ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ІНСТИТУТ ЗАОЧНО-ДИСТАНЦІЙНОГО НАВЧАННЯ

ФОРМА НАВЧАННЯ ЗАОЧНА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ ІНФОРМАТИКИ

Допускається до захисту

Завідуючий кафедрою _____ О.О. Ємець
(підпис, ініціали, прізвище)
«____»_____ 2021 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА ДО ДИПЛОМНОЇ РОБОТИ

на тему

ТРЕНАЖЕР З ТЕМИ «КОНТЕКСТОВІЛЬНІ МОВИ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ ПРОГРАМУВАННЯ»

зі спеціальності 122«Комп'ютерні науки»

Виконавець роботи
Лєбєдєва Марія Олександрівна _____ «__»_____2021 р.
(підпис)

Науковий керівник
к.ф.-м.н, доцент Черненко Оксана Олексіївна _____ «__»_____2021 р.
(підпис)

Полтава – 2021 р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, символів
GUI	Графічний інтерфейс користувача
IDE	Integrated development environment – Інтегроване середовище розробки
Java	Об'єктно-орієнтована мова програмування
Moodle	Система управління навчанням
Алфавіт	Скінченна множина символів
Мова	Довільна множина рядків у деякому алфавіті
Рядок (у алфавіті)	Скінченна послідовність символів з деякого алфавіту

ВСТУП

В навчанні студентів дуже важливим є самостійне навчання студентів. Особливо це актуально для студентів заочної та дистанційної форм навчання, а також у випадках, коли стаціонарна форма навчання є недоступною з певних причин.

Для самостійного навчання неважко знайти теоретичну інформацію. Для цього можна використати підручники, довідники або просто знайти потрібну інформацію в мережі Internet. Також для студентів Полтавського університету економіки і торгівлі доступні дистанційні курси [1], електронні посібники [2] та інші джерела інформації. Використовуючи такі матеріали можна отримати гарну теоретичну підготовку. Основним недоліком такого підходу є слабка практична підготовка. А саме практичні навички є найбільш важливими у підготовці фахівців з комп'ютерних наук.

Для кращої теоретичної підготовки на кафедрі математичного моделювання та соціальної інформатики створено багато навчальних тренажерів [3-26]. Тренажер призначений для того, щоб допомогти здобути чи закріпити практичні навички з певної теми деякої навчальної дисципліни.

З дисципліни «Теорія програмування» розроблено тренажери з різних тем [3-13], але з теми «Контекстовільні мови» тренажера немає. Тому розробка тренажера з теми «Контекстовільні мови» дистанційного навчального курсу «Теорія програмування» є **актуальною**.

Метою роботи є програмна реалізація тренажера з теми «Контекстовільні мови» дисципліни «Теорія програмування». Для досягнення цієї мети потрібно розв'язати такі основні **задачі**:

1) аналіз існуючих тренажерів із подібних тем;

2) підбір завдань для тренажера;

3) розробка алгоритму роботи тренажера та його блок-схеми;

4) вибір технології та мови програмування;

5) програмна реалізація тренажера;

6) перевірка правильності роботи тренажера.

Об’єктом розробки є програмне забезпечення для самостійного навчання студентів.

Предмет розробки – навчальний тренажер з теми «Контекстовільні мови» дисципліни «Теорія програмування».

Методи: методи теорії програмування, методи розробки на мові програмування Java.

Робота складається із вступу, чотирьох розділів, висновків, списку літератури та додатку, в якому розміщений сирцевий код тренажера.

Для розробки тренажера використовується об'єктно-орієнтована мова програмування Java. Графічний інтерфейс користувача (GUI) створений за допомогою технології JavaFX. Тренажер розроблений за допомогою інтегрованого середовище розробки (IDE) IntelliJ IDEA Community.

Новизною роботи є розроблений тренажер з теми «Контекстовільні мови» дисципліни «Теорія програмування». Тренажер може використовуватися для самостійного вивчення відповідної теми студентами різних форм навчання.

1 ПОСТАНОВКА ЗАДАЧІ

Головною задачею магістерської роботи є розробка навчального тренажера з теми «Контекстовільні мови» дистанційного курсу «Теорія програмування». Для зручності користувачів тренажер буде створений у вигляді окремої програми, яку студент зможе завантажити на свій комп'ютер і запускати в зручний для нього час. Робота тренажера не буде залежати від наявності підключення до мережі Інтернет.

Тренажер призначений для самостійної роботи студента при вивченні відповідної теми дисципліни «Теорія програмування». Тому до тренажера висуваються певні вимоги.

1. Тренажер повинен мати простий і зрозумілий графічний інтерфейс користувача, який би дозволив студентам працювати з ним, не читаючи довгих інструкцій та не використовуючи інших довідкових матеріалів.

2. Тренажер має бути доступним у використанні навіть тим студентам, які погано опанували теоретичний матеріал з теми «Контекстовільні мови» та з попередніх тем дисципліни. Перед початком роботи з тренажером студент має ознайомитися із відповідним навчальним матеріалом. Але навіть якщо студент допускає помилки, то тренажер повинен допомогти їх виправити.

3. Також до тренажера є технічні вимоги. Тренажер повинен працювати на різних операційних та апаратних платформах, оскільки студенти будуть працювати на власних комп'ютерах. Тренажер має запускатися без різних додаткових налаштувань.

Виконання поставленої задачі буде здійснюватися в наступних етапах.

1. Аналіз існуючих тренажерів подібної тематики. Виділення їхніх сильних та слабких сторін.

2. Створення завдань для студентів, які будуть реалізовані в тренажері.

3. Розробка алгоритму роботи тренажера. На цьому етапі потрібно визначити, як саме буде працювати тренажер: які будуть етапи, порядок їх проходження, реакція тренажера на помилкові відповіді.

4. Розробка блок-схеми відповідного алгоритму. Цей етап є обов'язковим і дозволяє краще зрозуміти алгоритм роботи тренажера перед його реалізацією.

5. Розробка тренажера за допомогою однієї із мов програмування.

6. Перевірка роботи тренажера.

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

На кафедрі математичного моделювання та соціальної інформатики за останні кілька років було розроблено велику кількість тренажерів з різних дисциплін [3-26]. Розглянемо тренажери з дисципліни «Теорія програмування».

Тренажер з теми «Мови і граматики» [3] містить 3 основні завдання, для розв'язання кожного з яких потрібно пройти кілька кроків. Кожен із кроків є запитанням із можливими варіантами відповіді. Після правильної відповіді відбувається перехід до наступного кроку. Якщо дано неправильну відповідь, то виводиться повідомлення, що містить підказку. Після цього також відбувається перехід до наступного кроку.

Перед початком проходження тренажера є можливість ознайомитися із теоретичним матеріалом із відповідної теми. Тренажер створено за допомогою мови програмування Python.

В тренажері можна відзначити простий і зрозумілий інтерфейс, розбиття задачі на нескладні кроки та інформативні підказки у випадку неправильних відповідей.

Тренажер [4] допомагає розібратися студентам в питаннях теми «Алгоритмічно нерозв'язні проблеми» з дисципліни «Теорія програмування». Зокрема, в роботі розглядаються наступні задачі.

1) Розподіл дев'яток у запису числа π.

2) Десята проблема Гільберта, що полягає у знаходженні універсального методу цілочислового розв'язання довільного алгебраїчного діофантового рівняння.

3) Обчислення досконалих чисел, тобто таких чисел, що дорівнюють сумі своїх дільників.

4) Позиціонування машини Поста на останньому позначеному ящику.

5) Проблема еквівалентності алгоритмів.

Тренажер написаний на мові програмування Java. Початкове вікно тренажера показане на рис. 2.1.

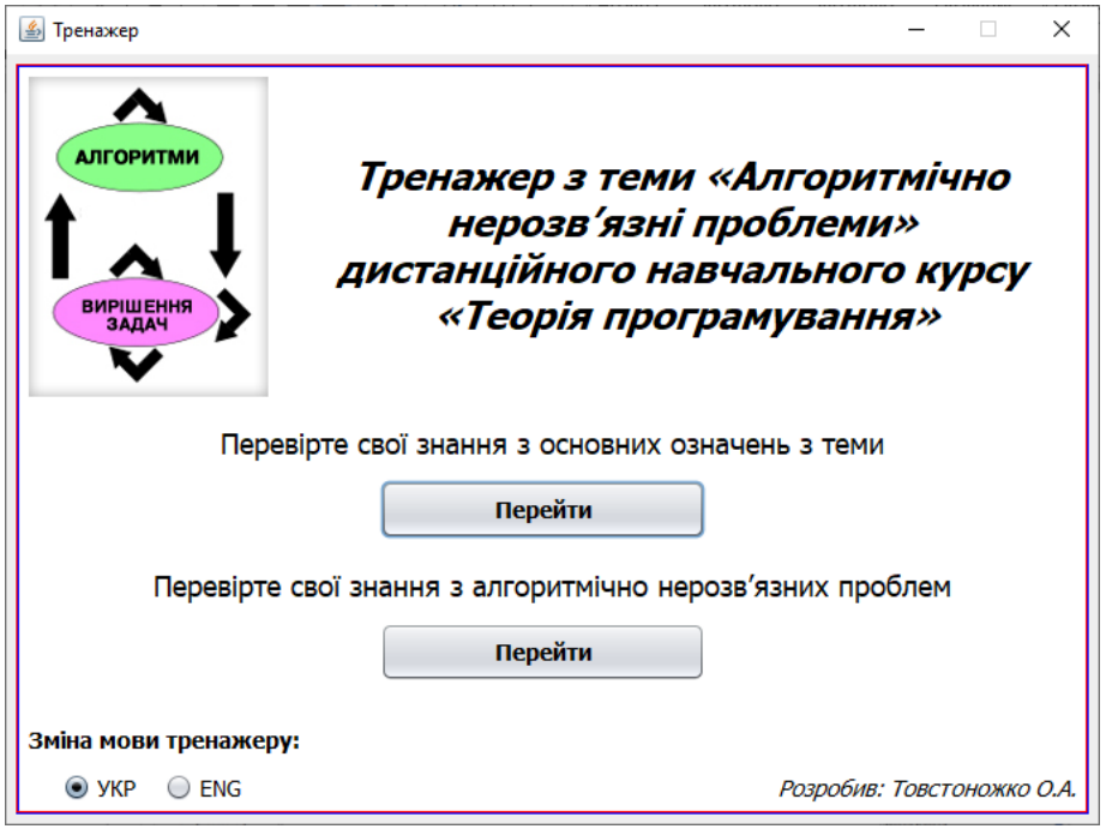

Рис. 2.1 Стартове вікно тренажера з теми «Алгоритмічно нерозв'язні проблеми»

Тренажер складається із набору кроків, на кожному із яких дається запитання в тестовій формі. Користувач повинен вибрати правильну відповідь або розташувати кроки доведення у правильній послідовності. Якщо дано неправильну відповідь, то стає активною вкладка із поясненням правильної відповіді.

Серед позитивних сторін тренажера можна виділити:

1) простий та зрозумілий графічний інтерфейс;

2) наявність детального пояснення правильних відповідей на кожному кроці;

3) підтримка тренажером двох мов: української та англійської.

Недоліком тренажера є невелика кількість завдань, як для непростої теми «Алгоритмічно нерозв'язні проблеми».

В [5] розглядається алгоритмізація та програмування тренажера з теми «Способи задання мов». Робота тренажера також розбита на кроки. На кожному кроці тренажера виводиться питання та варіанти відповідей. Користувач повинен

вибрати правильну відповідь. Якщо вибрана неправильна відповідь, то виводиться повідомлення про це і користувач повинен повторити спробу.

Тренажер написаний на мові програмування C#, для виведення інформації користувачу використовується технологія HTML.

Недоліком тренажера є те, що у випадку неправильної відповіді користувач лише отримує повідомлення про помилку, але не отримує підказки чи пояснення, чому саме відповідь неправильна.

Тренажер з теми «Обрізання основ» [6] написаний на мові програмування Java. Тренажер містить теоретичну та практичну частини. В тренажері можна розв'язати 2 задачі. Розв’язок кожної розбитий на кроки. На кожному кроці потрібно вибрати правильну відповідь. Якщо відповідь правильна, то відбувається перехід до наступного кроку (на рис. 2.2. показаний вигляд вікна тренажера на одному із кроків). У випадку неправильної відповіді користувач може спробувати дати відповідь знову. Якщо знову дано неправильну відповідь, то виводиться підказка із правильною.

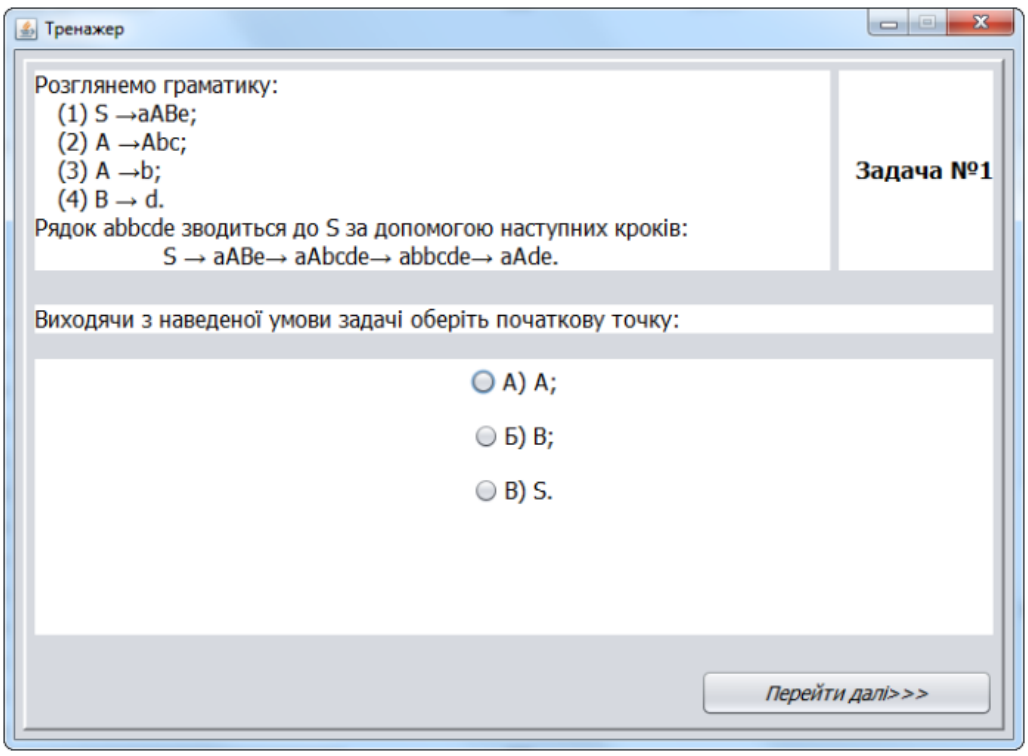

Рис. 2.2 Тренажер з теми «Обрізання основ»

Після проходження всіх кроків задачі користувач отримує інформацію про сумарну кількість допущених помилок та пропозицію пройти тренажер ще один раз.

Серед сильних сторін тренажера можна виділити:

1) простота та приємний дизайн графічного інтерфейсу;

2) наявність теоретичної інформації;

3) інформація про сумарну кількість помилок, яка виводиться після проходження тренажера.

Недоліком є те, що, як і в попередньому розглянутому тренажері, у випадку неправильної відповіді не пояснюється, чому відповідь неправильна.

Тренажер з теми «Контекстовільні граматики» [7] містить теоретичний матеріал та один приклад, розв’язання якого розбите на кроки. На кожному кроці пропонується кілька варіантів відповіді на поставлене запитання. Перший крок тренажера показаний на рис. 2.3.

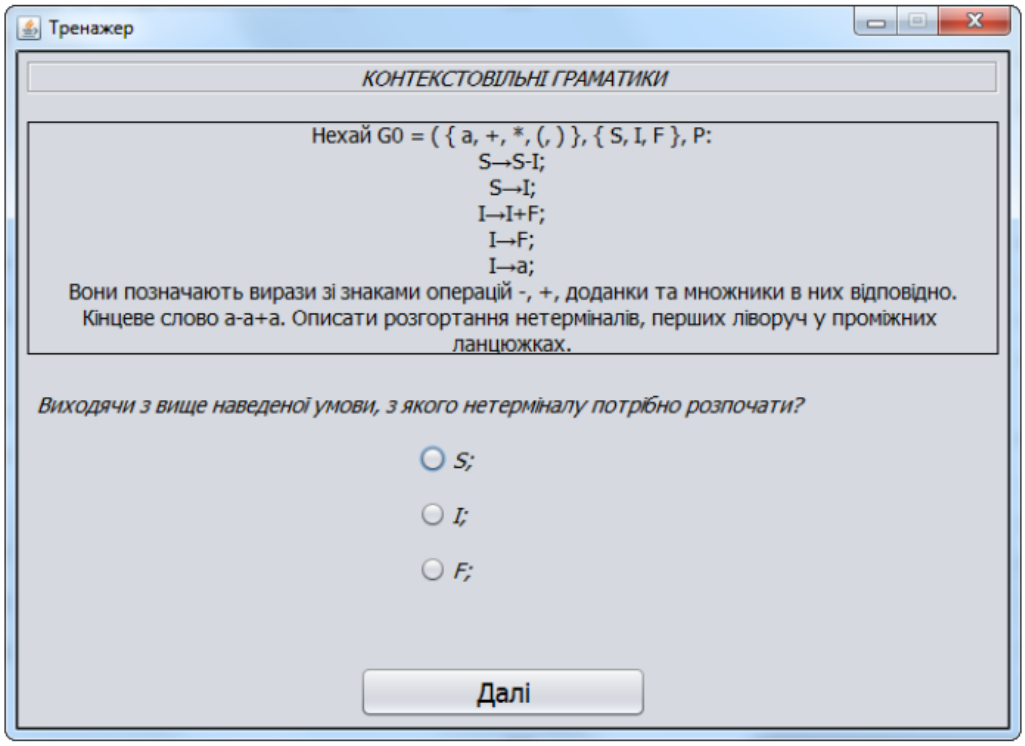

Рис. 2.3 Перший крок тренажера з теми «Контекстовільні граматики»

Якщо користувач дає неправильну відповідь, то виводиться інформація про помилку та правильна відповідь.

