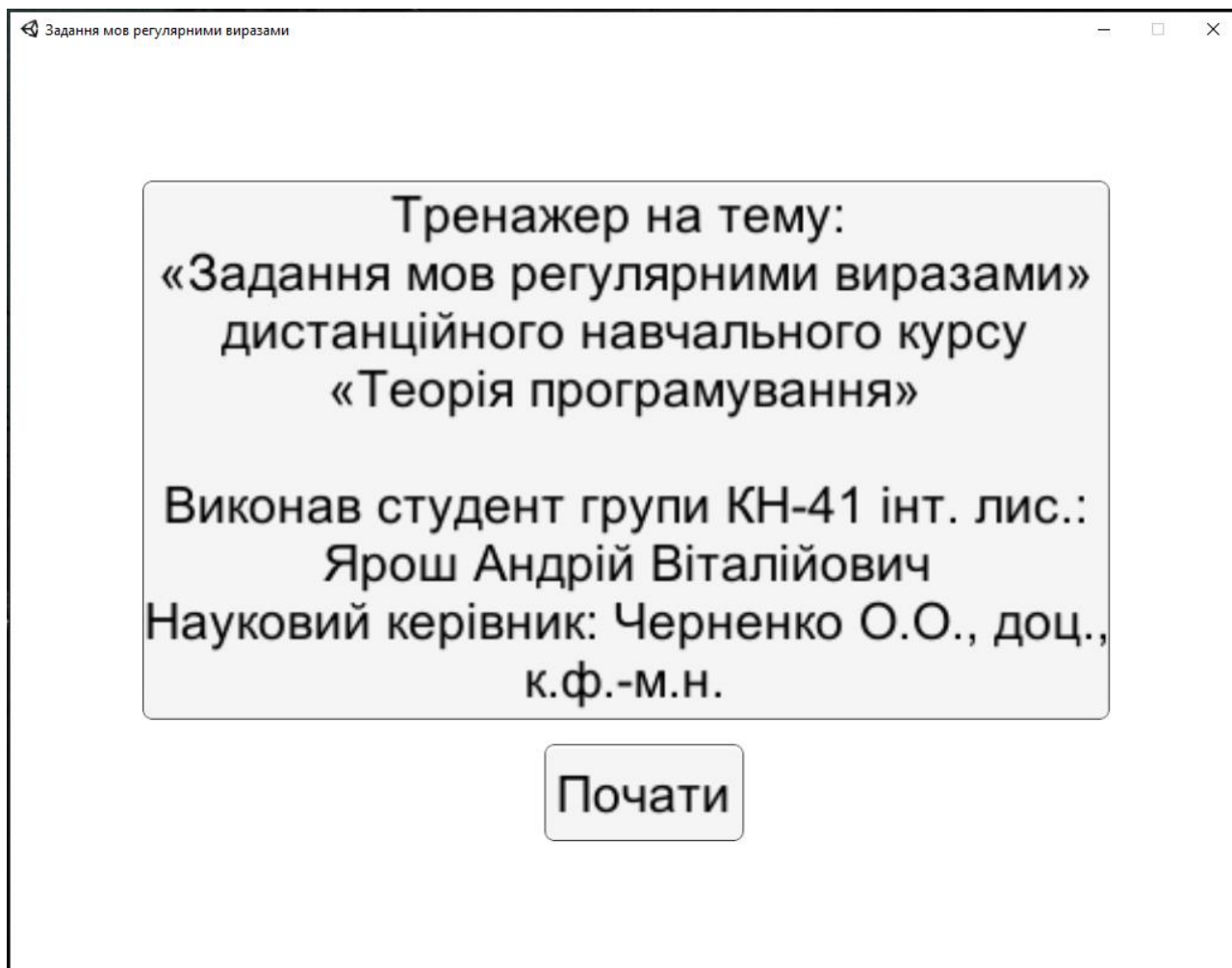


Рисунок 4.7 – Головне меню тренажеру

Після натиснення на кнопку «Почати» користувач переходить до вікна з інструкцією до тренажеру.

