ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ
Навчально-науковий інститут денної освіти
Форма навчання денна
Кафедра комп'ютерних наук та інформаційних технологій

Допускається до захисту
Завідувач кафедри
_____ Олена ОЛЬХОВСЬКА
(підпис)

«___» _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на тему
ТРЕНАЖЕР З ТЕМИ «ФОРМАЛЬНІ МОВИ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ ПРОГРАМУВАННЯ» ТА РОЗРОБКА ЙОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

зі спеціальності 122 Комп'ютерні науки
освітня програма «Комп'ютерні науки»
ступеня бакалавра

Виконавець роботи Козир Володимир Анатолійович
_____ «____»_____ 2023 р.
(підпис)

Науковий керівник к.ф.-м.н. доцент, Черненко Оксана Олексіївна
_____ «____»_____ 2023 р.
(підпис)

Рецензент

ПОЛТАВА 2023 р.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, скорочень, символів
Imports	Використовується для введення простору імен (namespace) або класів, щоб мати можливість використовувати їх у програмі без повторного введення повного шляху до кожного об'єкта.
Module	Використовується для оголошення модуля, який є контейнером для збереження процедур, функцій та змінних. Він дозволяє групувати пов'язаний код разом для кращої організації та структури програми.
Show	Використовується для відображення вікна або форми на екрані. Він використовується зазвичай для показу або активації форми користувачу.
Hide	Він забезпечує можливість зробити форму невидимою для користувача, але продовжити виконувати програму.

ВСТУП

У світі, де технології стають все більш впровадженими та поширеними, дистанційне навчання здобуває все більшу вагу та важливість. Дистанційне навчання, також відоме як онлайн-навчання або е-навчання, є процесом отримання знань та навичок за допомогою електронних засобів комунікації, таких як комп'ютери, Інтернет та спеціалізовані платформи.

Важливість дистанційного навчання виникає з численних переваг, які воно пропонує. По-перше, воно забезпечує гнучкість у навчанні, дозволяючи студентам вибирати свій власний темп, місце та час для вивчення матеріалів. Це особливо корисно для студентів, які мають робочі або сімейні зобов'язання, а також для тих, хто проживає в віддалених регіонах або не має доступу до традиційних освітніх установ.

Дистанційне навчання також сприяє глобалізації освіти, дозволяючи студентам з різних країн спілкуватися та співпрацювати, обмінюватися досвідом та культурними перспективами. Це розширює можливості навчання та сприяє розвитку міжкультурної толерантності та розуміння.[2]

Мета кваліфікаційної роботи – розробка програмного забезпечення тренажера з теми «Формальні мови» дистанційного навчального курсу «Теорія програмування».

Об’єкт розробки – алгоритм, блок-схема, програмна реалізація програмного забезпечення тренажеру.

Предмет розробки – тренажер з теми «Формальні мови» дистанційного навчального курсу «Теорія програмування» та розробка його програмного забезпечення.

Кваліфікаційна робота складається з чотирьох розділів, в кожному з яких розглядаються різні аспекти реалізації навчального тренажеру. У

першому розділі надається постановка завдання, яке стоїть перед тренажером. Другий розділ присвячений формальним мовам, де детально розглядаються їх характеристики та особливості. Третій розділ містить теоретичний матеріал, що стосується тематики тренажеру, а також докладний опис алгоритму роботи та блок-схеми тренажеру. У четвертому розділі описується процес програмної реалізації тренажеру, а також надається інструкція для користувача.

1. ПОСТАНОВКА ЗАВДАННЯ

Основним завданням кваліфікаційної роботи є створення програмного забезпечення для реалізації тренажеру з теми "Формальні мови" дистанційного навчального курсу "Теорія програмування". В рамках роботи передбачається виконання наступних завдань:

1. Вивчити матеріал з теми "Формальні мови".
2. Розробити алгоритм роботи тренажеру.
3. Створити блок-схему алгоритму роботи тренажеру.
4. Реалізувати програмно тренажер.
5. Описати результати програмно реалізованого тренажеру.

У процесі програмування програмного забезпечення тренажеру, необхідно врахувати такі вікна:

1. Стартове вікно тренажеру: це вікно міститиме інформацію про тематику тренажеру, кнопку, за допомогою якої розпочинається тренування, а також інформацію про розробника і наукового керівника курсового проекту з фаху.

2. Інші вікна тренажеру: ці вікна включатимуть питання по темі тренажеру у формі тесту, на які користувач повинен правильно або неправильно відповісти.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Класифікація формальних мов

Для віднесення граматики до того або іншого типу необхідна відповідність всіх її правил (продукцій) деяким схемам. Згідно Хомського, формальні граматики поділяються на чотири типи:

- Тип 0 — *необмежені*. Вони не мають обмежень на вид правил. У цьому випадку для граматики G = (T, N, P, S), V=T∪N всі правила мають вид:

$$\alpha \to \beta,$$

де α — будь-яка послідовність символів, що містить хоча б один нетермінал, а β — будь-яка послідовність символів алфавіту.

Практичного застосування в силу своєї складності такі граматики не мають.