Перевагою тренажера є наявність теоретичного матеріалу.

Недоліком є те, що у випадку помилкової відповіді, студент отримує одразу правильну відповідь без пояснення, чому саме та чи інша відповідь правильна чи неправильна.

В [8] розглядається розробка тренажера з теми «Математичні основи теорії алгоритмів». Тренажер містить теоретичні відомості та завдання в тестовій формі. Для розробки використовується мова програмування Java.

В [9] розглядається тренажер з теми «Видалення лівої рекурсії». Робота тренажера також розбита на кроки, на кожному із яких студент отримує питання в тестовій формі. Тренажер розроблений за допомогою мови програмування Java.

В [10-11] розглядається тренажер з теми «Дерева розбору». Цей тренажер також розроблений на мові програмування Java.

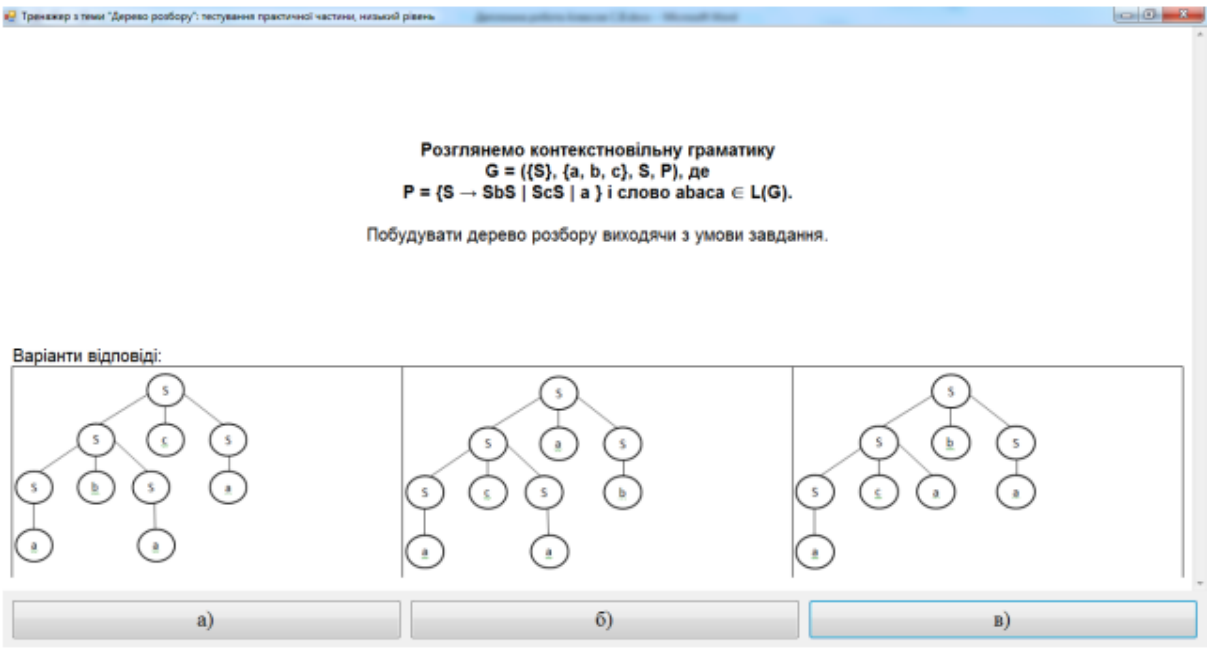

Рис. 2.4 Один із кроків тренажера з теми «Дерева розбору»

Робота тренажера розбита на кроки, один із яких показано на рис. 2.4. На кожному кроці користувач має вибрати правильну відповідь із кількох запропонованих варіантів. Серед переваг тренажера можна виділити:

1) велика кількість кроків із різними запитаннями;

2) кроки розбиті на різні рівні складності;

3) після проходження кожного рівня складності виводиться інформація про кількість помилок.

В [12] розглядається тренажер з теми «Побудова предикативних синтаксичних аналізаторів». Вигляд стартового вікна тренажера показаний на рис. 2.5.

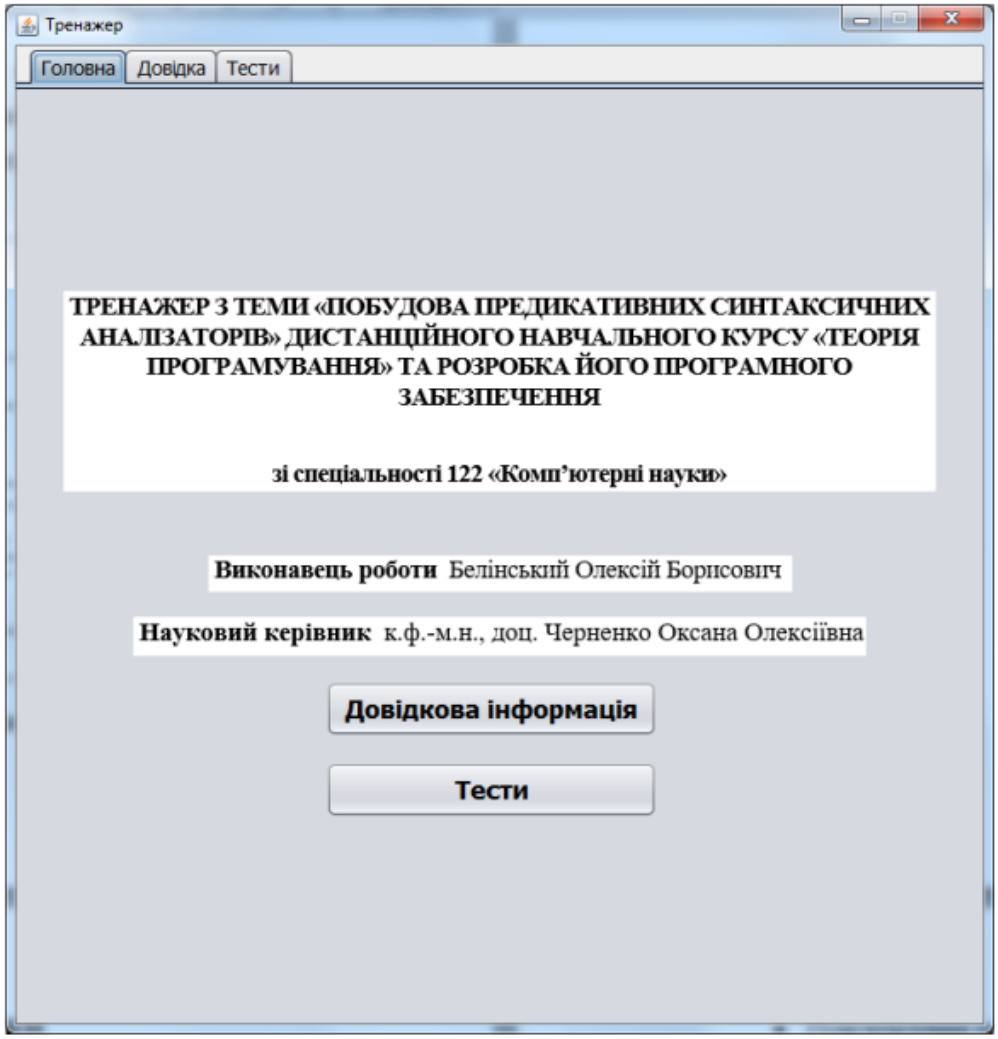

Рис. 2.5 Тренажер з теми «Побудова предикативних синтаксичних аналізаторів»

Тренажер також створений за допомогою мови програмування Java. Тренажер містить теоретичну частину та практичну.

Отже, проаналізувавши значну кількість тренажерів з дисципліни «Теорія програмування», можна виділити деякі спільні речі в них.

1. Типовим алгоритмом роботи тренажера і розбиття деякого прикладу чи прикладів на певну кількість послідовних кроків. На кожному кроці користувач має вибрати правильну відповідь із кількох запропонованих.

2. В тренажерах міститься велика кількість теоретичного матеріалу.

3. Переважна більшість тренажерів розроблена за допомогою мови програмування Java.

3 ТЕОРЕТИЧНА ЧАСТИНА

3.1 Контекстовільні мови

Розглянемо основні теоретичні відомості, необхідні для розробки алгоритму роботи тренажера [27].

Алфавіт – це скінченна множина символів.

Наприклад:

1) Множина *{0, 1} є* бінарним алфавітом.

2) 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.

Мова– це довільна множина рядків у деякому алфавіті.

Над мовами можливі операції, що породжують нові мови. Перелік основних операцій наводиться у табл. 3.1.

Таблиця 3.1 Основні операції над мовами

Операція	Визначення
Об'єднання L і M: $L \cup M$.	$L \cup M = \{s \mid s \in L \text{ або } s \in M\}$
Конкатенація L і M: LM	$LM = \{ st \mid s \in L \ i \ t \in M \}$
Замикання Кліні, або ітерація L: L^*.	$L^* = \bigcup_{i=0}^{\infty} L^i$, L^* означає *0* або більше конкатенацій L.
Позитивне замикання L: L^+.	$L^+ = \bigcup_{i=1}^{\infty} L^i$, L^+ означає одну або більше конкатенацій L.

Можна також узагальнити оператор зведення у ступінь для мови, визначивши L^0 як $\{\varepsilon\}$, а L^i – як $L^{i-1}L$.

Нехай L– множина букв *{A, ..., z}*, а D – множина *{0, ..., 9}* . Прикладами мов можуть бути:

$L \cup D$ – множина букв і цифр;

LD – множина рядків з букви і цифри;

L^4 – множина чотирибуквених рядків;

L^* – множина усіх рядків з букв, включаючи ε .

Найпростішу мову можна описати регулярним виразом, що будується із символів за допомогою конкатенації, об'єднання і повторення.

У той самий час мову $\{x^n y^n \mid n>0\}$ неможливо описати регулярним виразом, оскільки в регулярних виразах немає засобів указати, що кількість x повинна дорівнювати кількості y.

Більш могутній механізм – використання продукцій, для даного випадку вони виглядають так:

$$S \to xSy, \quad S \to xy.$$

Символ $\to$ читається, як "може мати вигляд".

Генерація рядків мови:

1) S заміняється правою частиною продукції;

2) якщо отриманий рядок не містить символів S, він є рядком мови.

Приклад послідовності рядків

$S \Rightarrow xSy \Rightarrow xxSyy \Rightarrow xxxyyy$.

Знак $\Rightarrow$ читається як “породжує”.

Послідовність кроків генерації рядка із застосуванням продукцій називається **породженням** (derivation) рядка**.**

Граматика визначається як наступна четвірка компонентів (V_T, V_N, P, S).

Де:

V_T– алфавіт термінальних символів або терміналів;

V_N– алфавіт нетермінальних символів або нетерміналів, $V_T \cap V_N = \varnothing$;

P – множина продукцій (або правил) вигляду $\alpha \to \beta$, α складається з одного або більше символів V, β – з нуля або більше символів V, де $V = V_T \cup V_N$;

S – стартовий символ (або аксіома).

Приклад 1.1

Мова $\{ x^n y^n \mid n>0\}$ описується граматикою

$G_1=(\{x,y\}, \{S\}, P, S)$.

Тут $P = \{ S \to xSy, S \to xy \}$.

Приклад 2.2

Граматикою для мови $\{ x^m y^n \mid m, n \geq 0 \}$ є

$G_2=(\{x,y\}, \{S,B\}, P, S)$.

Тут набір продукцій P має вигляд

$S \to xS,$ $S \to y,$

$S \to yB,$ $B \to yB,$

$S \to x,$ $B \to y.$

Оскільки порожній рядок також належить мові, у набір P також входить продукція $S \to \varepsilon$.

Рядок $xxyyy$ генерується в такий спосіб:

$S \Rightarrow xS \Rightarrow xxS \Rightarrow xxyB \Rightarrow xxyyB \Rightarrow xxyyy.$

Кожен з рядків, що фігурують у породженні, називається **сентенціальною формою**, а останній рядок, що складається з терміналів, – **сентенцією** (пропозицією) мови.

При цьому

$\Rightarrow$ означає один крок,

$\overset{*}{\Rightarrow}$ нуль або більше кроків,

$\overset{+}{\Rightarrow}$ один або більше кроків.

Запис $\begin{cases} B \to yB, \\ B \to y \end{cases}$ можна записувати у більш стислому вигляді: $B \to yB \mid y$.

Ту ж саму мову можна згенерувати багатьма граматиками.

Наше визначення граматики допускає граматики і більш загальних типів.

Наприклад: $G_3=(\{a\}, \{S, N, Q, R\}, P, S)$, де P містить продукції:

$S \to QNQ,$

$Q \to QR,$

$$RQ \to NNQ,$$

…

Тут *N* переходить у *R*, якщо *N* стоїть після *Q*, а *R* переходить у *NN*, якщо воно стоїть перед *Q*.

У даному прикладі продукції є контекстозалежними**.** Контекстнонезалежні продукції мають порожній контекст.

Хомський визначив 4 класи граматик. Граматики 0-го типу, **загального вигляду**, або рекурсивно-перерелічувані визначаються як граматики, що відповідають нашому визначенню без обмежень на типи продукцій. Це найбільш загальний клас, інші граматики можуть бути отримані накладенням обмежень на продукції граматик 0-го типу. Граматики 0-го типу еквівалентні машинам Тьюринга.

Перше обмеження: задати, щоб для всіх продукцій $\alpha \to \beta$ довжина рядка α, обчислена в кількості символів, була не більше довжини рядка β. Граматики, усі продукції яких задовольняють дане обмеження, називаються граматиками 1-го типу, або **контекстозалежними**. Граматики 1-го типу еквівалентні лінійно-обмеженим автоматам.

Якщо, крім уже названого обмеження, у лівій частині продукції повинен знаходитися тільки один нетермінал, граматика називається граматикою 2-го типу або **контекстовільною** граматикою. Приклади G_1, G_2 представляють 2-й тип. У контекстовільних граматиках зручно дозволити продукцію $S \to \varepsilon$, хоча, строго кажучи, вона не дозволена навіть у контекстозалежних граматиках. Граматики 2-го типу еквівалентні магазинним автоматам (push-down).

Останній клас граматик – граматики 3-го типу, або **регулярні** (автоматні) граматики. Мова такої граматики називається регулярною. Регулярну мову можна визначити регулярним виразом і навпаки. Регулярні мови і регулярні вирази еквівалентні скінченним автоматам.

Ієрархія Хомського є включаючою (рис 3.1).

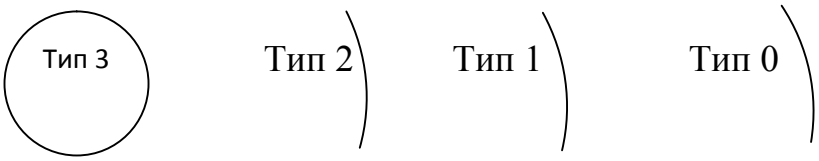

Рисунок 3.1 - Ієрархія Хомського для граматик

Формальні мови класифікуються відповідно до типів граматик, якими вони задаються. Однак та сама мова може бути задана різними граматиками, що належать до різних типів. У такому випадку вважається, що мова належить до найбільш простого з них. Так, мова, описана граматикою з фразовою структурою, контекстозалежною і контекстовільною граматиками, буде контекстовільною.

Найбільш складні – мови загального вигляду (сюди можна віднести природні мови), найпростіші – регулярні мови.

Теорема. Кожна контекстовільна мова породжується граматикою, усі продукції якої мають форму $A \to BP$ або $A \to a$, де *A, B, P* – змінні;*a* – термінал.

Ця форма називається нормальною формою Хомського.

Лема про розширення для контекстно-вільних мов. Нехай L – контекстовільна мова. Тоді існує така константа n, що якщо z – довільний ланцюжок довжини не менше n, то її можна подати у вигляді $z = uvwxy$, де

1) $|vwx| \leq n$;

2)$vx \neq \varepsilon$;

3) $uv^i wx^i y \in L$ для всіх $i \geq 0$.

Лема про розширення часто використовується при доведенні того, що мова не є контекстовільною.

Контекстовільні мови замкнені щодо операцій об'єднання, конкатенації, замикання і суперпозиції.

3.2 Розробка алгоритму роботи тренажера

Тема «Контекстовільні мови», як і вся дисципліна «Теорія програмування», є більш теоретично-орієнтованою. Але ця дисципліна важлива для підготовки програмістів. Студенту самостійно розібратися із темою «Контекстовільні мови» не

завжди просто. Для допомоги студентам і створюється навчальний тренажер. Цей тренажер буде частиною дистанційного курсу з дисципліни «Теорія програмування».

В дистанційному курсі з дисципліни «Теорія програмування» будуть зокрема розміщені інструкція, лекції, тести, контрольні роботи та інші матеріали. Оцінювання студента не є задачею навчального тренажера – для цього є інші елементи дистанційного курсу. Також тренажер не буде містити теоретичної інформації, яка вже доступна в лекціях.

Отже, основним завданням навчального тренажера з теми «Контекстовільні мови» дистанційного навчального курсу «Теорія програмування» допомогти студенту розібратися із відповідною темою.

Тема «Контекстовільні мови» містить багато математичних формул. Користувачу буде незручно вводити їх, використовуючи клавіатуру. Тому вибрано наступний формат роботи тренажера: на кожному етапі виводиться питання або завдання, на яке або дано варіанти відповіді, з яких потрібно вибрати вірну або вірні відповіді.

У випадку вибору певної відповіді студенту буде виводитися інформація про правильність відповіді (якщо відповідь правильна) або підказка (у випадку неправильної відповіді).