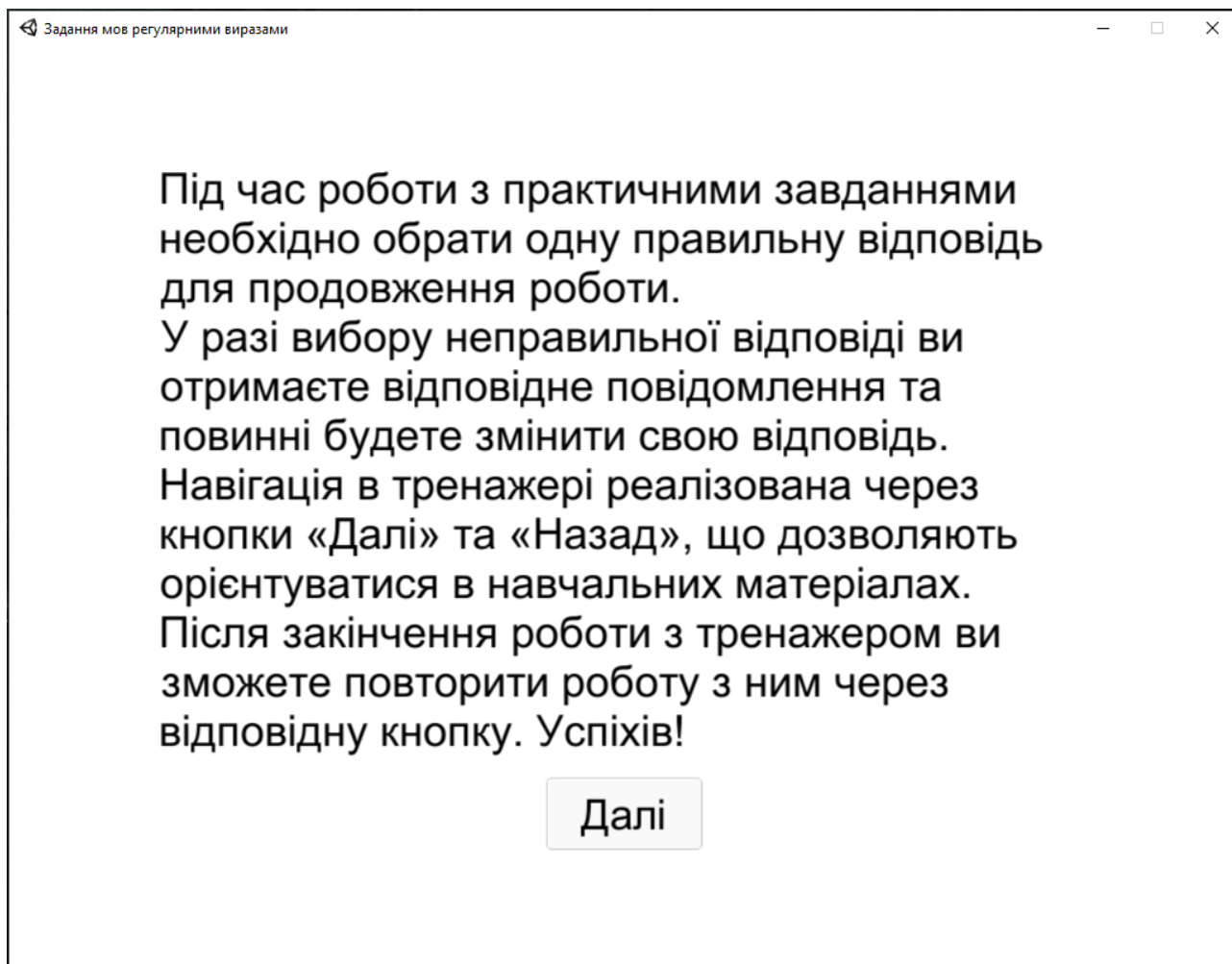


Рисунок 4.8 – Інструкція для роботи з тренажером

Після ознайомлення з інструкцією користувач переходить до вікна з практичними завданнями.

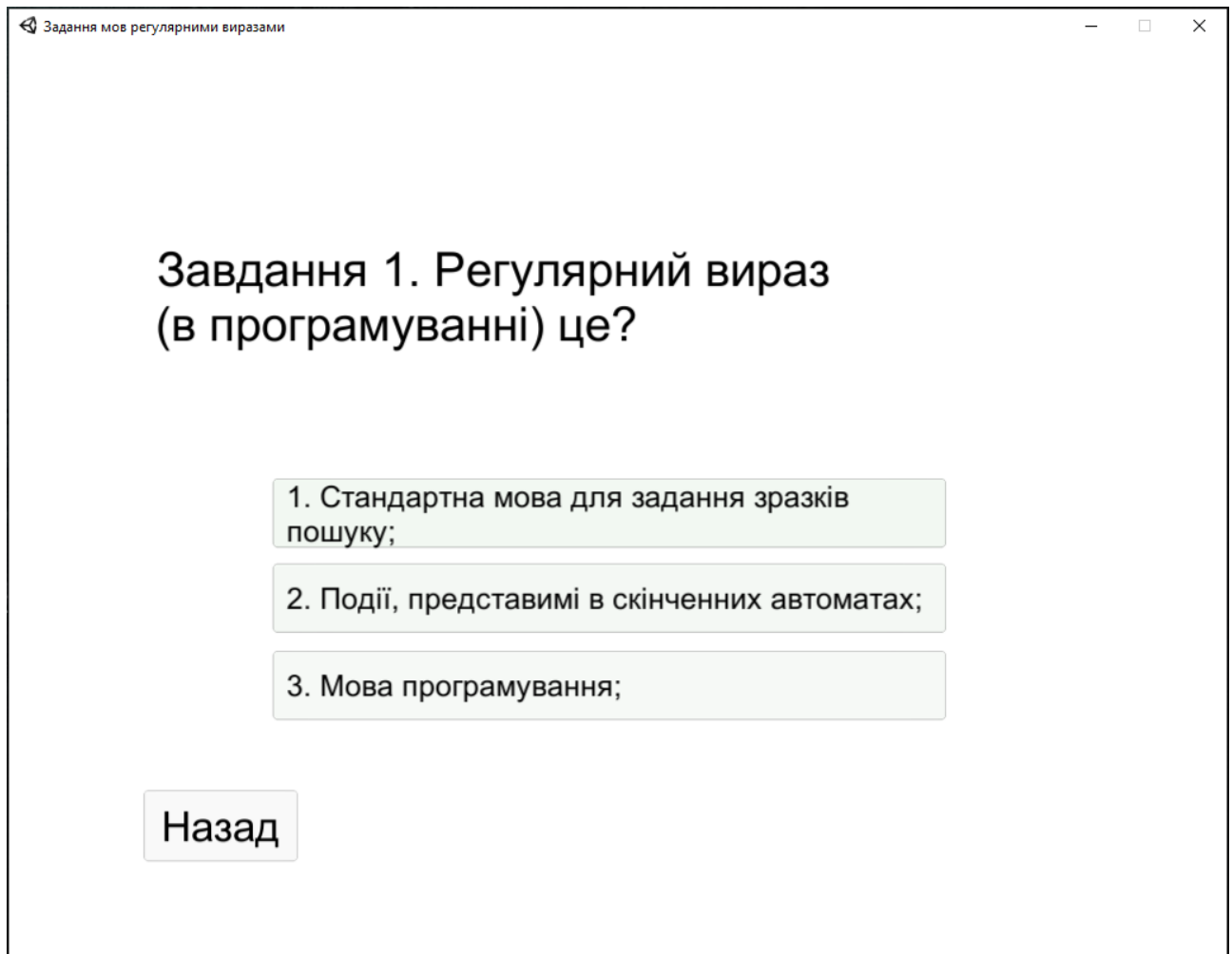


Рисунок 4.9 – Практичне завдання

Після вибору неправильної відповіді користувач отримує повідомлення «Неправильно» та отримує можливість виправити свою відповідь.