- Тип 1 — *контекстно-залежні* (*КЗ*). Правила підстановок, що застосовуються до послідовності символів, залежать від їх контексту. Тобто, для граматики G = (T, N, P, S), V=T∪N всі правила мають вид:

$$\alpha A \beta \to \alpha \gamma \beta, \quad \text{де } \alpha, \beta \in V^*, \gamma \in V^+, A \in N.$$

Послідовності символів α і β є *контекстом правила*. Символ A замінюється рядком γ, якщо знаходиться в контексті α, β.

- Тип 2 — *контекстно-вільні* (*КВ*). Правила підстановок застосовуються до послідовності символів (до окремих символів) незалежно від тих символів, які їх оточують (від контексту), і при виконанні підстановок цей контекст залишається без змін. У цьому випадку для граматики G = (T, N, P, S), V=T∪N всі правила мають вид:

$$A \to \beta, \quad \text{де } A \in N, \beta \in V^+.$$

Така граматика допускає появу в лівій частині правила тільки нетерминального символа. При цьому β - непусте слово словника граматики.

- Тип 3 — *регулярні граматики* (автоматні). Є контекстно-вільними, але з обмеженими можливостями. Це граматики, в яких правила мають вид

  $A \to \gamma$ або $A \to \gamma B$, де $A, B \in N, \gamma \in V^+$.

  Самі прості із формальних граматик. Застосовуються для опису найпростіших конструкцій: ідентифікаторів, констант, тощо.

Дані типи граматик виходять від базової, необмеженої граматики (типу 0), на правила продукції якої послідовно накладаються обмеження.

В залежності від типу граматики, на основі якої генерується задана мова, формальні мови поділяють на відповідні категорії: від типу 0 до типу 3. Мови тим більш обмежені, чим глибше знаходяться в ієрархії Хомського.

Простіші мови, з одного боку, простіше піддаються комп'ютерній обробці, а з іншого — їм бракує виразності. Наприклад, для пошуку деяких шаблонів у тексті використовують регулярні вирази, які відповідають граматикам Хомського типу 3. Їхньої виразності досить, аби шукати різноманітні фрагменти тексту, але їх недостатньо, для, наприклад, описання мов програмування. Для цього використовують граматики типу 2 (КВ-граматику).

Загалом формальні мови будуються за наступною схемою:

- Спочатку вибирається алфавіт із символів якого будуть будуватися всі вирази мови. Символами алфавіту формальної мови можуть бути букви алфавітів природних мов, дужки, спеціальні знаки, тощо.
- Потім описується синтаксис мови, тобто правила побудови осмислених виразів. Для кожної формальної мови сукупність цих правил має бути строго визначена і модифікація будь-якого з них приводить найчастіше до появи нового різновиду (діалекту) цієї мови.

- Далі з символів алфавіту, за описаними правилами, складаються слова і вирази. Осмислені вирази отримуються у формальній мові лише якщо дотримані всі визначені в ній правила продукції.

Для формальної мови характерна однозначність визначення її словника, чіткі правила побудови виразів і їх тлумачення, що забезпечує несуперечливе, точне і компактне відображення властивостей і відношень предметної області (об'єктів, що моделюються).

Важливість формальних граматик в інформатиці порівняна з важливістю арифметики в математиці, логіки у філософії і т.ін. На принципах формальної граматики проектуються і створюються нові мови програмування, пишуться програми для перевірки орфографії і стилю, програмуються "мовні" інтерфейси, створюються програми для корекції помилок і "оптимізації" виконання програм.

Подібно до того як правила граматики описують природну мову, правила формальної мови управляють роботою програміста, вказуючи, як можна сполучати слова і символи, що використовуються в мові, при побудові складних виразів і задаючи правила форматування, у тому числі вживання пропусків і знаків пунктуації.

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Формальні мови

Формáльна мóва — це множина скінчених послідовностей символів (“слів” і “речень”), спосіб побудови яких описується правилами певного виду.

Будь-яка формальна мова характеризується:

- *алфавітом* - набором знаків (символів) з яких можна складати слова і фрази даної мови;
- *синтаксисом* (*граматикою*) - правилами побудови із символів алфавіту таких мовних конструкцій, як “слова”, “речення” і “тексти”;
- *семантикою* - набором правил використання мовних конструкцій, які встановлюють відповідність між структурними одиницями тексту і правилами їх інтерпретації.

Якщо T — алфавіт мови, то як T^* позначають множину рядків (послідовностей символів), які можуть бути побудовані з символів алфавіту T. При цьому передбачається, що порожній рядок (позначається як ε) також входить у множину T^*:

$$T^* = T^+ \cup \{\varepsilon\},$$

де T^+ — множина усіх можливих рядків, складених з символів алфавіту T, окрім порожнього рядка ε.

Кожна мова в алфавіті T - це підмножина множини T^*. Наприклад, якщо алфавіт заданий як $\{a, b\}$, а мова L включає в себе всі слова над ним, то слово *ababba* належить L.