Завдання тренажера та відповідні підказки:

1) Граматики 0-го типу за класифікацією Хомського еквівалентні:

а) машині Тьюрінга;

б) лінійно-обмеженим автоматам;

в) магазинним автоматам;

г) регулярним виразам.

Правильна відповідь: а) машині Тьюрінга.

Підказки:

а) Так, це правильна відповідь!

б) Ні, граматики 1-го типу еквівалентні лінійно-обмеженим автоматам

в) Ні, граматики 2-го типу еквівалентні магазинним автоматам

г) Ні, граматики 3-го типу еквівалентні регулярним виразам

2) Граматики 1-го типу за класифікацією Хомського еквівалентні:

а) машині Тьюрінга;

б) лінійно-обмеженим автоматам;

в) магазинним автоматам;

г) регулярним виразам.

Правильна відповідь: б) лінійно-обмеженим автоматам.

Підказки:

а) Ні, граматики 0-го типу еквівалентні машині Тьюрінга

б) Так, це правильна відповідь!

в) Ні, граматики 2-го типу еквівалентні магазинним автоматам

г) Ні, граматики 3-го типу еквівалентні регулярним виразам

3) Граматики 2-го типу за класифікацією Хомського еквівалентні:

а) машині Тьюрінга;

б) лінійно-обмеженим автоматам;

в) магазинним автоматам;

г) регулярним виразам.

Правильна відповідь: в) магазинним автоматам.

Підказки:

а) Ні, граматики 0-го типу еквівалентні машині Тьюрінга

б) Ні, граматики 1-го типу еквівалентні лінійно-обмеженим автоматам

в) Так, це правильна відповідь!

г) Ні, граматики 3-го типу еквівалентні регулярним виразам

4) Граматики 3-го типу за класифікацією Хомського еквівалентні:

а) машині Тьюрінга;

б) лінійно-обмеженим автоматам;

в) магазинним автоматам;

г) регулярним виразам.

Правильна відповідь: г) регулярним виразам.

Підказки:

а) Ні, граматики 0-го типу еквівалентні машині Тьюрінга

б) Ні, граматики 1-го типу еквівалентні лінійно-обмеженим автоматам

в) Ні, граматики 2-го типу еквівалентні магазинним автоматам

г) Так, це правильна відповідь!

5) Контекстовільні граматики еквівалентні:

а) машині Тьюрінга;

б) лінійно-обмеженим автоматам;

в) магазинним автоматам;

г) регулярним виразам.

Правильна відповідь: в) магазинним автоматам.

Підказки:

а) Ні, граматики загального вигляду еквівалентні машині Тьюрінга

б) Ні, контекстозалежні граматики еквівалентні лінійно-обмеженим автоматам

в) Так, це правильна відповідь!

г) Ні, регулярні граматики еквіваленті регулярним виразам

6. Граматиками загального вигляду називають

а) граматики 0-го типу (за класифікацією Хомського);

б) граматики 1-го типу (за класифікацією Хомського);

в) граматики 2-го типу (за класифікацією Хомського);

г) граматики 3-го типу (за класифікацією Хомського).

Правильна відповідь: а) граматики 0-го типу (за класифікацією Хомського).

Підказки:

а) Так, це правильна відповідь!

б) Ні, граматики 1-го типу називають контекстозалежними граматиками

в) Ні, граматики 2-го типу називають контекстовільними граматиками

г) Ні, граматики 3-го типу називають регулярними граматиками

7. Контекстозалежними граматиками називають

а) граматики 0-го типу (за класифікацією Хомського);

б) граматики 1-го типу (за класифікацією Хомського);

в) граматики 2-го типу (за класифікацією Хомського);

г) граматики 3-го типу (за класифікацією Хомського).

Правильна відповідь: б) граматики 1-го типу (за класифікацією Хомського).

Підказки:

а) Ні, граматики 0-го типу називають граматиками загального вигляду

б) Так, це правильна відповідь!

в) Ні, граматики 2-го типу називають контекстовільними граматиками

г) Ні, граматики 3-го типу називають регулярними граматиками

8. Контекстовільними граматиками називають

а) граматики 0-го типу (за класифікацією Хомського);

б) граматики 1-го типу (за класифікацією Хомського);

в) граматики 2-го типу (за класифікацією Хомського);

г) граматики 3-го типу (за класифікацією Хомського).

Правильна відповідь: в) граматики 2-го типу (за класифікацією Хомського).

Підказки:

а) Ні, граматики 0-го типу називають граматиками загального вигляду

б) Ні, граматики 1-го типу називають контекстозалежними граматиками

в) Так, це правильна відповідь!

г) Ні, граматики 3-го типу називають регулярними граматиками

9. Регулярними граматиками називають

а) граматики 0-го типу (за класифікацією Хомського);

б) граматики 1-го типу (за класифікацією Хомського);

в) граматики 2-го типу (за класифікацією Хомського);

г) граматики 3-го типу (за класифікацією Хомського).

Правильна відповідь: г) граматики 3-го типу (за класифікацією Хомського).

Підказки:

а) Ні, граматики 0-го типу називають граматиками загального вигляду

б) Ні, граматики 1-го типу називають контекстозалежними граматиками

в) Ні, граматики 2-го типу називають контекстовільними граматиками

г) Так, це правильна відповідь!

10. Якщо мова може бути задана кількома граматиками, що відносяться до різних типів, то мова буде класифікована відповідно до ...

а) найпростішої граматики;

б) найскладнішої граматики;

в) може бути віднесена до будь-якого типу;

г) така ситуація неможлива.

Правильна відповідь: а) найпростішої граматики.

Підказки:

а) Так, це правильна відповідь!

б) Відповідь неправильна

в) Ні, граматики класифікуються однозначно

г) Така ситуація трапляється постійно

11. Якщо мова може бути задана контекстовільною і контекстозалежною граматиками, то вона буде класифікована як ...

а) контекстозалежна;

б) контекстовільна;

в) може бути віднесена до будь-якого типу;

г) така ситуація неможлива.

Правильна відповідь: б) контекстовільна.

Підказки:

а) Відповідь неправильна! Якщо мова може бути задана кількома граматиками, що відносяться до різних типів, то мова буде класифікована відповідно до найпростішої граматики

б) Так, це правильна відповідь!

в) Ні, граматики класифікуються однозначно

г) Така ситуація трапляється часто

12. Якщо мова може бути задана контекстовільною і регулярною граматиками, то вона буде класифікована як ...

а) регулярна;

б) контекстовільна;

в) може бути віднесена до будь-якого типу;

г) така ситуація неможлива.

Правильна відповідь: а) регулярна.

Підказки:

а) Так, це правильна відповідь!

б) Відповідь неправильна! Якщо мова може бути задана кількома граматиками, що відносяться до різних типів, то мова буде класифікована відповідно до найпростішої граматики

в) Ні, граматики класифікуються однозначно

г) Така ситуація трапляється часто

13. Виберіть операції, відносно яких контекстовільні мови є замкненими (множинний вибір):

а) об'єднання;

б) конкатенації;

в) замикання;

г) суперпозиції.

Правильні відповіді: всі.

Підказки:

а) Контекстовільні мови є замкненими операції об'єднання

б) Контекстовільні мови є замкненими операції конкатенації

в) Контекстовільні мови є замкненими операції замикання

г) Контекстовільні мови є замкненими операції суперпозиції

14. Якою із граматик може бути описана мова $\{ x^n y^n \mid n>0\}$?

а) $G_1=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to xSy, S \to xy \}$.

б) $G_2=(\{x,z\}, \{S\}, P, S)$, де $P = \{ S \to xSz, S \to xz \}$.

в) $G_3=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to xSy, S \to yx \}$.

г) $G_4=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to ySx, S \to xy \}$.

Правильна відповідь: *а)* $G_1=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to xSy, S \to xy \}$.

Підказки:

а) Відповідь правильна

б) Відповідь неправильна

в) Відповідь неправильна

г) Відповідь неправильна

15. Якою із граматик може бути описана мова $\{ 0^m 1^n \mid m,n>0,1,2,...\}$?

а) $G_1=(\{0\}, \{S\}, P, S)$, де $P = \{ S\to 0S1, S\to \varepsilon \}$.

б) $G_2=(\{0,1\}, \{S\}, P, S)$, де $P = \{ S\to 0, S\to 1, S\to \varepsilon \}$.

в) $G_3=(\{0\}, \{S\}, P, S)$, де $P = \{ S\to 0S, S\to S1, S\to \varepsilon \}$.

г) $G_4=(\{0,1\}, \{S\}, P, S)$, де $P = \{ S\to 0S, S\to S1, S\to \varepsilon \}$.

Правильна відповідь: *г)* $G_4=(\{0,1\}, \{S\}, P, S)$, де $P = \{ S\to 0S, S\to S1, S\to \varepsilon \}$.

Підказки:

а) Відповідь неправильна

б) Відповідь неправильна

в) Відповідь неправильна

г) Відповідь правильна

16. Чи є мова $\{ x^n y^n \mid n>0\}$ контекстовільною?

а) так;

б) ні.

Правильна відповідь: а) так.

Підказки:

а) Відповідь правильна

б) Відповідь неправильна

17. Чи є мова $\{ a^m b^n \mid m,n>0,1,2,...\}$ контекстовільною?

а) так;

б) ні.

Правильна відповідь: б) ні.

Підказки:

а) Відповідь неправильна

б) Відповідь правильна

18. Мова $\{ x^n y^n \mid n>0\}$ ...

а) регулярна;

б) контекстовільна;

в) контекстозалежна.

Правильна відповідь: б) контекстовільна.

Підказки:

а) Відповідь неправильна

б) Відповідь правильна

в) Відповідь неправильна

19. Мова $\{a^m b^n \mid m,n>0,1,2,...\}$ ...

а) регулярна;

б) контекстовільна;

в) контекстозалежна.

Правильна відповідь: а) регулярна.

Підказки:

а) Відповідь правильна

б) Відповідь неправильна

в) Відповідь неправильна

Враховуючи те, що деякі питання є однотипними, всі питання розбиті на групи, відповідно до таблиці 3.2. Із кожної групи вибирається випадковим чином лише певна кількість питань (стовпець 4 таблиці). Якщо користувач під час проходження тренажера робить помилку, то із відповідної групи додається ще одне питання, яке буде показане в кінці (зрозуміло, лише в тому випадку, коли такі питання є).

Таким чином, під час проходження тренажера користувач може відповідати на різну кількість питань, в залежності від того робить він помилки чи ні. Ця кількість може змінюватися від 11 до 19.

Тобто, якщо студент під час проходження тренажера правильно відповість на всі питання з першого разу, то він зможе завершити роботу швидше, адже йому доведеться відповідати лише на 11 запитань.

І навпаки – якщо студент багато буде помилятися, то від буде працювати із тренажером довше.

Таблиця 3.2 Групи запитань

Номер групи	Номера питань групи	Кількість питань в групі	Кількість обов'язкових питань	Максимальна кількість додаткових питань
1	1-4	4	2	2
2	5	1	1	0
3	6-9	4	2	2
4	10-12	3	2	1
5	13	1	1	0
6	14-15	2	1	1
7	16-19	4	2	2

Після успішного проходження всіх кроків підраховується відсоток помилок, які допустив користувач. В залежності від цього користувач отримує повідомлення.

Якщо відсоток правильних відповідей більше 90, то виводиться повідомлення "Вітаємо! Ви успішно пройшли тренажер".

Якщо правильних відповідей від 75% до 90%, то виводиться повідомлення "Вітаємо! Ви успішно пройшли тренажер. Матеріал забувається, тому згодом пройдіть його ще раз".

Якщо правильних відповідей від 60% до 75%, то виводиться повідомлення "Вітаємо! Ви пройшли тренажер. Матеріал забувається, тому згодом пройдіть його ще раз".

Якщо правильних відповідей менше 60%, то користувач отримує повідомлення "Вітаємо! Ви пройшли тренажер. Повторіть матеріал та пройдіть його ще раз".

Алгоритм роботи тренажера

Крок 1. Запускається тренажер, на головному вікні виводиться основна інформація про цей тренажер. Користувачу пропонується почати роботу.

Крок 2. Виводиться завдання.

Крок 3. Номер етапу встановлюється рівним нулю.

Крок 4. Відбувається перевірка: чи номер поточного етапу не став рівний номеру останнього етапу. Якщо так, то перехід на крок 9. Якщо ні, то перехід на наступний крок.

Крок 5. Виводиться питання поточного етапу

Крок 6. Користувач вводить відповідь на питання поточного кроку.

Крок 7. Відповідь перевіряється на правильність. Якщо відповідь правильна, то номер етапу збільшується на 1, перехід на крок 5. Якщо відповідь неправильна, то перехід на наступний крок.

Крок 8. Виводиться довідкова інформація про допущену помилку. Додається ще одне питання із відповідної групи, якщо такі питання ще є. Кількість етапів збільшується на 1. Перехід на крок 5.

Крок 9. Виводиться інформація про успішне проходження тренажера. Завершення роботи.

Блок-схема алгоритму зображена на рис. 3.2.

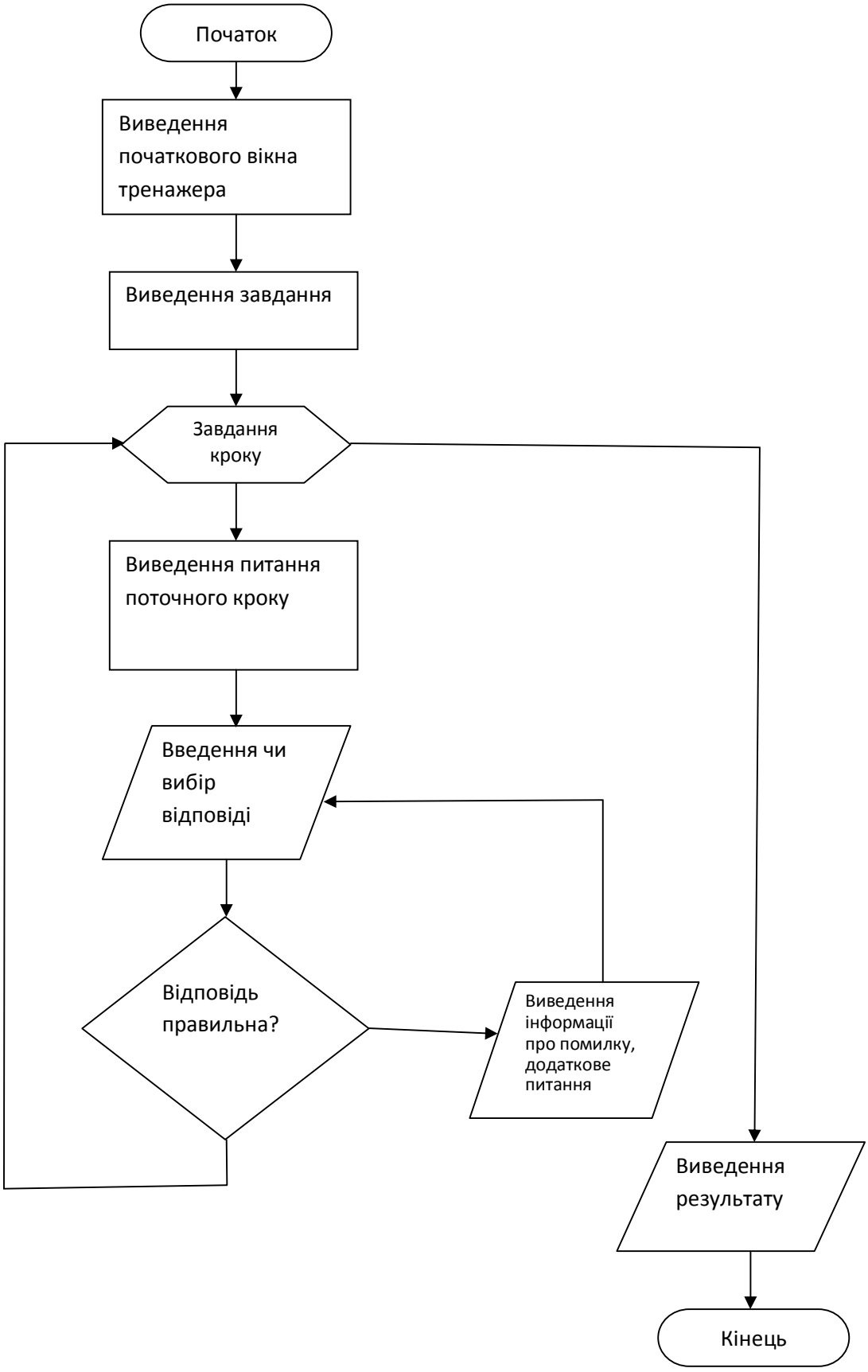

Рис 3.2 Блок-схема алгоритму роботи тренажера

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ

4.1 Розробка тренажера

Для розробки тренажера використовувалася мова програмування Java. Тренажер створювався в інтегрованому середовищі розробки IntelliJ IDEA Community. Проект має наступну структуру (рис. 4.1).

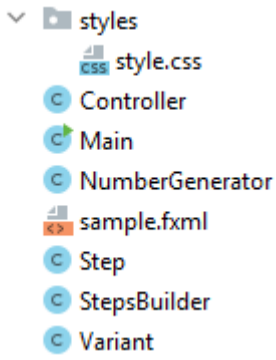

Рис 4.1 Структура проекту

У файлі **style.css**, який знаходиться у папці styles, описані стилі, що визначають зовнішній вигляді тренажера. Наприклад, наступним чином задані стилі для кнопок (button):

```
.button {
    -fx-background-color:
        linear-gradient(#f0ff35, #a9ff00),
        radial-gradient(center 50% -40%, radius 200%, #b8ee36 45%,
#80c800 50%);
    -fx-background-radius: 6, 5;
    -fx-background-insets: 0, 1;
    -fx-effect: dropshadow( three-pass-box , rgba(0,0,0,0.4) , 5, 0.0
, 0 , 1 );
    -fx-text-fill: #395306;
}
```

Стилі для всього вікна визначаються таким блоком:

```
.root {
  -fx-font-size: 16pt ;
```

```
  -fx-background-color:#B8860B;
}
```

У класі **Controller** визначається взаємодія графічного інтерфейсу користувача та логіки роботи програми.