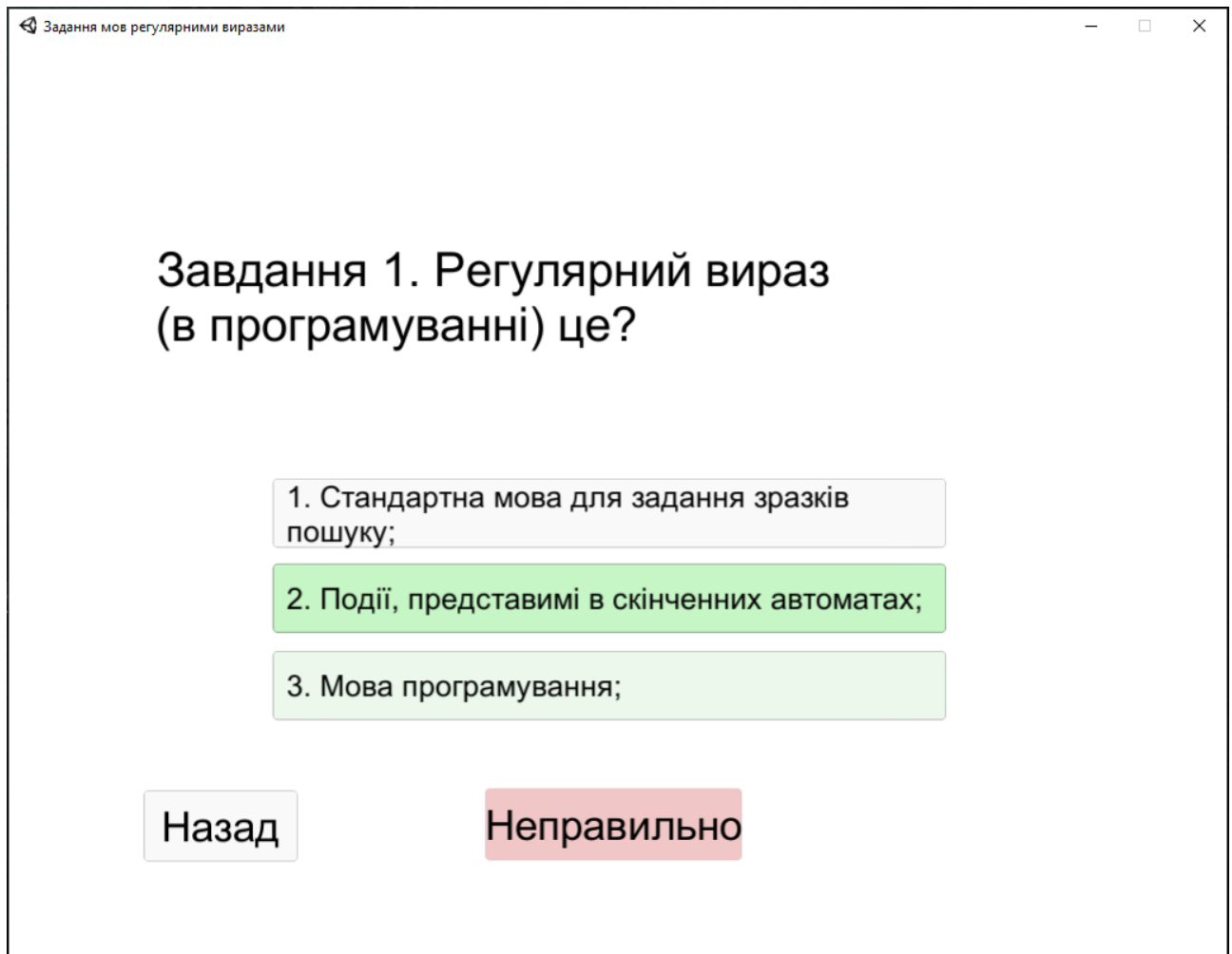


Рисунок 4.10 – Практичне завдання після вибору неправильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