Розглянемо методи цього класу. Метод start викликається при початку проходження тренажера. Цей метод приховує стартове повідомлення та виводить перше запитання із списку за допомогою методу showQuestion. Код методу наведений нижче.

```
public void start(ActionEvent actionEvent) {
    root.getChildren().remove(0);
    root.getChildren().add(taskOneAnswer);
    showQuestion(stepsBuilder.getStep(listNumbers.get(0)));
}
```

Метод showQuestion безпосередньо виводить запитання певного кроку, який передається йому в якості параметру. Метод перевіряє, скільки правильних відповідей є на даному кроці. Якщо правильна відповідь одна, то використовується шаблон, в якому можна вибрати лише одну відповідь. Якщо правильних відповідей кілька, то використовується шаблон, що допускає кілька відповідей.

Метод getAnswer викликається для зчитування відповідей, які користувач відмітив як вірні. Цей метод повертає масив цілих чисел – номерів вибраних відповідей на певному кроці. Якщо відповідь вибрана лише одна, то цей масив містить лише одне число.

Метод answerClick викликається, коли користувач натискає кнопку "Відповісти".

Метод initialize викликається при запуску тренажера. Він відповідає за вивід інформації на стартовому вікні.

Клас **Main** відповідає за початок роботи тренажера. Зокрема, цей клас містить метод start, в якому задаються розміри вікна програми, заголовок та файл із fxml-розміткою. Код методу:

```
public void start(Stage primaryStage) throws Exception{
    Parent root =
FXMLLoader.load(getClass().getResource("sample.fxml"));
    primaryStage.setTitle("Тренажер Контекстовільні мови");
    primaryStage.setScene(new Scene(root, 600, 575));
    primaryStage.show();
}
```

Клас NumberGenerator призначений для генерації номерів запитань, які будуть виводиться користувачу. Номера питань генеруються випадковим чином, відповідно до таблиці 3.2. Генерація відбувається в конструкторі класу.

Метод addNumber додає новий номер до списку. Цей метод викликається в тому випадку, коли користувач дає неправильну відповідь на одне із питань. Код методу наведений нижче.

```
public void addNumber(int mistakeNum){
    if(mistakeNum<=3){
        if(!listPart1.isEmpty()){
            list.add(listPart1.remove(0));
        }
    }else if(mistakeNum>=5 && mistakeNum<=8){
        if(!listPart3.isEmpty()){
            list.add(listPart3.remove(0));
        }
    }else if(mistakeNum>=9 && mistakeNum<=11){
        if(!listPart4.isEmpty()){
            list.add(listPart4.remove(0));
        }
    }else if(mistakeNum>=13 && mistakeNum<=14){
        if(!listPart6.isEmpty()){
            list.add(listPart6.remove(0));
        }
    } else if(mistakeNum>=15&&mistakeNum<=18){
        if(!listPart7.isEmpty()){
            list.add(listPart7.remove(0));
        }
    }
}
```

Метод getListNumbers повертає список номерів.

Файл sample.fxml містить розмітку вікна тренажера у форматі FXML. Цей файл складається з блоків коду, кожен із яких відповідає за ту чи іншу частину

вікна. Ці блоки можуть динамічно активуватися чи деактивуватися під час роботи програми. Наприклад, блок, що відповідає за стартове вікно має наступний вигляд:

```
<VBox fx:id="helloBox" spacing="10" alignment="CENTER" >
    <Label text='Тренажер з теми "Контекстовільні мови" '
textAlignment="CENTER" styleClass="title"></Label>
    <Label text='дисципліни "Теорія програмування" '
textAlignment="CENTER" styleClass="title"></Label>
    <Label text="Розробила Лєбєдєва Марія " textAlignment="CENTER"
></Label>
    <Label text='студентка спеціальності "Комп&amp;apos;ютерні науки" '
textAlignment="CENTER" ></Label>
    <Label text="Полтавського університету економіки і торгівлі"
textAlignment="CENTER" ></Label>
    <Label text="Керівник: к.ф.-м.н., доцент Черненко О.О."
textAlignment="CENTER" ></Label>
    <Button fx:id="startButton" onAction="#start" text="Почати
роботу"></Button>
</VBox>
```

Блок, що відповідає за показ питань з кількома допустимими варіантами відповіді має такий вигляд:

```
<VBox fx:id="taskMultyAnswer" spacing="15">
    <TextArea fx:id="question" text="Питання"></TextArea>
    <CheckBox fx:id="check1" text="відповідь 1"/>
    <CheckBox fx:id="check2" text="відповідь 2"/>
    <CheckBox fx:id="check3" text="відповідь 3"/>
    <CheckBox fx:id="check4" text="відповідь 4"/>
    <Button text="Відповісти" onAction="#answerClick"/>
</VBox>
```

Питання, що допускають лише один варіант відповіді, показуються за допомогою наступного блоку:

```
<VBox fx:id="taskOneAnswer" spacing="15">
    <TextArea fx:id="question2" text="Питання"></TextArea>
    <RadioButton fx:id="radio1" text="відповідь 1" >
        <toggleGroup>
            <ToggleGroup fx:id="toggleGroup"/>
        </toggleGroup>
    </RadioButton>
    <RadioButton fx:id="radio2" toggleGroup="$toggleGroup"
text="відповідь 2"/>
```

```
    <RadioButton fx:id="radio3" toggleGroup="$toggleGroup"
text="відповідь 3"/>
    <RadioButton fx:id="radio4" toggleGroup="$toggleGroup"
text="відповідь 4"/>
    <Button text="Відповісти" onAction="#answerClick"/>
</VBox>
```

Клас **Step** описує один крок тренажера, тобто одне питання. Це клас містить такі поля:

1) id – номер питання;

2) question – питання поточного кроку;

3) variantList – список варіантів відповідей та відповідних підказок, що показуються при виборі користувачем відповідної відповіді;

4) correctNumbers – масив номерів правильних відповідей. У випадку, коли питання допускає лише одну правильну відповідь, цей масив містить лише одне число.

Метод checkAnswer цього класу перевіряє, чи правильну відповідь обрав користувач.

```
public boolean checkAnswer(int [] numbers){
    if(numbers.length!=correctNumbers.length){
        return false;
    }
    for(int i=0;i<numbers.length;i++){
        if(numbers[i]!=correctNumbers[i]){
            return false;
        }
    }
    return true;
}
```

Клас **StepsBuilder** призначений для формування кроків тренажеру. Всі можливі питання із варіантами відповідей формуються у конструкторі цього класу. Наприклад, наступний код формує перше питання.

```
question="Граматики 0-го типу за класифікацією Хомського
еквівалентні:";
step=new Step(0,question);
int [] corectNumbers1={0};
```

```
step.setCorrectNumbers(corectNumbers1);
//a
var="машині Тьюрінга";
tip="Так, це правильна відповідь!";
variant=new Variant(var,tip);
step.addVariant(variant);
//b
var="лінійно-обмеженим автоматам";
tip="Ні, граматики 1-го типу еквівалентні лінійно-обмеженим
автоматам";
variant=new Variant(var,tip);
step.addVariant(variant);
//c
var="магазинним автоматам";
tip="Ні, граматики 2-го типу еквівалентні магазинним автоматам";
variant=new Variant(var,tip);
step.addVariant(variant);
//d
var="регулярним виразам";
tip="Ні, граматики 3-го типу еквівалентні регулярним виразам";
variant=new Variant(var,tip);
step.addVariant(variant);
steps.add(step);
```

Метод getStep повертає крок із відповідним номером, що передається методу в якості параметру.

```
public Step getStep(int index){
    return this.steps.get(index);
}
```

Клас Variant описує один варіант відповіді та відповідну підказку, що отримує користувач при виборі цього варіанту.

```
public class Variant {
    private  String variant;
    private String tip;
    public Variant(String variant, String tip) {
        this.variant = variant;
        this.tip = tip;
    }
    public String getVariant() {
        return variant;
    }
    public String getTip() {
        return tip;
```

```
        }
}
```

Повний код навчального тренажера розміщений в додатку.

4.2 Тестування тренажера

Після запуску тренажера з'являється стартове вікно, що містить основну інформацію про тренажер (рис. 4.2). Розмір вікна підібраний таким чином, щоб нормально відображалися елементи на всіх кроках. Але за потреби користувач може легко змінити розмір, використовуючи стандартні дії операційної системи, або розгорнути програму на весь екран.

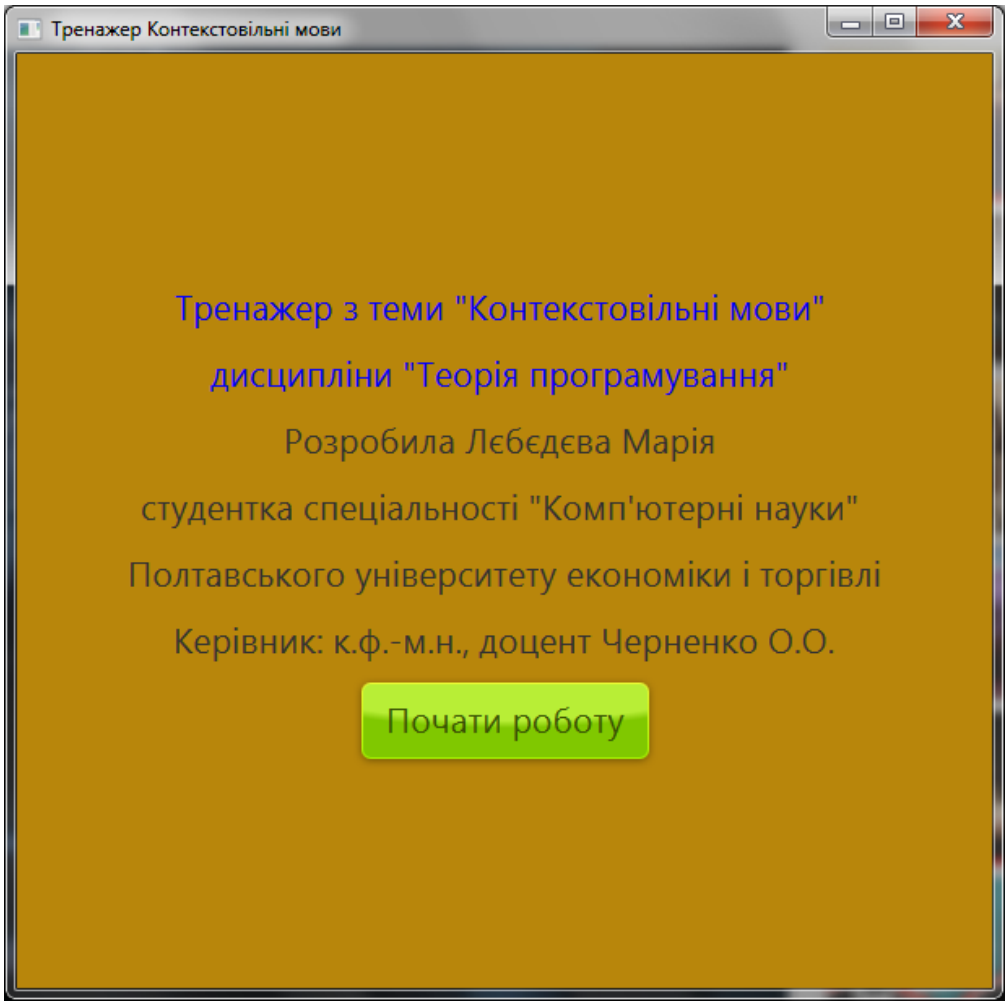

Рис. 4.2 Стартове вікно тренажера

Для початку роботи користувач повинен натиснути відповідну кнопку. Після цього з'являється вікно із питанням (рис. 4.3).

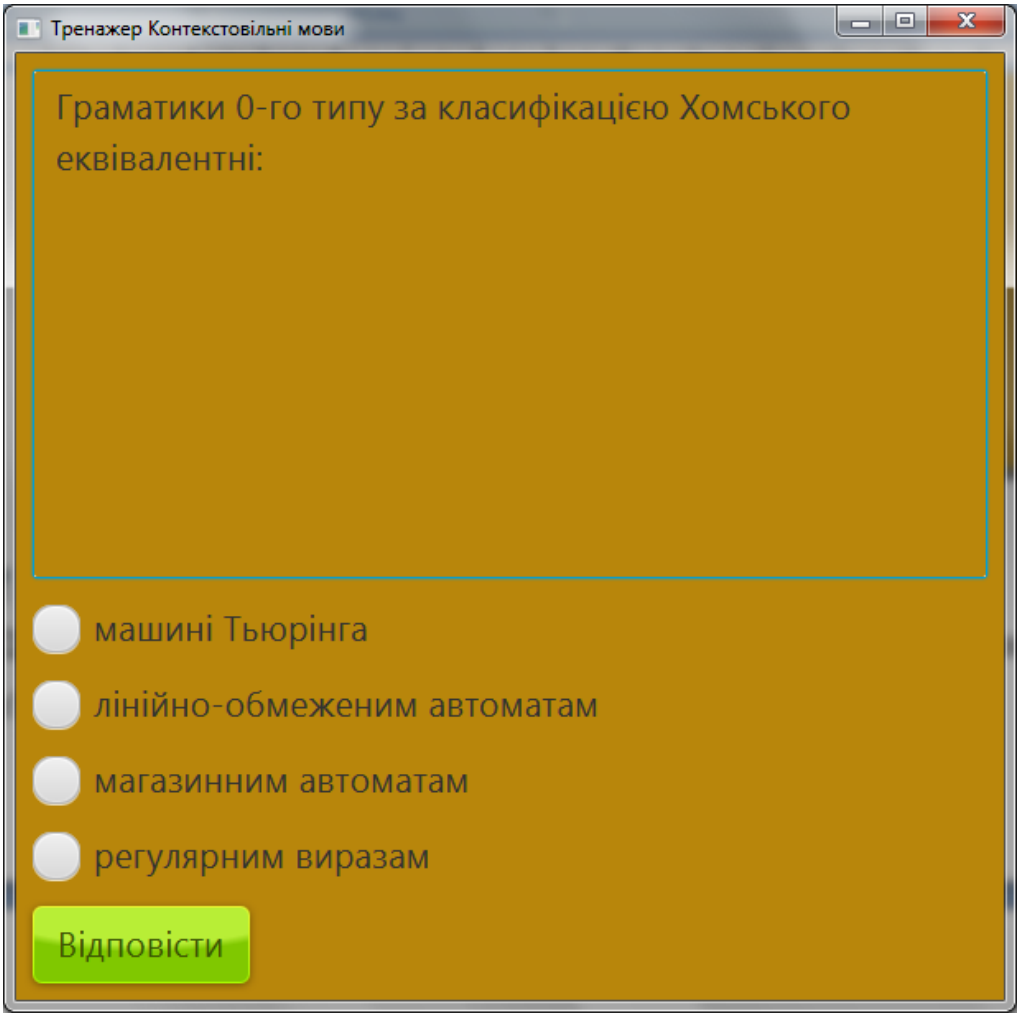

Рис. 4.3 Перше питання тренажера

Саме питання знаходиться вгорі вікна і виділене рамкою. В ніжній частині вікна розміщені варіанти відповіді, під ними – кнопка "Відповісти". Користувач має вибрати варіант відповіді і натиснути кнопку "Відповісти". Якщо вибрати неправильну відповідь, то з'являється діалогове вікно із повідомленням, що відповідь неправильна. Крім цього діалогове вікно містить інформацію, чому саме відповідь неправильна.

Наприклад, якщо вибрати варіант відповіді "магазинним автоматам", то виведеться підказка такого змісту: "Ні, граматики 2-го типу еквівалентні магазинним автоматам" (рис. 4.4).

Користувач має натиснути кнопку "OK", або просто закрити це діалогове вікно. Після цього він знову має відповідати на поставлене запитання.

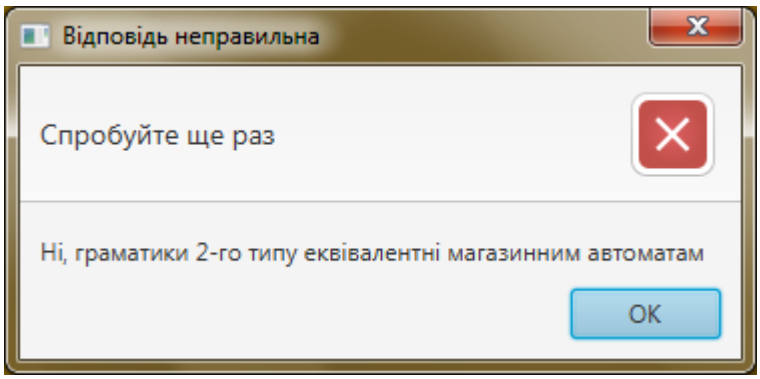

Рис. 4.4 Діалогове вікно із підказкою

Якщо знову дати неправильну відповідь: "лінійно-обмеженим автоматам", то знову з'явиться діалогове вікно (рис. 4.5). Цього разу воно буде містити такий текст: "Ні, граматики 1-го типу еквівалентні лінійно-обмеженим автоматам".

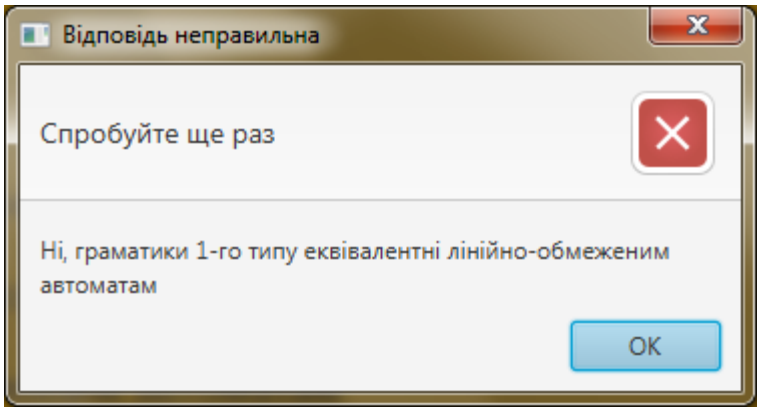

Рис. 4.5 Діалогове вікно із іншою підказкою

Користувач знову має закрити це вікно та спробувати ще раз. Якщо дати правильну відповідь, то з'явиться діалогове вікно з повідомленням про те, що відповідь правильна (рис. 4.6).

Для переходу до наступного питання користувач має закрити діалогове вікно.

Наступне питання (рис. 4.7) аналогічне до попереднього, оскільки вони знаходяться в одній групі, відповідно до табл. 3.2.

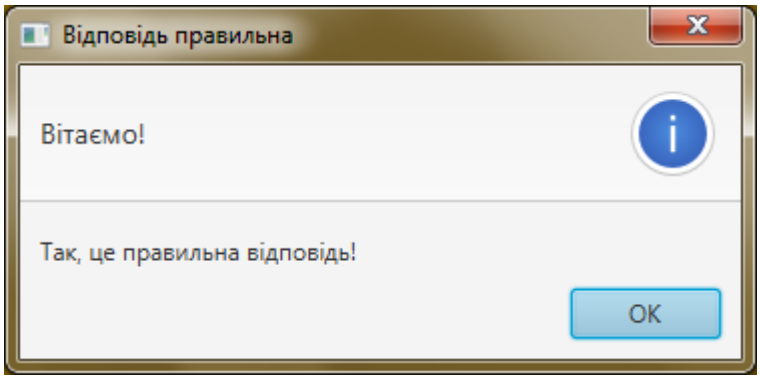

Рис. 4.6 Діалогове вікно після правильної відповіді

У випадку правильної чи хибної відповіді з'являються діалогові вікна аналогічні до діалогових вікон з рис 4.4-4.6.

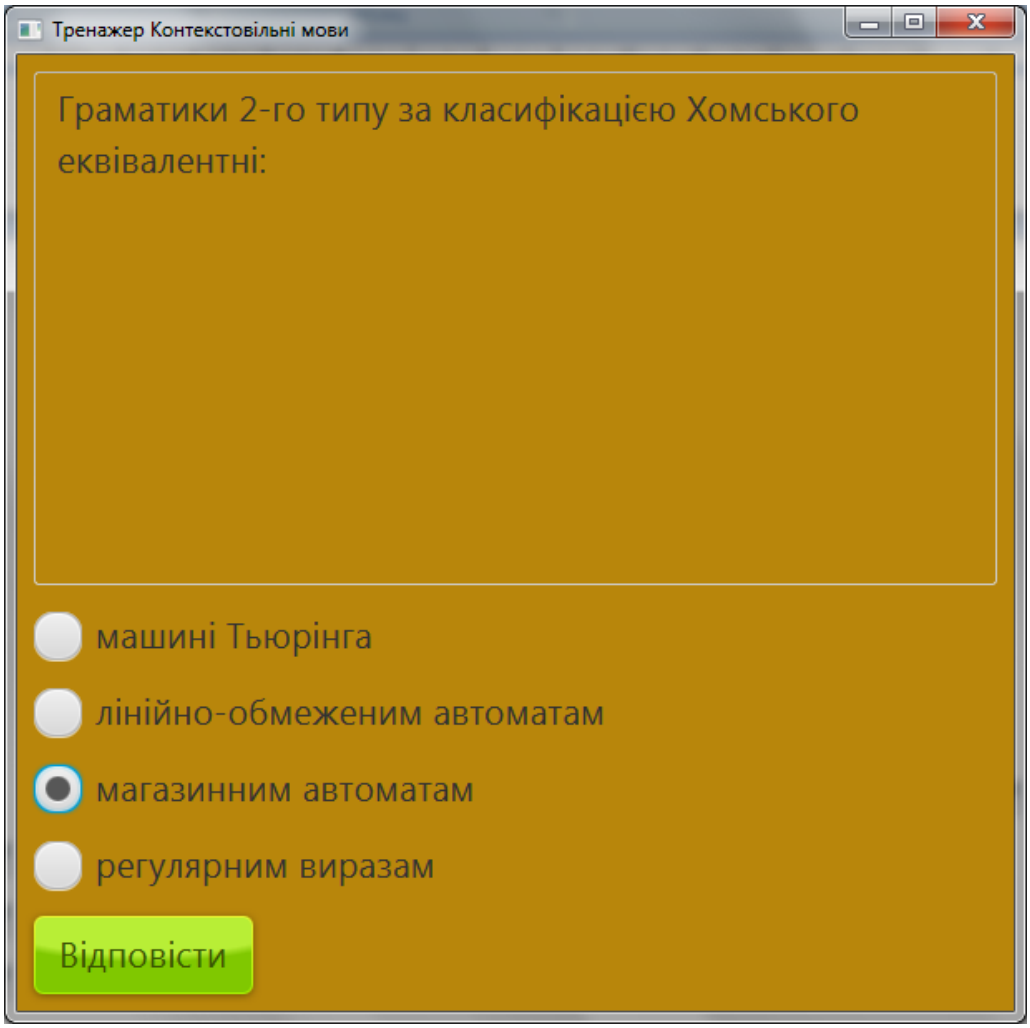

Рис. 4.7 Питання №2

На рис. 4.8 показане питання №3.

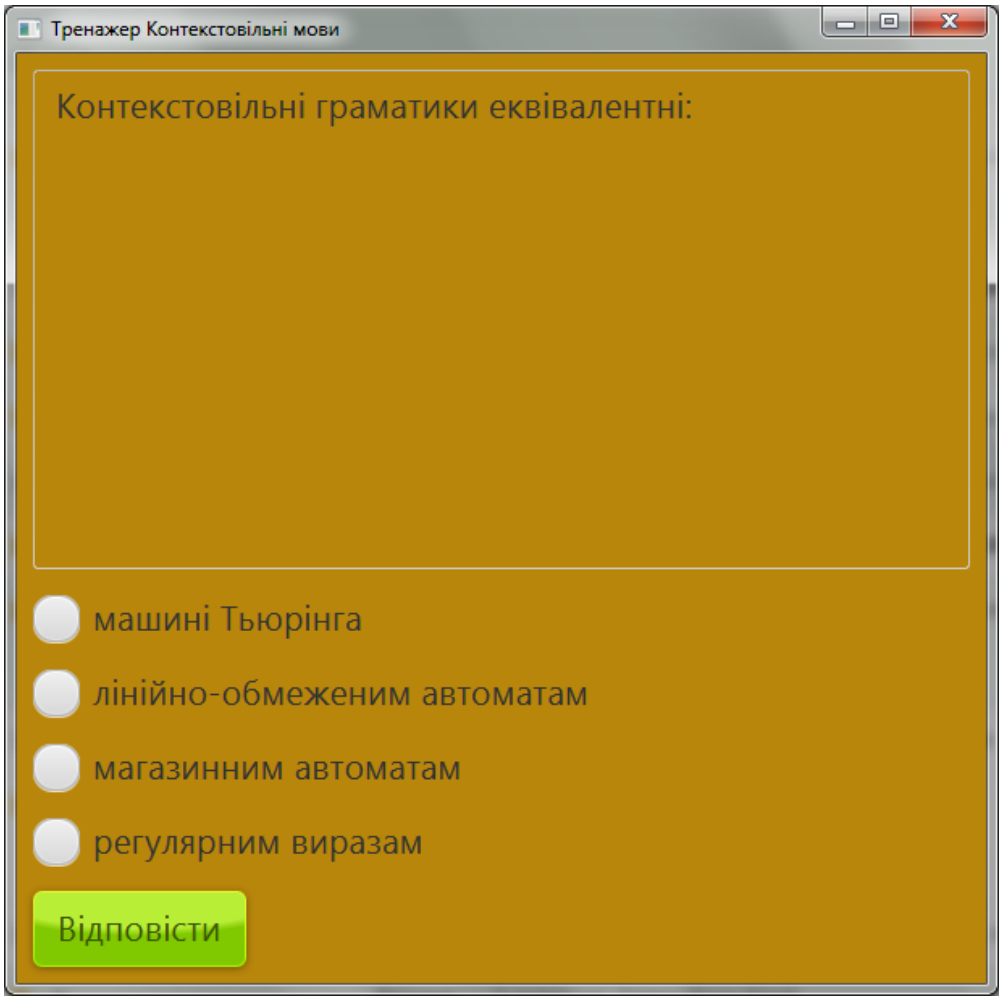

Рис. 4.8 Питання №3

Якщо вибрати відповідь "регулярним виразам", яка є неправильною, то отримаємо діалогове вікно із повідомленням: "Ні, регулярні граматики еквівалентні регулярним виразам".

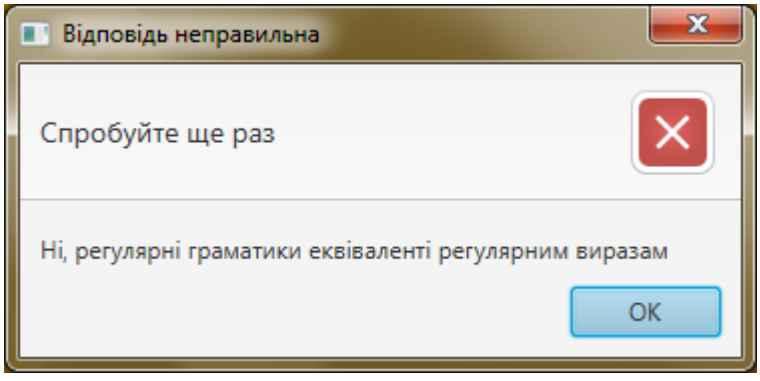

Рис. 4.9 Підказка при неправильній відповіді на питання №3

Після правильної відповіді з'являється повідомлення аналогічне до рис. 4.6 відбувається перехід до наступного питання (рис. 4.10).

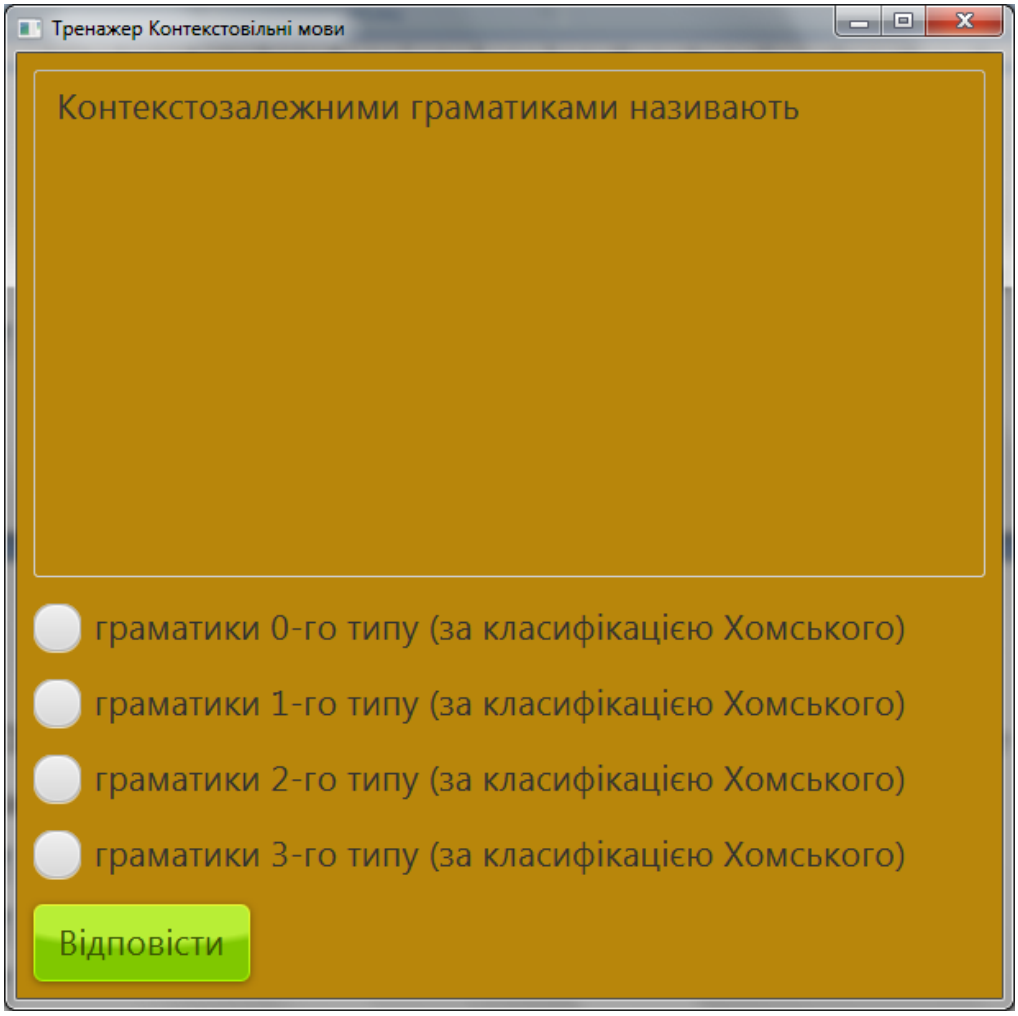

Рис. 4.10 Питання №4

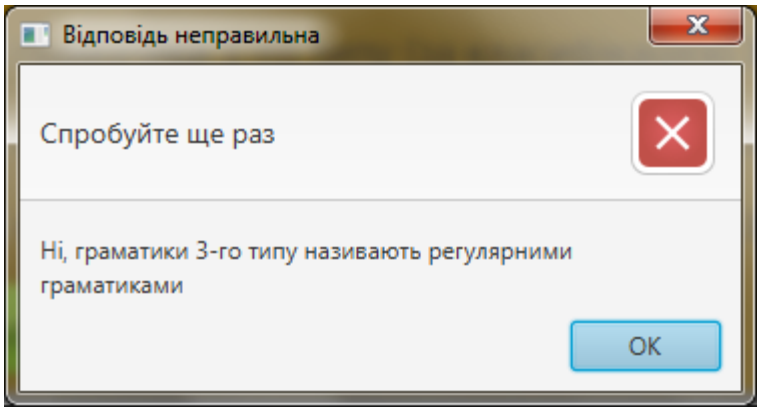

Рис. 4.11 Підказка до питання №4

Якщо вибрати неправильну відповідь "граматики 3-го типу (за класифікацією Хомського)", то з'явиться діалогове вікно із текстом: "Ні, граматики третього типу називають регулярними граматиками" (рис. 4.11).

Якщо вибрати неправильну відповідь "граматики 0-го типу (за класифікацією Хомського)", то отримаємо повідомлення: "Ні, граматики 0-го типу називають граматиками загального вигляду" (рис. 4.12).

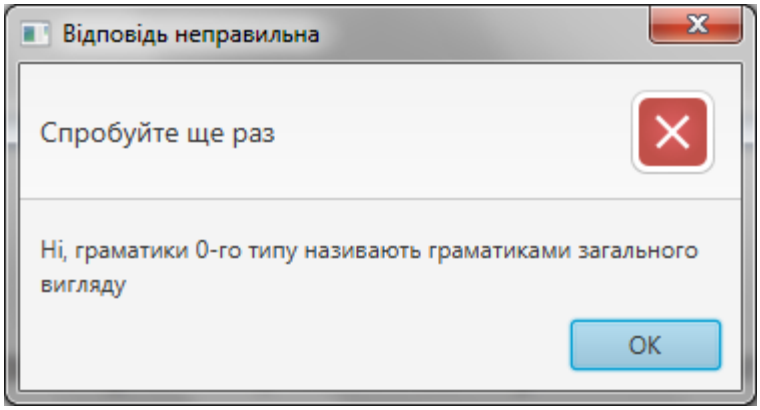

Рис. 4.12 Інша підказка до питання №4

Якщо вибрати неправильну відповідь "граматики 2-го типу (за класифікацією Хомського)", то отримаємо повідомлення: "Ні, граматики 2-го типу називають контекстовільними граматиками " (рис. 4.13).

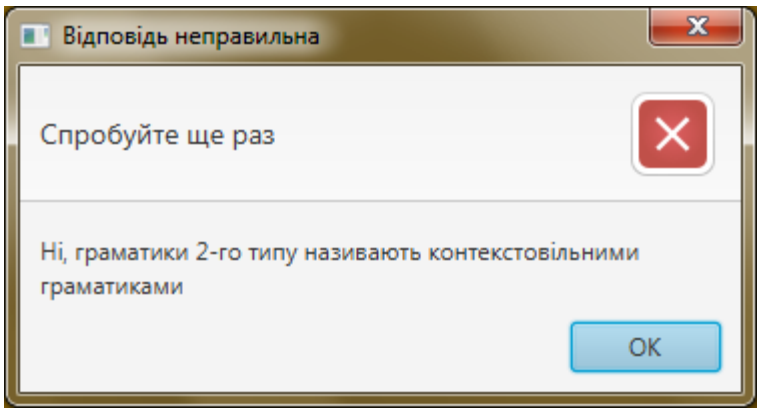

Рис. 4.13 Ще одна підказка до питання №4

В разі правильної відповіді з'являється діалогові вікно, що інформує про правильність відповіді. Після закриття якого відбувається перехід до наступного

запитання (рис. 4.14). Це питання з групи №3 (табл. 3.2), тому воно аналогічне до попереднього. Підказки, які отримує користувач у разі неправильних відповідей, також аналогічні до підказок з рис. 4.11 -4.13.