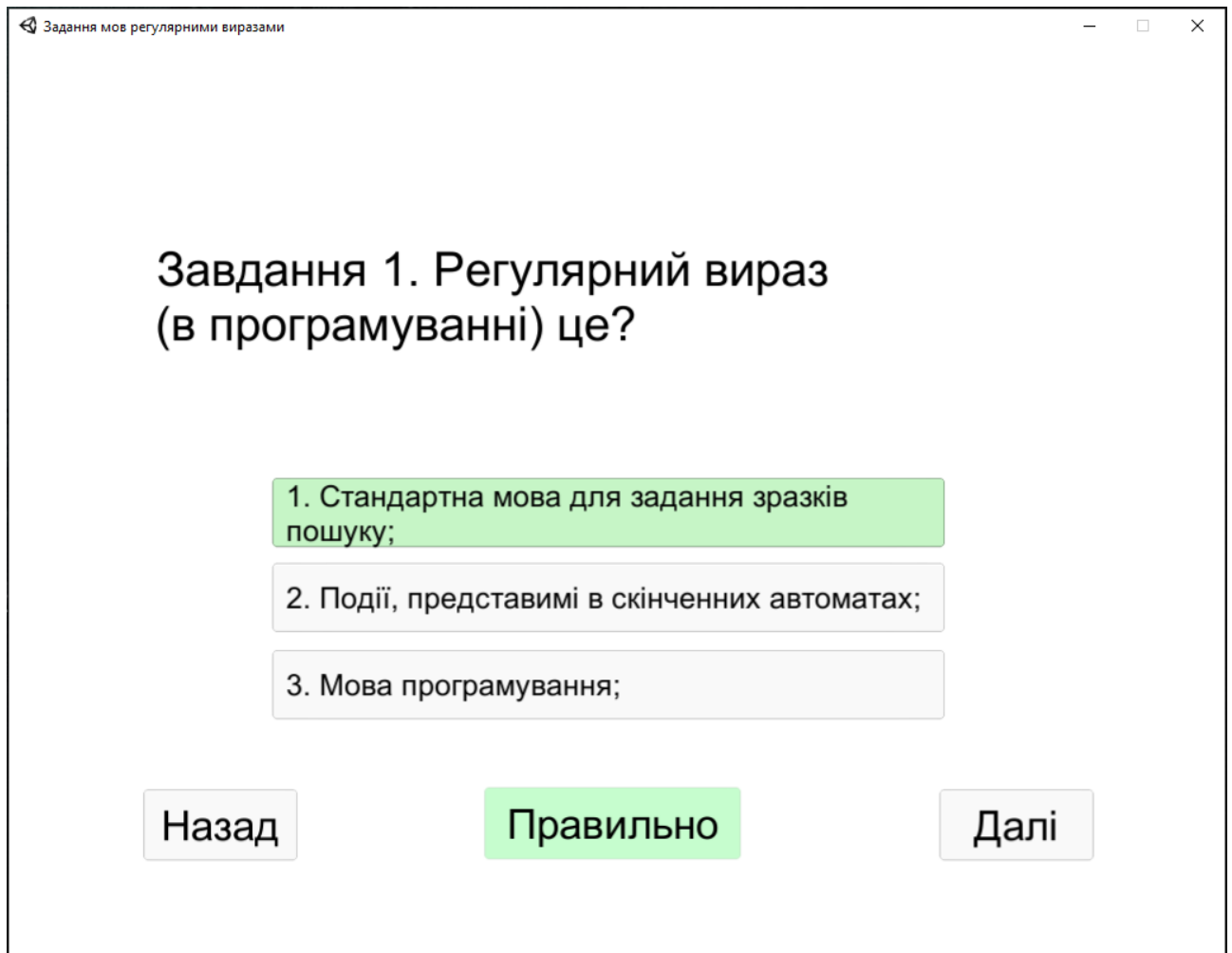


Рисунок 4.11 – Практичне завдання №1 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

Задання мов регулярними виразами

Завдання 2. Скільки разів повинен виконатися вислів A при записі регулярного виразу $L(A^+)$?

1. Один раз;

2. Один або декілька разів;

3. Більше одного разу;

Назад Правильно Далі

Рисунок 4.12 – Практичне завдання №2 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