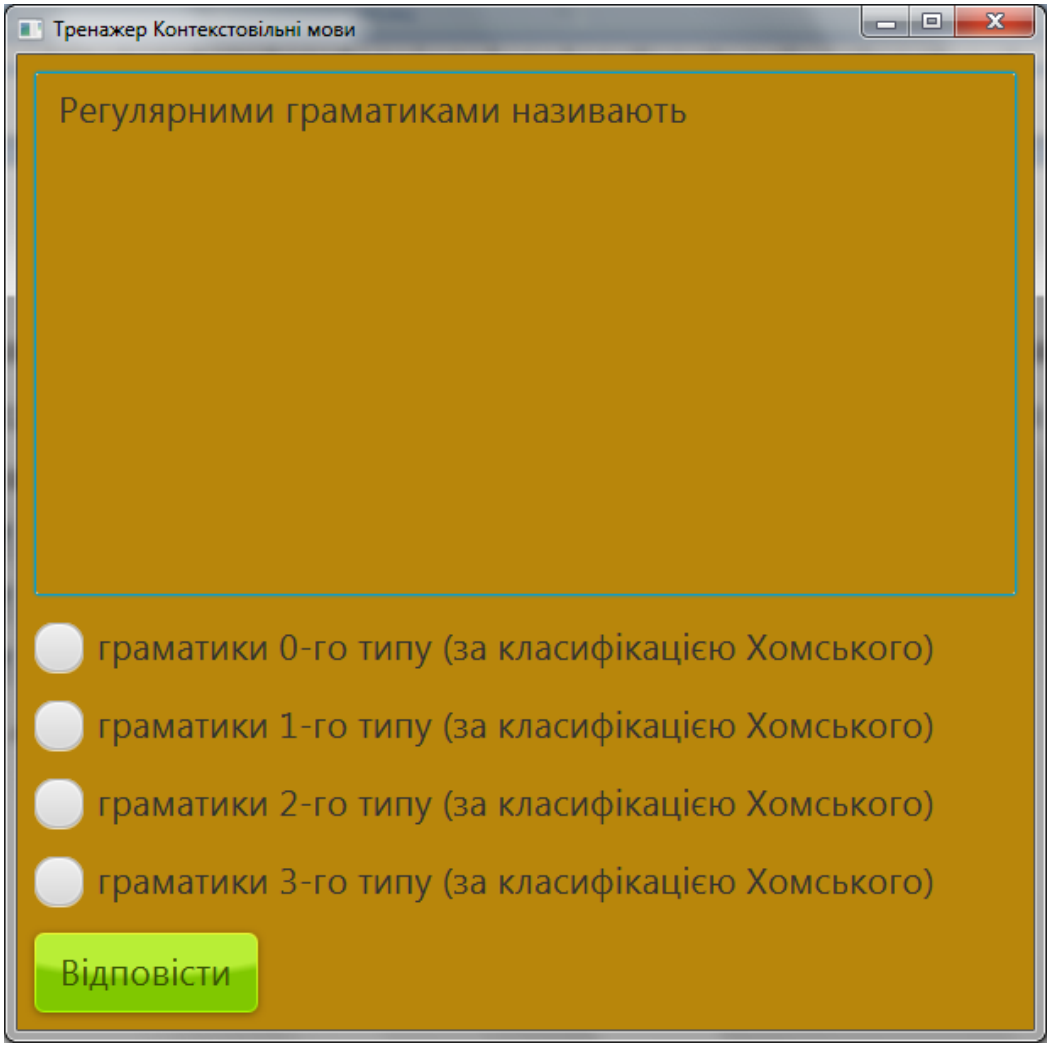

Рис. 4.14 Питання №5

Наступне питання відноситься до групи №4 і формулюється так: "Якщо мова може бути задана кількома граматиками, що відносяться до різних типів, то мова буде класифікована відповідно до ..." (рис. 4.15).

Якщо вибрати відповідь "така ситуація неможлива", то отримаємо підказку: "Така ситуація трапляється постійно" (рис. 4.16).

Якщо вибрати відповідь "може бути віднесена до будь-якого типу", то підказка буде така: "Ні, граматики класифікуються однозначно" (рис. 4.17).

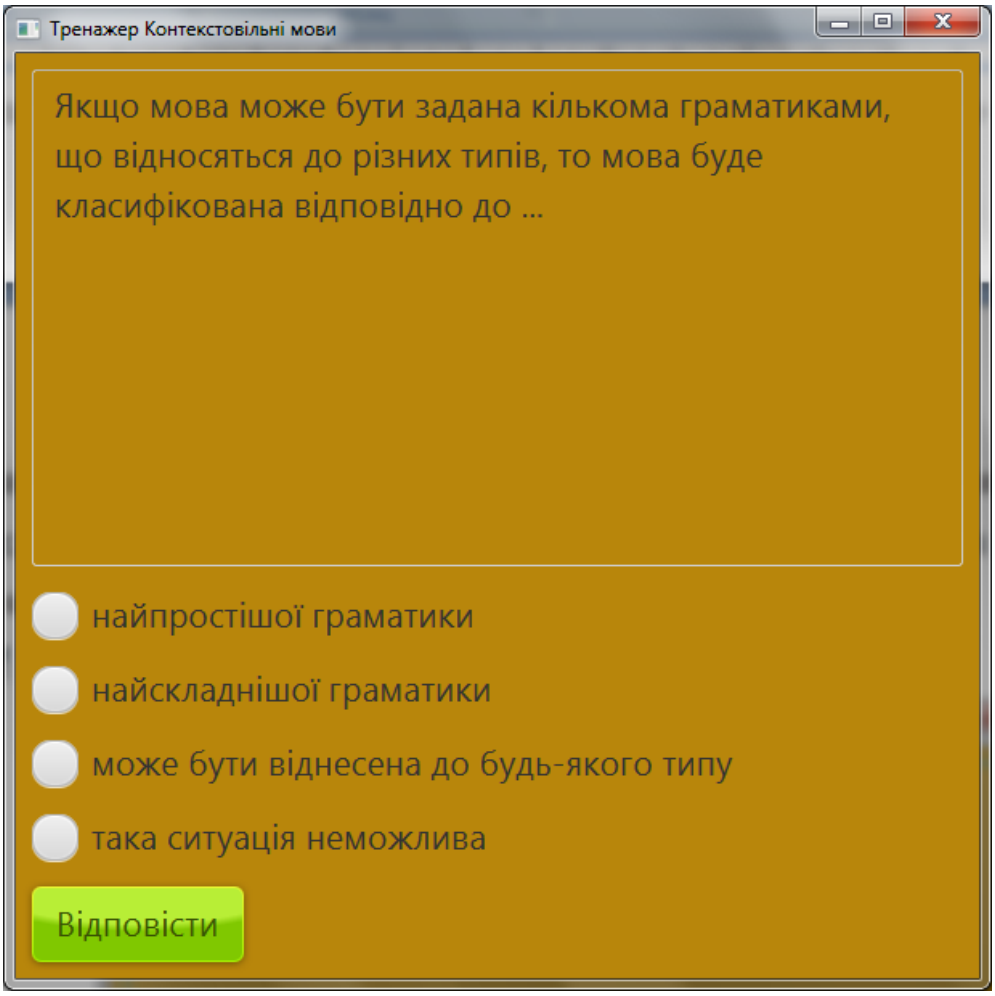

Рис. 4.15 Питання №6

Якщо вибрати варіант відповіді "найскладнішої граматики", то просто отримаємо діалогове вікно з інформацією, що відповідь неправильна (рис. 4.18).

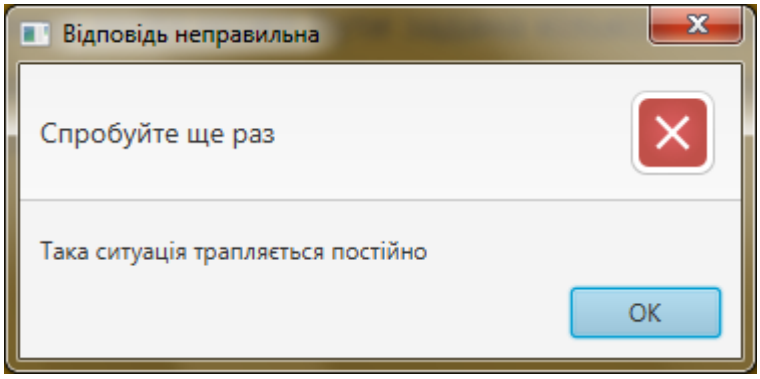

Рис. 4.16 Підказка при виборі відповіді №4 на питання №6

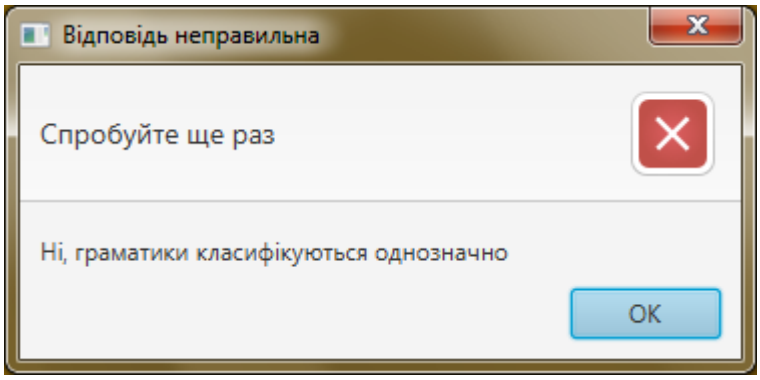

Рис. 4.17 Підказка при виборі відповіді №3 на питання №6

У випадку правильної відповіді ("найпростішої граматики") виводиться повідомлення, що відповідь правильна.

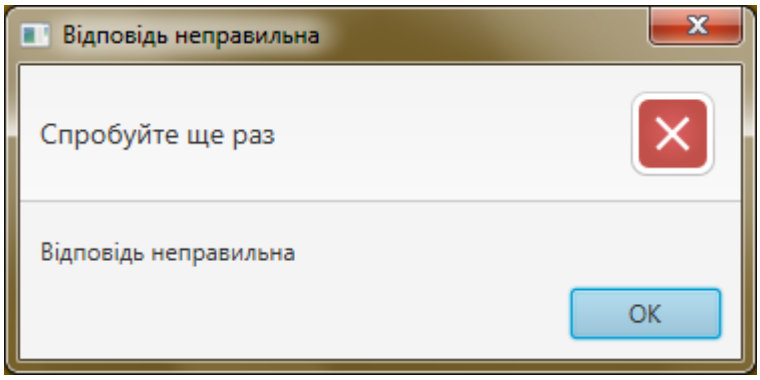

Рис. 4.18 Підказка при виборі відповіді №2 на питання №6

Наступне питання (рис. 4.19) з тієї ж групи, що й попереднє. Але питання сформульовано не в загальному вигляді, а на прикладі конкретних типів граматик. Формулювання запитання: "Якщо мова може бути задана контекстовільною і регулярною граматиками, то вона буде класифікована як ...". Користувач повинен визначити, яка із двох граматик є простішою і дати відповідь на запитання.

У випадку неправильної відповіді виводяться підказки аналогічні підказкам із попереднього кроку.

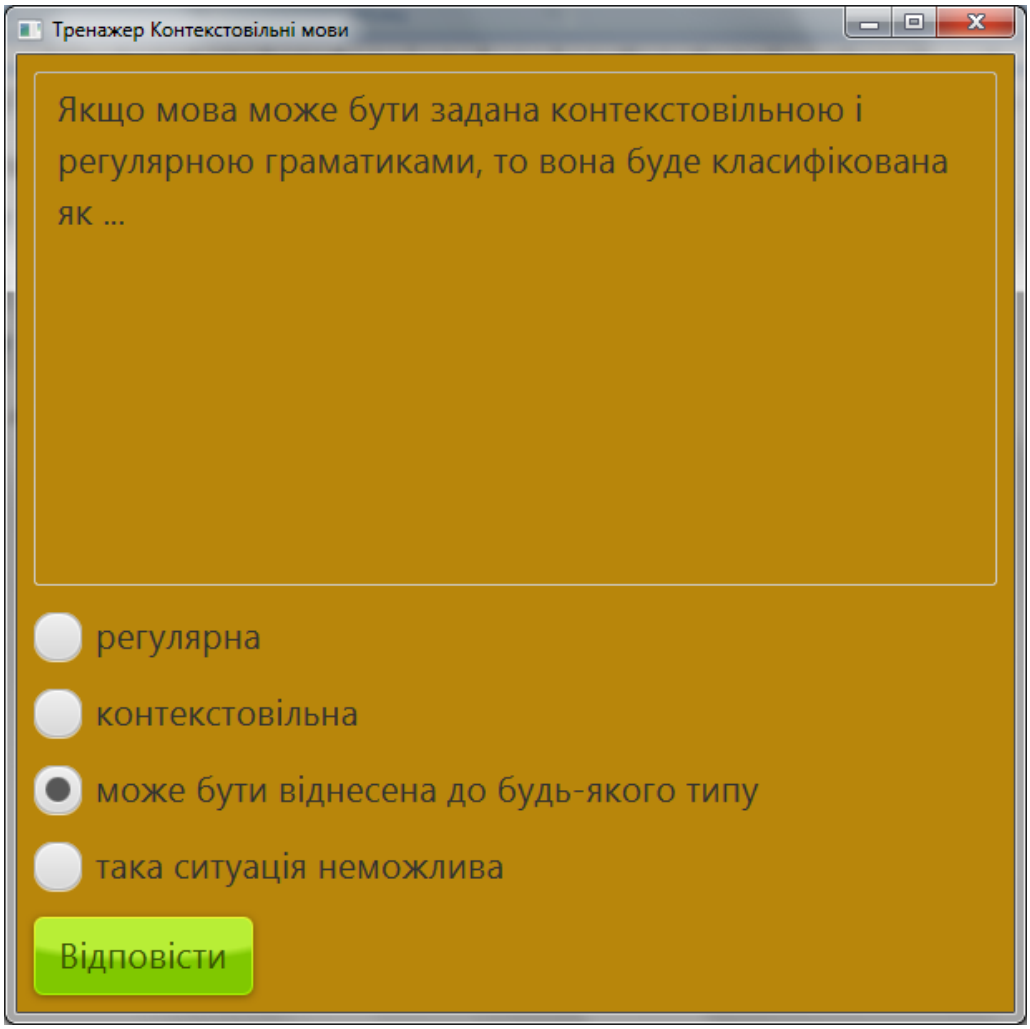

Рис. 4.19 Питання №7

На наступному кроці виводиться питання, що допускає кілька варіантів відповіді (рис. 4.20). Це питання єдине в своїй групі, тому обов'язково потрапляє у випадкову вибірку. В цьому завданні потрібно вибрати всі операції, відносно яких контекстовільні мови є замкненими. Варіанти відповіді:

1) об'єднання;

2) конкатенації;

3) замикання;

4) суперпозиції.

Контекстовільні мови є замкненими відносно усіх цих операцій, тому правильною відповіддю є вибір усіх варіантів. Якщо вибрати не всі варіанти, то для вибраних варіантів буде підтвердження, що вони правильні, а про невибрані не буде жодної інформації.

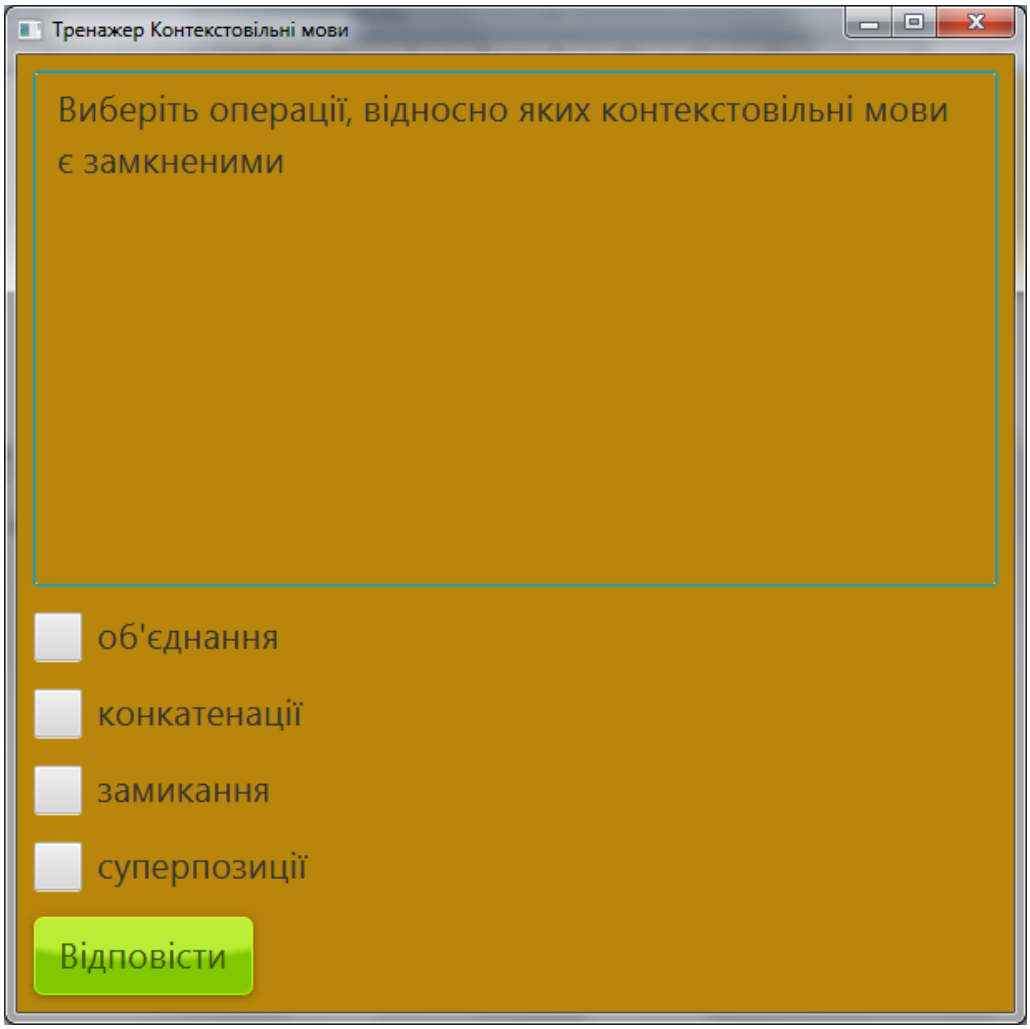

Рис. 4.20 Питання №8

Наприклад, якщо вбрати лише варіанти №2 і №3, то отримаємо таку підказку (рис. 4.21)

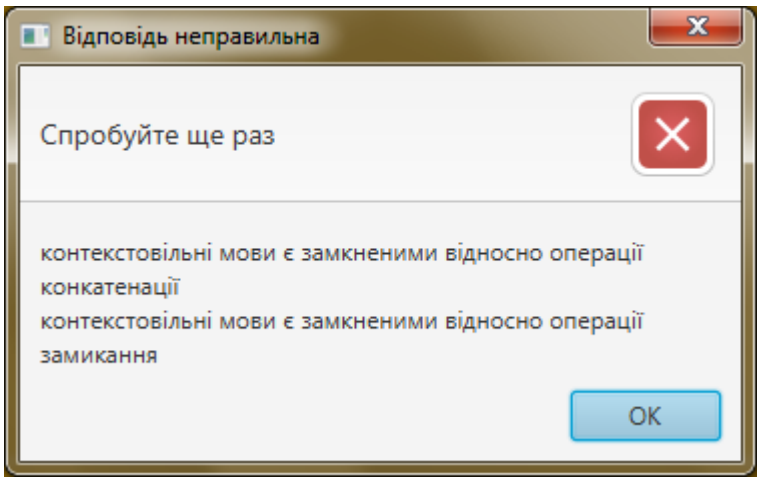

Рис. 4.21 Підказка для питання №8

Якщо вибрати всі варіанти, то отримаємо наступне повідомлення (рис. 4.22).