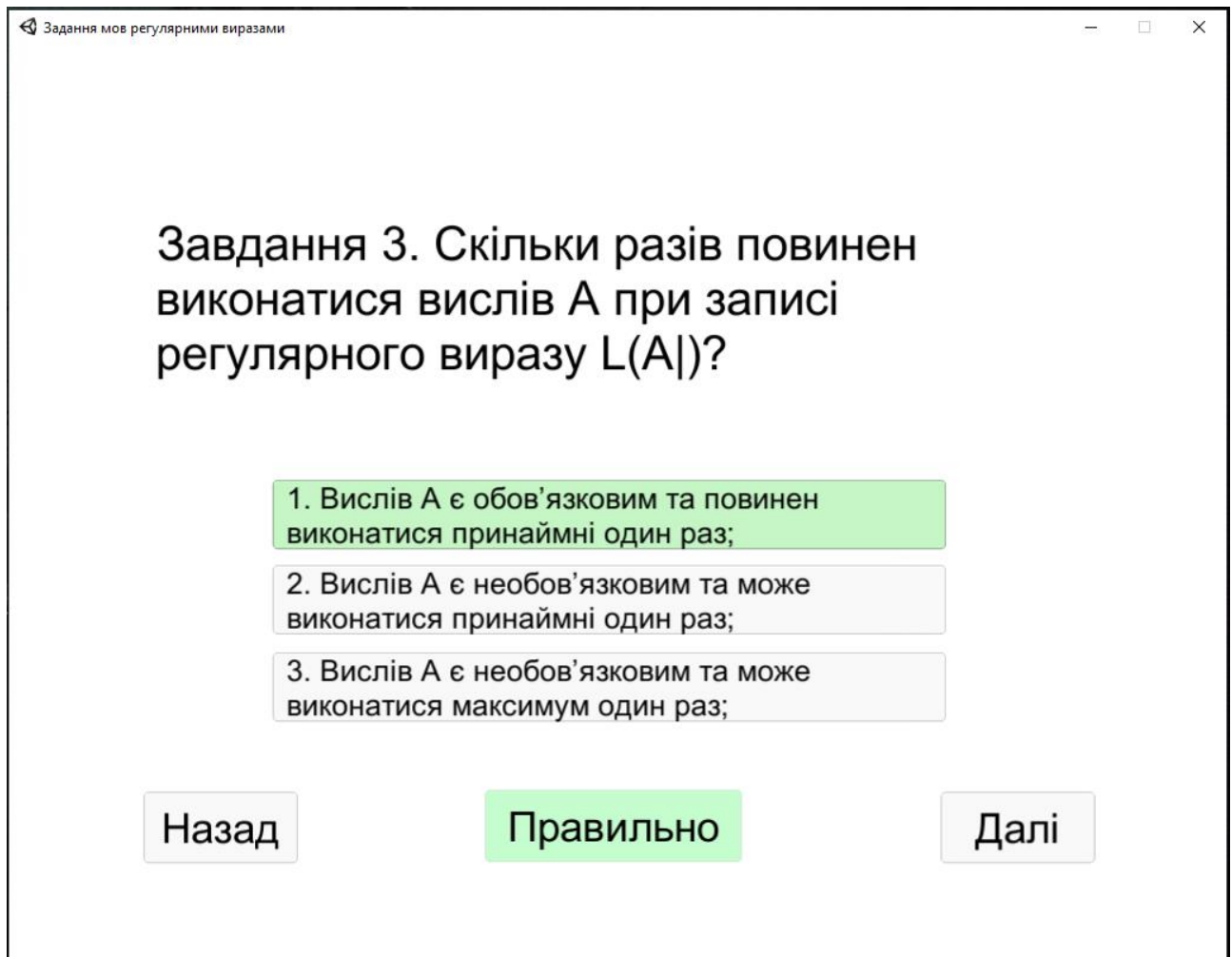


Рисунок 4.13 – Практичне завдання №3 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

Задання мов регулярними виразами

Завдання 4. Скільки разів повинен виконатися вислів A при записі регулярного виразу $L(A^*)$?

1. Вираз повинен збігтися нуль разів;
2. Вираз повинен збігтися нуль чи більше разів;
3. Вираз повинен збігтися один чи більше разів;

Назад Правильно Далі

Рисунок 4.14 – Практичне завдання №4 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

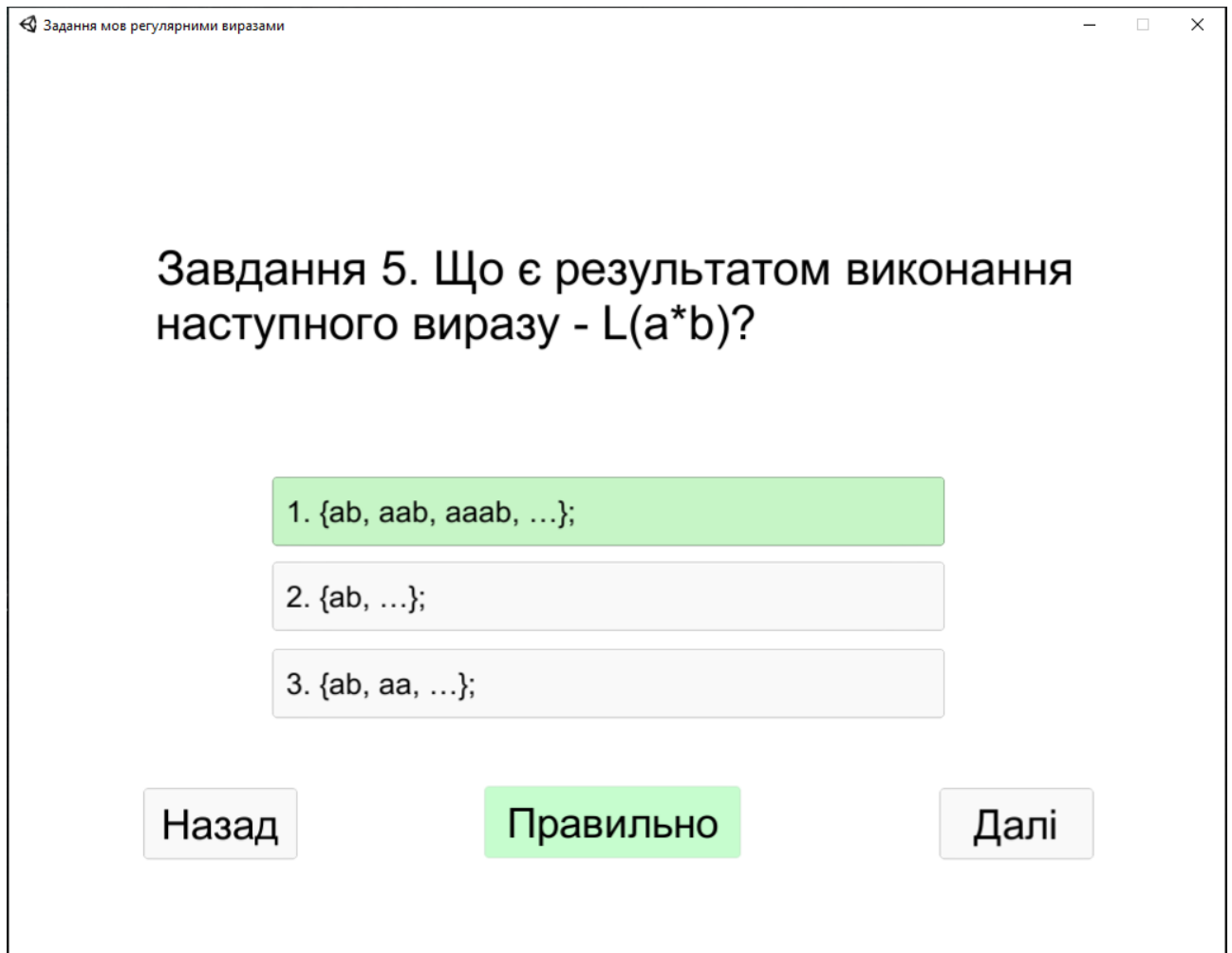
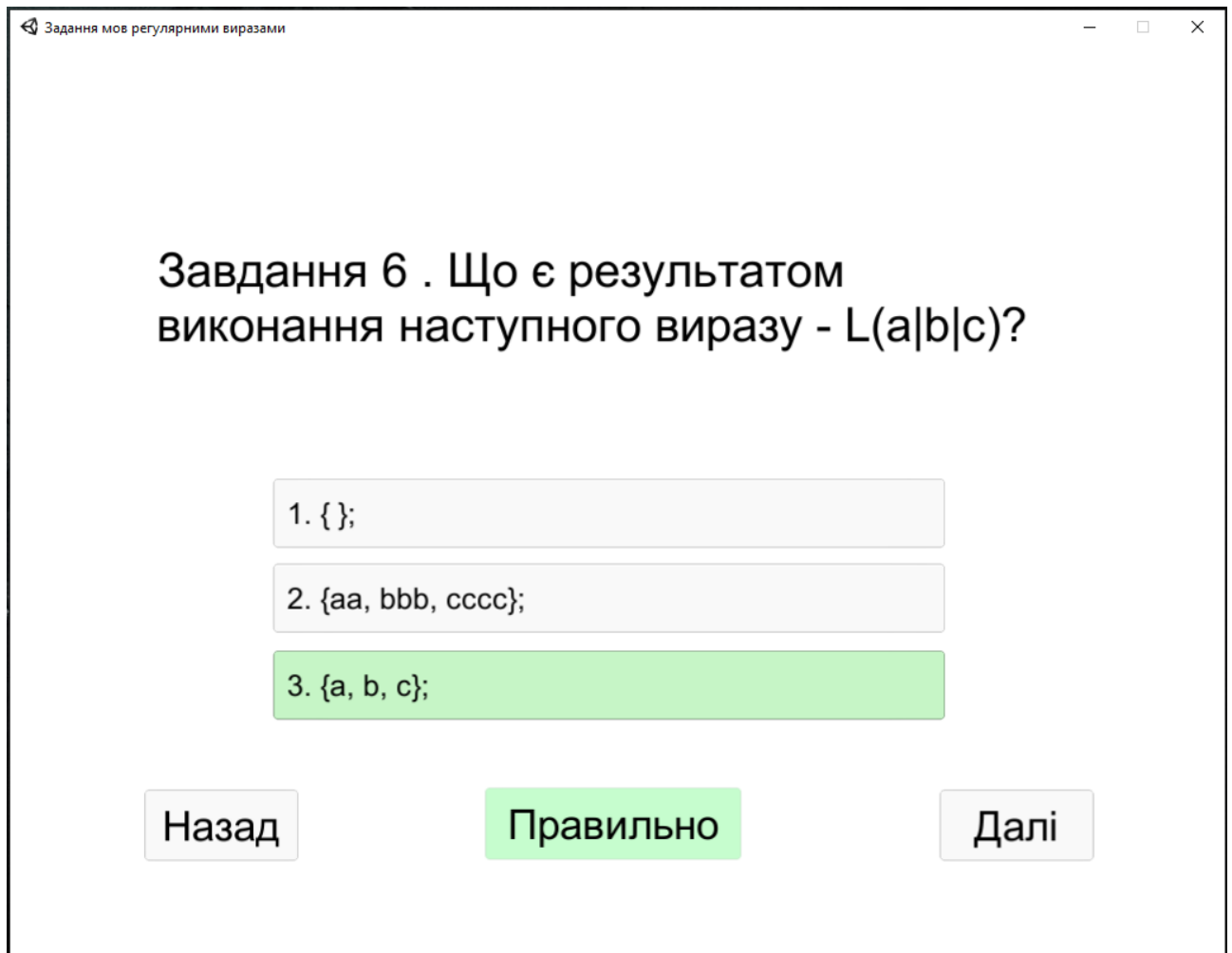