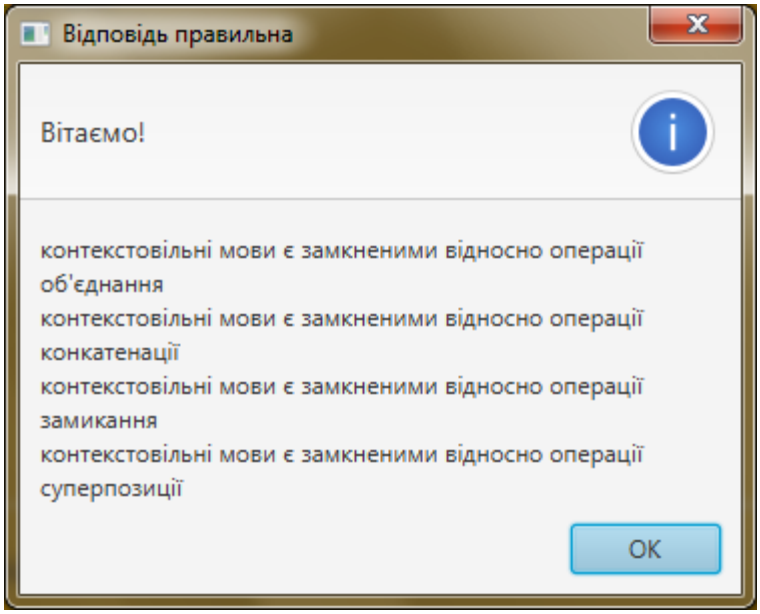

Рис. 4.22 Повідомлення про вірну відповідь

На наступному кроці потрібно вказати, за допомогою якої із граматик можна описати контекстовільну мову (рис. 4.23).

На вибір запропоновано 4 різні граматики, серед яких потрібно знайти правильну відповідь:

а) $G_1=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to xSy, S \to xy \}$.

б) $G_2=(\{x,z\}, \{S\}, P, S)$, де $P = \{ S \to xSz, S \to xz \}$.

в) $G_3=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to xSy, S \to yx \}$.

г) $G_4=(\{x,y\}, \{S\}, P, S)$, де $P = \{ S \to ySx, S \to xy \}$.

Якщо на цьому кроці вибрано невірну відповідь, то виводиться повідомлення, що відповідь помилкова і потрібно повторити спробу (рис. 4.24).

У випадку правильної відповіді виводиться привітальне повідомлення. Після закриття цього повідомлення відкривається нове запитання (рис. 4.25): "Мова $\{ x^n y^n \mid n>0\}$ ...". Запропоновані такі варіанти відповіді:

а) регулярна;

б) контекстовільна;

в) контекстозалежна.

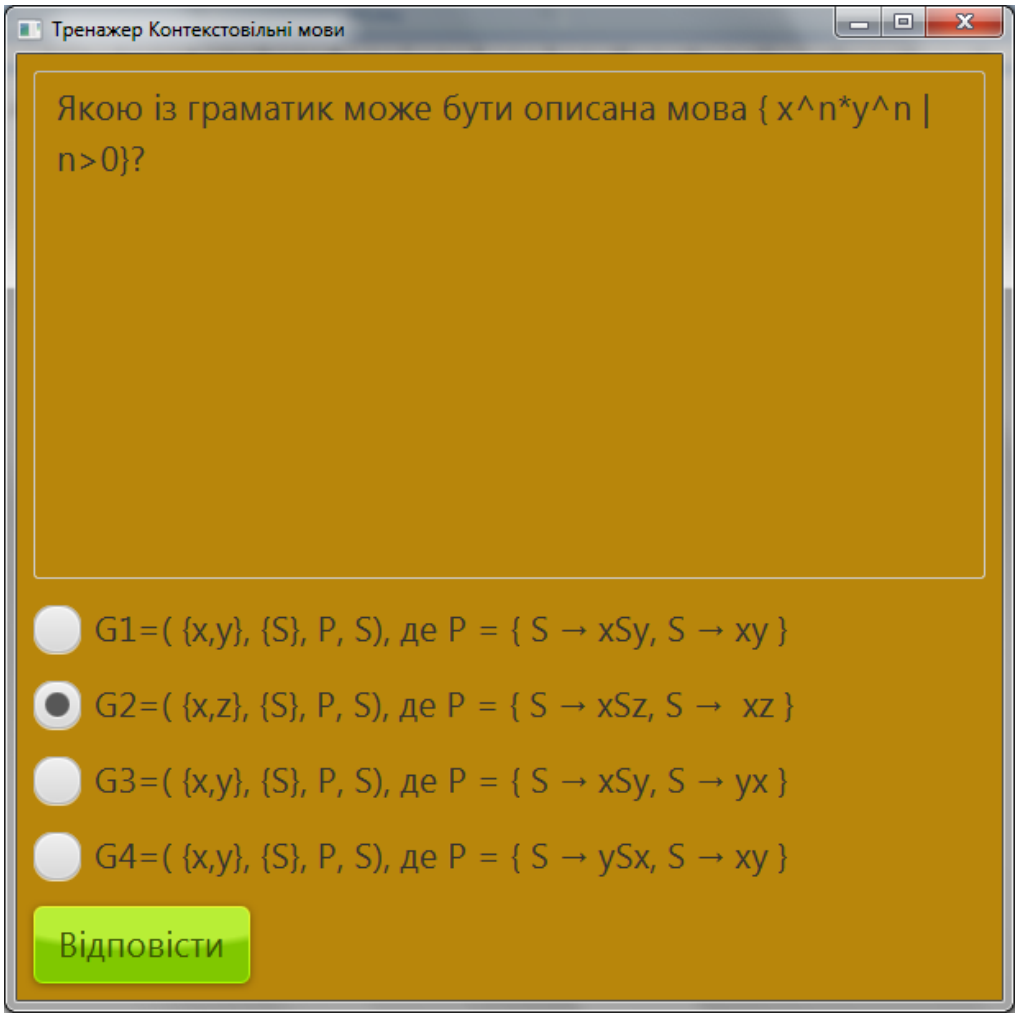

Рис. 4.23 Питання №9

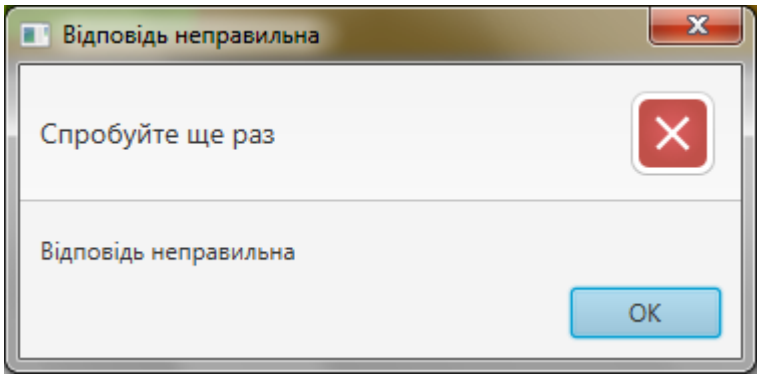

Рис. 4.24 Повідомлення про неправильну відповідь

Для цього питання у випадку неправильної відповіді виводиться повідомлення про помилку (рис. 4.24).

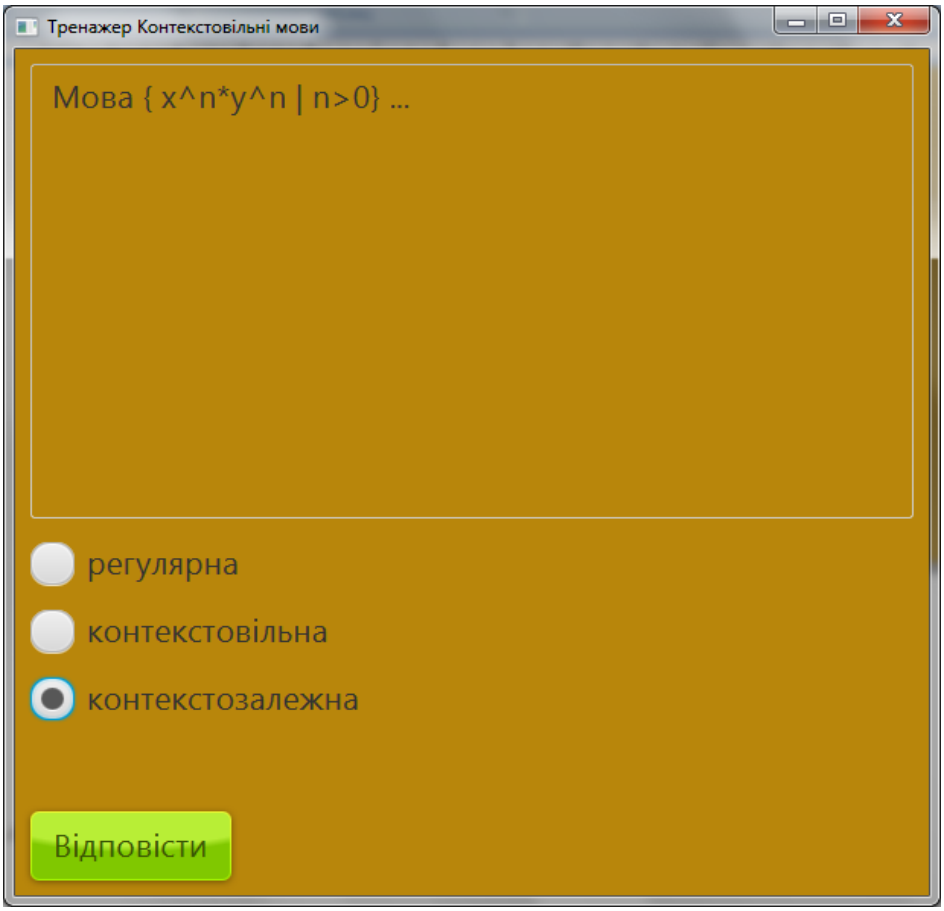

Рис. 4.25 Питання №10

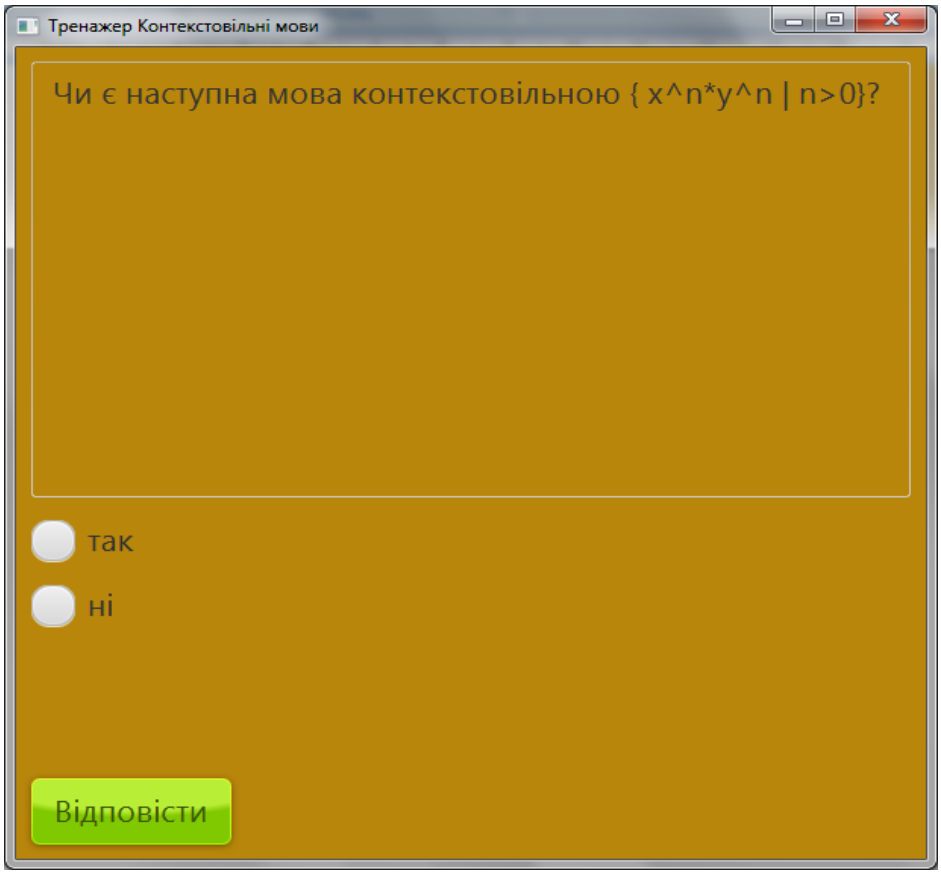

Рис. 4.26 Питання №11

В останньому питанні потрібно відповісти, чи є мова контекстовільною. Для даного питання лише 2 варіанти відповіді: так і ні.

Тобто всього користувач повинен відповісти мінімум на 11 запитань. Відмітимо, що нумерація кроків тут наведена в порядку виведення запитань при тестуванні. При іншому запуску із бази будуть вибрані випадковим чином інші питання.

Якщо під час проходження тренажера були допущені помилки, то з'являються додаткові питання. Додаткові питання беруться із тих груп, в яких були допущені помилки (табл. 3.2). Якщо в групі більше немає запитань, то додаткові питання не виводяться. Також додаткові питання не виводяться, якщо користувач не допустив жодної помилки.

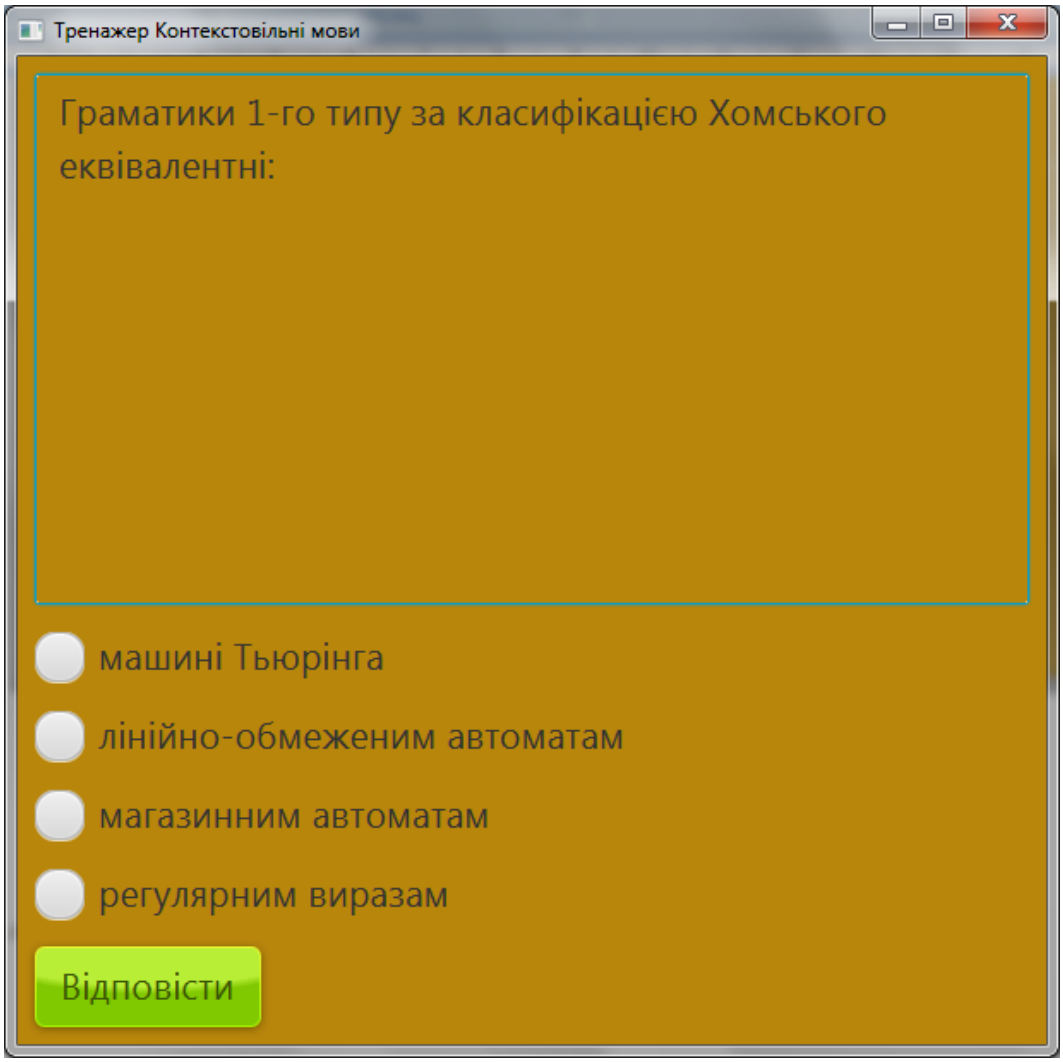

Рис. 4.27 Додаткове питання

Приклад додаткового питання показаний на рис 4.27.

Користувач жодним чином не інформується про те, що це питання є додатковим. Воно виглядає як звичайний крок тренажера.

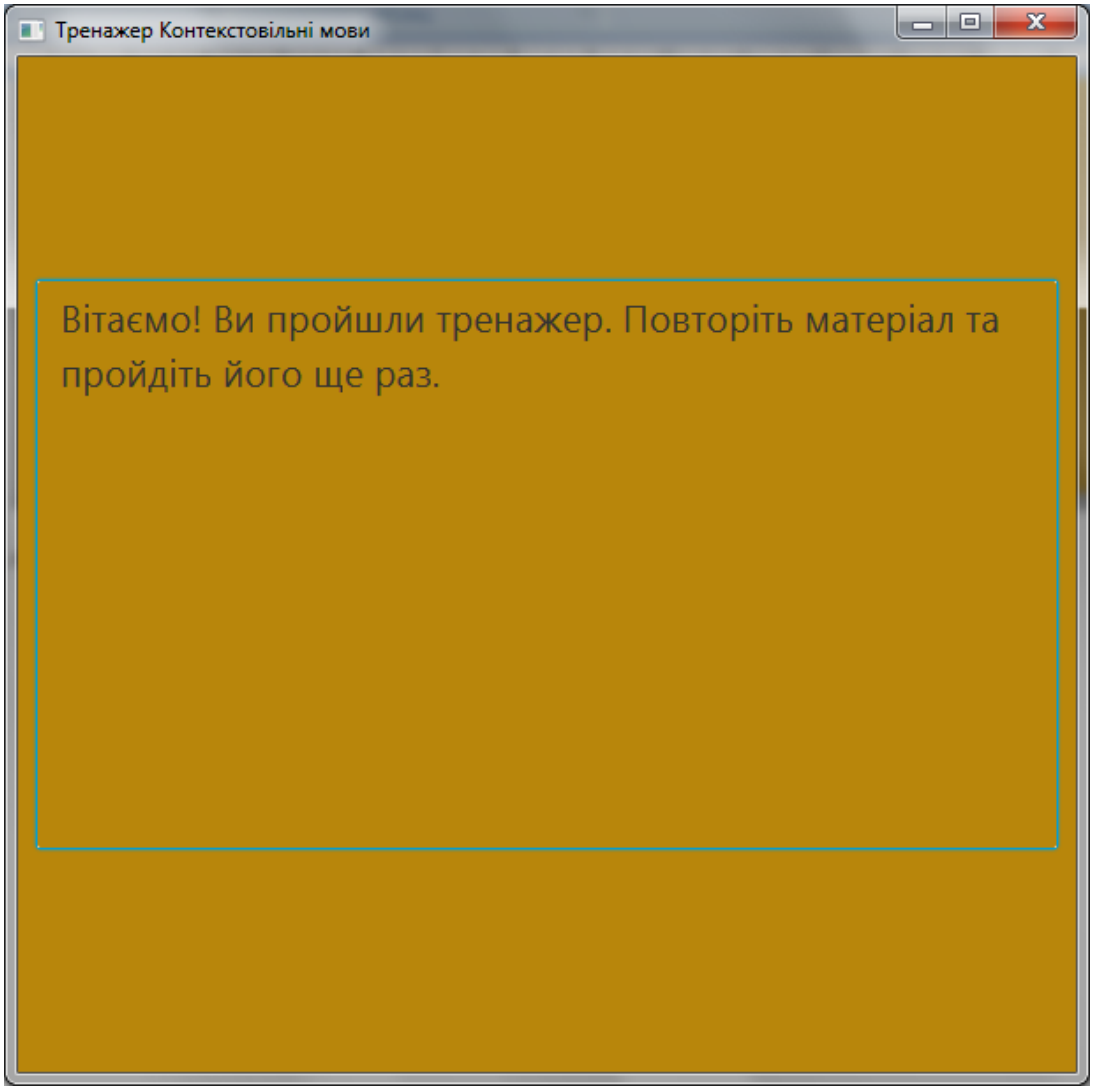

Рис. 4.28 Завершальне повідомлення

Після проходження всіх кроків з'являється фінальне вікно. Приклад такого вікна показаний на рис. 4.28. В залежності від кількості допущених помилок це вікно може містити різний текст:

- "Вітаємо! Ви успішно пройшли тренажер", якщо правильних відповідей більше 90%.

- "Вітаємо! Ви успішно пройшли тренажер. Матеріал забувається, тому згодом пройдіть його ще раз", якщо правильних відповідей від 75% до 90%.
- "Вітаємо! Ви пройшли тренажер. Матеріал забувається, тому згодом пройдіть його ще раз", якщо правильних відповідей від 60% до 75%.
- "Вітаємо! Ви пройшли тренажер. Повторіть матеріал та пройдіть його ще раз", якщо правильних відповідей менше 60%.

ВИСНОВКИ

В магістерській роботі розглядається розробка тренажера з теми «Контекстовільні мови» дистанційного навчального курсу «Теорія програмування».

Основними результатами роботи є розробка алгоритму роботи тренажера та його блок-схеми, реалізація та тестування тренажера.

Робота складається зі вступу, чотирьох розділів, висновків, списку літератури та одного додатку, в якому наведений сирцевий код тренажера.

В першому розділі розглядається загальна постановка задачі.

В другому розділі зроблено огляд створених навчальних тренажерів з дисципліни «Теорія програмування».

В третьому розділі описаний алгоритм роботи тренажера.

В четвертому розділі розглядається розробка тренажера та його тестування.

Для створення тренажера використовувалася об'єктно-орієнтована мова програмування Java. Тренажер реалізований за допомогою інтегрованого середовища розробки IntelliJ IDEA.

Тренажер може бути доданий до дистанційного курсу з дисципліни «Теорія програмування».

Результати роботи можуть використовуватися при вивченні студентами денної, заочної та дистанційної форм навчання при вивчені теми «Контекстовільні мови» дисципліни «Теорія програмування».

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Черненко О. О. Методичні підходи щодо створення дистанційного курсу з дисципліни "Теорія програмування"/ О. О.Черненко// Інформатика та системні науки (ІСН-2017): матеріали VIII Всеукраїнської науково-практичної конференції за міжнародною участю (м. Полтава, 16–18 березня 2017 р.) – Полтава: ПУЕТ, 2017. – С. 285-286. (http://dspace.puet.edu.ua/handle/123456789/5591)
2. Теркун Д.С. Удосконалення електронного навчального посібника з дисципліни "Теорія програмування" / Д.С. Теркун // Інформатика та системні науки (ІСН-2012) : матеріали III Всеукр-.наук.-практ. конф., (м. Полтава, 1–3 березня 2012 р.). – Полтава: ПУЕТ, 2012. – С. 250-251.
3. Данник О. І. Навчальний тренажер з теми «Мови і граматики» та його програмна реалізація / О. І. Данник, О. О. Черненко // Комп'ютерні науки і прикладна математика (КНіПМ-2018): матеріали науково-практичного семінару. Випуск 2 – Полтава: Кафедра ММСІ ПУЕТ, 2018. – С.4-5.
4. Товстоножко О. С. Пояснювальна записка до дипломної роботи на тему «Розробка програмного забезпечення тренажера з теми «Алгоритмічно нерозв'язні проблеми» дистанційного навчального курсу «Теорія програмування» / О. С. Товстоножко – Полтава: ПУЕТ, 2020. – 47 с. (http://dspace.puet.edu.ua/handle/123456789/10047)
5. Величко А. О. Пояснювальна записка до бакалаврської роботи на тему «Програмне забезпечення для тренажера з теми «Способи задання мов» дистанційного навчального курсу «Теорія програмування» / А. О. Величко – Полтава: ПУЕТ, 2020. – 57 с. (http://dspace.puet.edu.ua/handle/123456789/9031)
6. Цехмейстер В. О. Пояснювальна записка до бакалаврської роботи на тему «Тренажер з теми «Метод обрізання основ» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення» / В. О. Цехмейстер – Полтава: ПУЕТ, 2020. – 35 с. (http://dspace.puet.edu.ua/handle/123456789/9028)

7. Скромінський М. В. Пояснювальна записка до бакалаврської роботи на тему «Розробка програмного забезпечення для тренажера з теми «Контекстовільні граматики» дистанційного навчального курсу «Теорія програмування» / М. В. Цехмейстер – Полтава: ПУЕТ, 2020. – 39 с. (http://dspace.puet.edu.ua/handle/123456789/9025)
8. 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» / І. І. Костромін – Полтава: ПУЕТ, 2020. – 49 с. (http://dspace.puet.edu.ua/handle/123456789/9005)
9. Князєв О. М. Пояснювальна записка до бакалаврської роботи на тему «Алгоритмізація та програмування тренажера «Видалення лівої рекурсії» дистанційного навчального курсу «Теорія програмування» / О. М. Князєв – Полтава: ПУЕТ, 2020. – 47 с. (http://dspace.puet.edu.ua/handle/123456789/9003)
10. Алексов С. В. Пояснювальна записка до бакалаврської роботи на тему «Алгоритмізація та програмування тренажера «Дерева розбору» дистанційного навчального курсу «Теорія програмування» / С. В. Алексов – Полтава: ПУЕТ, 2020. – 84 с. (http://dspace.puet.edu.ua/handle/123456789/8998)
11. Алексов С. В. Тренажер з теми «Дерево розбору» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення / С. В. Алексов, О. О. Черненко // Актуальні питання розвитку науки та забезпечення якості освіти у XXI столітті: тези доповідей XLIII Міжнародної наукової студентської конференції за підсумками науково-дослідних робіт студентів за 2019 рік (м. Полтава, 07–08 квітня 2020 року). Частина 2 – Полтава: ПУЕТ, 2020 – С.115-117.
12. Белінський О. Б. Пояснювальна записка до бакалаврської роботи на тему «Тренажер з теми «Побудова предикативних синтаксичних аналізаторів» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення» / О. Б. Белінський – Полтава: ПУЕТ, 2020. – 48 с. (http://dspace.puet.edu.ua/handle/123456789/8999)

13. Волосевич М. В. Тренажер з теми «Марківські підстановки» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення / М. В. Волосевич, О. О. Черненко // Актуальні питання розвитку науки та забезпечення якості освіти у XXI столітті: тези доповідей XLIII Міжнародної наукової студентської конференції за підсумками науково-дослідних робіт студентів за 2019 рік (м. Полтава, 07–08 квітня 2020 року). Частина 2 – Полтава: ПУЕТ, 2020 – С.119-121.
14. Ємець О.О. Про розробку тренажерів для дистанційних курсів кафедрою ММСІ ПУЕТ / О.О. Ємець // Інформатика та системні науки (ІСН-2015): матеріали VI Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 19–21 берез. 2015 р.). – Полтава: ПУЕТ, 2015. – С. 152-161.
15. Ольховська О. В. Технології підтримки системи дистанційного навчання в Полтавському університеті економіки і торгівлі / О. В. Ольховська, Д. М. Ольховський // Інформатика та системні науки (ІСН-2016): матеріали VII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 10–12 берез. 2016 р.). – Полтава: ПУЕТ, 2016. – С. 219-221. (http://dspace.puet.edu.ua/handle/123456789/3074)
16. Чілікіна Т. В. Огляд тренажерів з дисципліни "Математичний аналіз" на прикладі розробок студентів напряму "Інформатика" / Т. В. Чілікіна // Інформатика та системні науки (ІСН-2016): матеріали VII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 10–12 берез. 2016 р.). – Полтава: ПУЕТ, 2016. – С. 329-330. (http://dspace.puet.edu.ua/handle/123456789/2993)
17. Парфьонова Т. О. Про розробку тренажерів для дистанційного навчального курсу "Алгебра і геометрія" / Т. О. Парфьонова // Інформатика та системні науки (ІСН-2016): матеріали VII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 10–12 берез. 2016 р.) – Полтава: ПУЕТ, 2016.
18. Кондрашев Д. М. Розробка тренажера дистанційного навчального курсу "Математична логіка та теорія алгоритмів" з теми "Машини Тюрінга" /

Д. М. Кондрашев // Інформатика та системні науки (ІСН-2017): матеріали VIII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 16-18 березня 2017 р.) – Полтава: ПУЕТ, 2017. (http://dspace.puet.edu.ua/handle/123456789/5497)

19. Сокол О. В. Розробка тренажера з теми «Нормальні алгоритми» дистанційного навчального курсу «Теорія алгоритмів» / О. В. Сокол, О. О. Черненко //Інформатика та системні науки (ІСН-2017): матеріали VIII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 16-18 березня 2017 р.) – Полтава: ПУЕТ, 2017. (http://dspace.puet.edu.ua/handle/123456789/5481)
20. Потерайло О. О. Розробка програмного забезпечення тренажера з теми «Застосування апарата алгебри логіки до логіко-математичної практики» дистанційного навчального курсу «Математична логіка та теорія алгоритмів» / О. О. Потерайло // Інформатика та системні науки (ІСН-2015): матеріали VI Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 19–21 берез. 2015 р.) – Полтава: ПУЕТ, 2015. (http://dspace.puet.edu.ua/handle/123456789/2503)
21. Русін В. С. Програмна реалізація елементів тренажера з теми "Аналіз алгоритму сортування вставками" дисципліни "Аналіз алгоритмів" / В. С. Русін, Ю. Ф. Олексійчук // Інформатика та системні науки (ІСН-2017): матеріали VIII Всеукраїнської науково-практичної конференції за міжнародною участю (м. Полтава, 16–18 березня 2017 р.) – Полтава: ПУЕТ, 2017. – С. 236-237. (http://dspace.puet.edu.ua/handle/123456789/5654)
22. Собко Д. О. Розробка програмної платформи для створення веб-тренажерів з математичних дисциплін / Д. О. Собко // Інформатика та системні науки (ІСН-2016): матеріали VII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 10–12 берез. 2016 р.) – Полтава: ПУЕТ, 2016. (http://dspace.puet.edu.ua/handle/123456789/2935)
23. Мандрика В. М. Розробка тренажеру з теми «1-R алгоритм» дисципліни «Комп’ютерний аналіз статистичних даних» / В. М. Мандрика,

Ю. Ф. Олексійчук // Комп'ютерні науки і прикладна математика (КНіПМ-2018): матеріали науково-практичного семінару. Випуск 2 – Полтава: Кафедра ММСІПУЕТ, 2018. – С. 10-13.

24. Кильник В. В. Програмна реалізація елементів тренажеру з теми «Навчання елементарного персептрону» дисципліни «Нейронно-мережеві технології в інформатиці» / В. В. Кильник, Ю. Ф. Олексійчук // Комп'ютерні науки і прикладна математика (КНіПМ-2018): матеріали науково-практичного семінару. Випуск 1 – Полтава: Кафедра ММСІ ПУЕТ, 2018. – С. 54-58. (http://dspace.puet.edu.ua/handle/123456789/6497)
25. Ярмоленко А. В. Алгоритм роботи тренажеру з теми «Асимптотичні оцінки функцій» дисципліни «Аналіз алгоритмів» / А. В. Ярмоленко, Ю. Ф. Олексійчук // Комп'ютерні науки і прикладна математика (КНіПМ-2018): матеріали науково-практичного семінару. Випуск 2 – Полтава: Кафедра ММСІ ПУЕТ, 2018. – С.14-16. (http://dspace.puet.edu.ua/handle/123456789/6974)
26. Марченко Д. А. Алгоритмізація та програмна реалізація тренажера з теми «Дослідження на збіжність числових рядів» дистанційного навчального курсу «Математичний аналіз» / Д. А. Марченко // Інформатика та системні науки (ІСН-2015): матеріали VI Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 19–21 берез. 2015 р.) – Полтава: ПУЕТ,2015.
27. Бабій М.С. Теорія програмування: Навчальний посібник / М. С. Бабій, О. П. Чекалов – Суми: Вид-во СумДУ, 2009.- 181с.
28. Moodle в Україні: Що таке Moodle – Доступний з<https://moodle.org/mod/page/view.php?id=8174>
29. Эккель Б. Философия Java. Библиотека программиста. 4-е изд./Б. Эккель — СПб: Питер, 2009. — 640 с.
30. Машнин Т. С. JavaFX 2.0: разработка RIA-приложений / Т. С. Машнин — СПб.: БХВ-Петербург, 2012. — 320 с.
31. Шилдт Г. Полный справочник по Java, 10-е издание. — К.: Издательский дом "Вильямс", 2017. — 1488 с.

32. Лєбєдєва М. О. Алгоритм тренажера з теми «Контекстовільні мови» дистанційного навчального курсу «Теорія програмування» / М. О. Лєбєдєва // Збірник наукових статей магістрів. Навчально-науковий інститут бізнесу та сучасних технологій. – Полтава: ПУЕТ, 2020. – С. 158-162.

ДОДАТОК А. КОД РОЗРОБЛЕНОГО ТРЕНАЖЕРА