Рисунок 4.15 – Практичне завдання №5 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.



Задання мов регулярними виразами

Завдання 6 . Що є результатом виконання наступного виразу - $L(a|b|c)$?

1. { };

2. {aa, bbb, cccc};

3. {a, b, c};

Назад Правильно Далі

Рисунок 4.16 – Практичне завдання №6 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

Задання мов регулярними виразами

Завдання 7. Що є результатом виконання наступного виразу - $L(ab^*c)$?

1. {a, ab, abc, ...};

2. {ac, abc, abbc, ...};

3. {abc, abbc, ...};

Назад Правильно Далі

Рисунок 4.17 – Практичне завдання №7 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

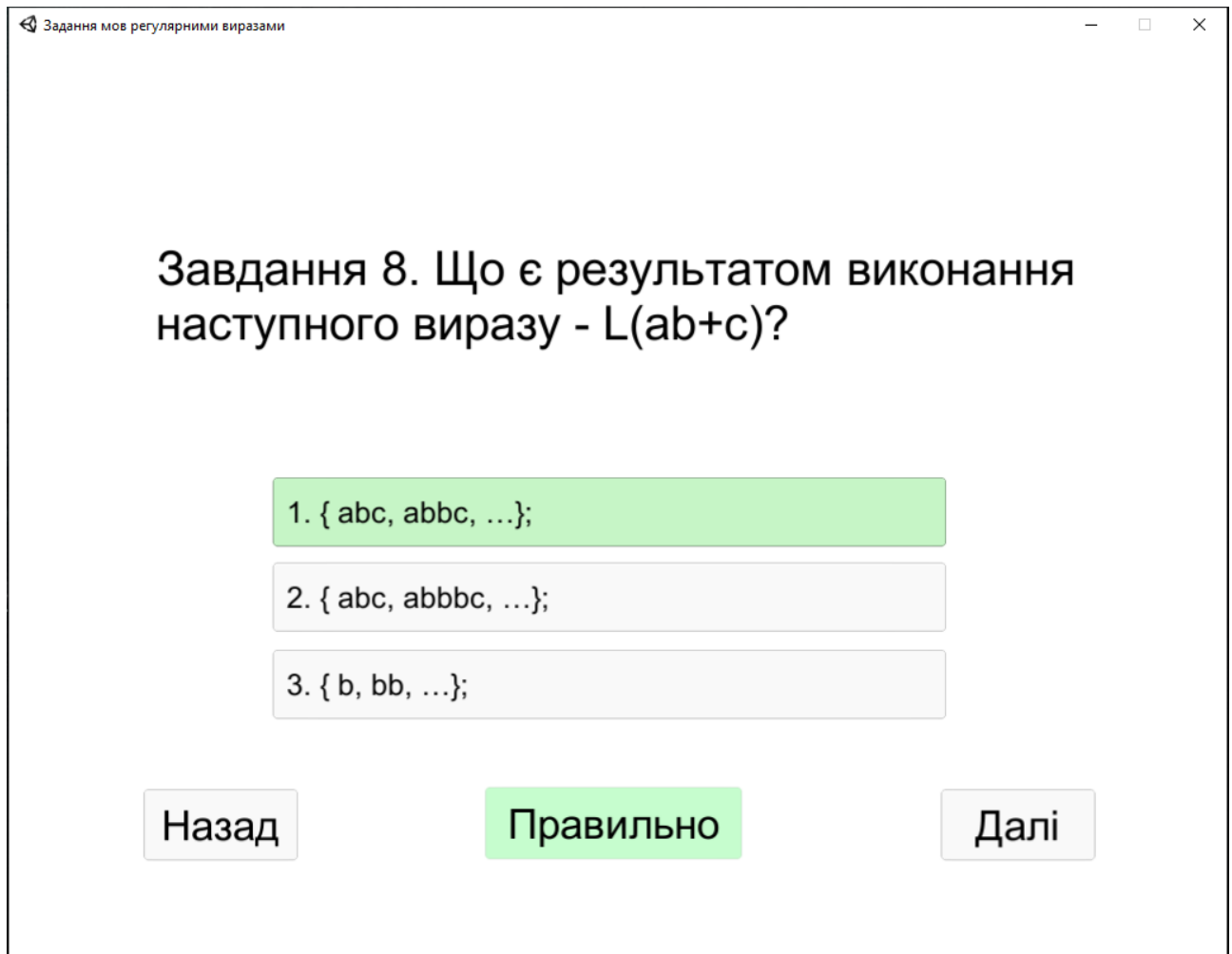


Рисунок 4.18 – Практичне завдання №8 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

Задання мов регулярними виразами

Завдання 9. Що є результатом виконання наступного виразу - $L(a(b|c)d)$?

1. {abc, abd};

2. {abd, acd};

3. {ad, ab, ac};

Назад Правильно Далі

Рисунок 4.19 – Практичне завдання №9 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з наступним практичним завданням.

Задання мов регулярними виразами

Завдання 10. Що є результатом виконання наступного виразу - $L(a(b+c)d|)$?

1. {ab, abb, ...};

2. {abcd, abbcd, ...};

3. {abcd, abbc, ...};

Назад Правильно Далі

Рисунок 4.20 – Практичне завдання №10 після вибору правильної відповіді

Після вибору правильного варіанту відповіді користувач переходить до вікна з повідомленням про завершення роботи з тренажером та кнопкою «Повтор».

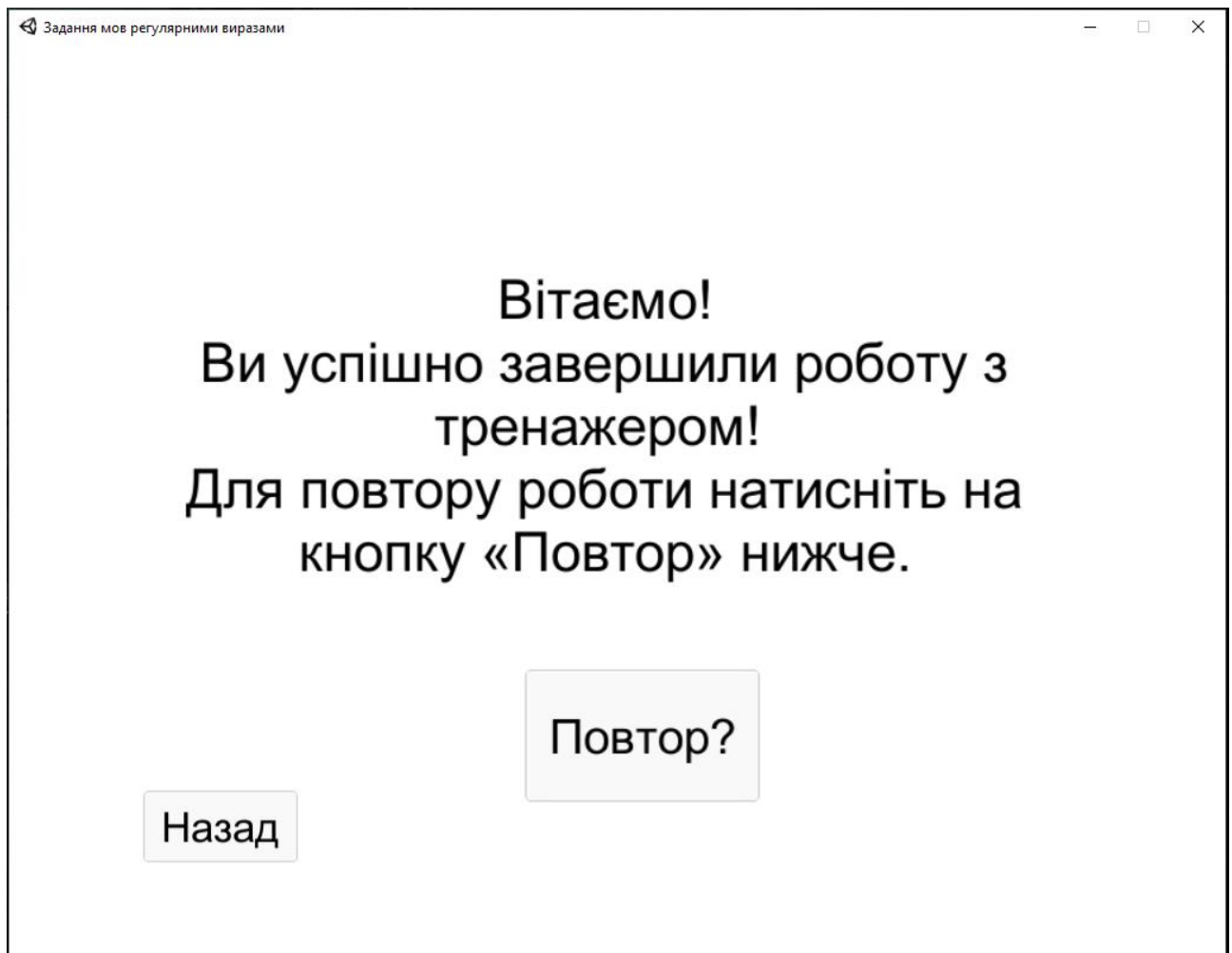


Рисунок 4.21 – Вікно з повідомленням про завершення роботи з тренажером та кнопка «Повтор»

ВИСНОВКИ

Результатами роботи над бакалаврською роботою є:

1. Знайдено та оформлено теоретичну частину для тренажеру;
2. Проведено інформаційний огляд тренажерів зі схожою тематикою;
3. Розроблено алгоритм роботи тренажеру;
4. Розроблено блок-схеми алгоритму тренажеру;
5. Створено програмне забезпечення у вигляді тренажеру за розробленим алгоритмом.

Позитивними сторонами програмного забезпечення у вигляді тренажеру є:

1. Практичні завдання в тренажері видаються лише після вибору правильної відповіді;
2. Практичні завдання в тренажері оформлені у вигляді тестів з однією правильною або декількома відповідями;
3. Після закінчення роботи з тренажером надано можливість повернутися в меню для повтору роботи з тренажером.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. 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.

2. Скромінський М.В. Пояснювальна записка до бакалаврської роботи на тему Розробка програмного забезпечення для тренажеру з теми «Контекстовільні граматики» дистанційного навчального курсу «Теорія програмування» / Скромінський М.В. [Електронний ресурс].– Режим доступу до ресурсу: <http://dspace.puet.edu.ua/handle/123456789/9025>

3. Костромін І.І. Пояснювальна записка до бакалаврської роботи на тему Тренажер з теми «Математичні основи теорії алгоритмів» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення / Костромін І.І. [Електронний ресурс].– Режим доступу до ресурсу: <http://dspace.puet.edu.ua/handle/123456789/9005>

4. Регулярний вираз [Електронний ресурс].– Режим доступу до ресурсу:

https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%83%D0%BB%D1%8F%D1%80%D0%BD%D0%B8%D0%B9_%D0%B2%D0%B8%D1%80%D0%B0%D0%B7

5. Черненко О.О. Теорія програмування, Регулярні мови [Електронний ресурс].– Режим доступу до ресурсу:

<http://www2.el.puet.edu.ua/st/mod/page/view.php?id=105314>

6. Черненко О.О. Теорія програмування, [Електронний ресурс].– Режим доступу до ресурсу: <http://www2.el.puet.edu.ua/st/course/view.php?id=1533>

7. Unity, [Електронний ресурс].– Режим доступу до ресурсу:

[https://uk.wikipedia.org/wiki/Unity_\(%D1%80%D1%83%D1%88%D1%96%D0%B9_%D0%B3%D1%80%D0%B8\)](https://uk.wikipedia.org/wiki/Unity_(%D1%80%D1%83%D1%88%D1%96%D0%B9_%D0%B3%D1%80%D0%B8))

8. Unity, Робота з платформою [Електронний ресурс] .– Режим доступу до ресурсу: <https://unity.com/ru>
9. Регулярні вирази та граматики. Синтаксичні діаграми [Електронний ресурс].– Режим доступу до ресурсу: <https://studfile.net/preview/7013828/page:4/>
10. Мова програмування С# в IDE MS Visual Studio [Електронний ресурс].– Режим доступу до ресурсу: <https://docs.microsoft.com/en-us/visualstudio/get-started/csharp/?view=vs-2019>
11. Майкл Фицджеральд, "Введение в регулярные выражения" (2012) \ Майкл Фицджеральд, 2012 – 258 с.

ДОДАТОК А

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.UI;
using UnityEngine;
public class QuitButton : MonoBehaviour
{
    public void QuitApp()
    {
        Application.Quit();
    }
void Start()
    {
        Button btn = nxtButton.GetComponent<Button>();
        btn.onClick.AddListener(TaskOnClick);
    }
void TaskOnClick()
    {
        Theme1.SetActive(false);
        Theme2.SetActive(true);
    }
void Start()
    {
        Button btn = menuButton.GetComponent<Button>();
        btn.onClick.AddListener(TaskOnClick);
    }
void TaskOnClick()
    {
        Slide8.SetActive(false);
```

```
MainMenu.SetActive(true);  
}
```