Для задання формальної мови підходить будь-яке математично коректне визначення множини слів в заданому алфавіті. Так формальна мова може бути визначена

- простим перерахуванням слів, що входять в дану мову (цей спосіб, в основному, прийнятний для визначення мов простої структури).

- за допомогою механізму породження слів (формальної граматики),
- словами, породженими регулярним виразом,
- словами, породженими БНФ-конструкцією, тощо.

Проте, якщо мати на увазі задання мови, при якому можливе алгоритмічне вирішення питання про приналежність слова мові, то потрібні більш обмежені засоби. Найбільш загальним із конструктивних способів задання формальних мов є спосіб породження їх за допомогою *формальних граматик*.

Формальна граматика (*граматика формальної мови*) — це множина правил породження слів, виразів і речень формальної мови. Ці правила називають ще *синтаксисом* мови.

Формальною граматикою G для породження формальної мови в алфавіті T є набір

$$G = \{T, N, P, S\},$$

де

- T – множина символів алфавіту (термінальних символів [1]) або термінальний словник мови (її *алфавіт*). Термін "алфавіт" щодо формальних мов майже повністю відповідає його неформальному визначенню в розмовних мовах і відрізняється лише тим, що включає всі можливі символи, в той час як, наприклад, крапка, кома, знак переносу, знак оклику і т.п. при неформальному визначенні розмовних мов в алфавіт не включаються. З точки зору мов програмування, це - імена змінних, службові слова, знаки операцій тощо.
- N – нетермінальний словник або множина нетерміналів, якими позначаються синтаксичні конструкції мови. Нетермінали - це елементи граматики, що мають власні імена і структуру. Кожен нетермін складається з одного або більшої кількості терміналів і/або нетерміналів, поєднання яких визначається правилами граматики. Стосовно розмовних мов це, наприклад, префікс, корінь, підмет,

присудок, просте речення і т. ін.; з точки ж зору мов програмування, це її поняття - оператори, списки, блоки тощо.

- *P* – множина правил граматики за допомогою яких будується нетермінальний словник (синтаксичні конструкції). Множину правил граматики ще називають *синтаксисом* мови.
- *S* - мета граматики (аксіома) - старший нетермінальний символ ($S \in N$), що визначає *клас* мовних об'єктів, для опису яких призначена граматика *G*. По аналогії з розмовною мовою це, наприклад, текст; в мовах програмування це - програма.

При формальному визначенні граматики мається на увазі, що будь-який нетермінал, у тому числі і мета граматики, є одна, нехай навіть дуже довга, послідовність символів (рядок).

Правила граматики описують відношення між нетерміналами і терміналами шляхом визначення нетермінального символа через комбінацію термінальних і нетермінальних символів. Кожне таке правило *P* має вигляд $\varphi \to \psi$, де φ, ψ — елементи із словника граматики $V = T \cup N$, причому елемент φ містить принаймні один нетермінал. Таке правило означає, що послідовність символів φ у процесі виводу можна замінити на рядок ψ.

Правило виводу $\varphi \to \psi$ може застосовуватися до послідовності символів u, тільки якщо φ є фрагментом цієї послідовності. Результатом застосування такого правила до u є рядок v, отриманий заміною будь-якого фрагмента φ в послідовності символів u на рядок ψ .Виведення нетерміналів мови починається з аксіоми S.

Якщо v — результат застосування деякого правила до рядка u, то це позначається наступним чином: $u \Rightarrow v$. Якщо $u \Rightarrow v_1 \Rightarrow v_2 \Rightarrow \ldots \Rightarrow v_n \Rightarrow v$, то скорочено це можна записати так: $u \Rightarrow\Rightarrow v$.

У формальних граматиках послідовності символів інтерпретуються як мовні об’єкти різних рівнів. На рівні граматики визначаються *поняття* (нетермінали), послідовне розкриття яких (їх виведення), в кінці кінців дає їх представлення у вигляді послідовностей термінальних символів. Поняття бувають

- осмислені – мовні конструкції, для яких визначена та або інша дія,
- допоміжні - потрібні лише для того, щоб зробити текст читабельним (самостійного сенсу вони не мають).

Мінімальні осмислені поняття відповідають *лексемам*; мінімальні допоміжні поняття подібні до знаків пунктуації.

Правила граматики (продукції, виводу) фактично визначають одні поняття через інші.

Для прикладу опишемо формальну граматику, яка породжує математичні вирази (суми):

1. *Сума* → *Цифра*
2. *Сума* → *Сума* + *Цифра*
3. *Сума* → *Сума* - *Цифра*
4. *Цифра* → 1..9

Тут кожен рядок визначає одне правило, термінальні символи мови підкреслені, а нетермінальні виділені курсивом. Аксіомою в цій мові є слово *Сума*.

Виходячи із даної аксіоми та застосовуючи наведені вище правила, можна отримати наступні вирази (рядки):

1	*Сума*	Аксіома
2	*Сума* − *Цифра*	Застосовано правило 3
3	*Сума* + *Цифра* − *Цифра*	Застосовано правило 2
4	*Цифра* + *Цифра* −	Застосовано

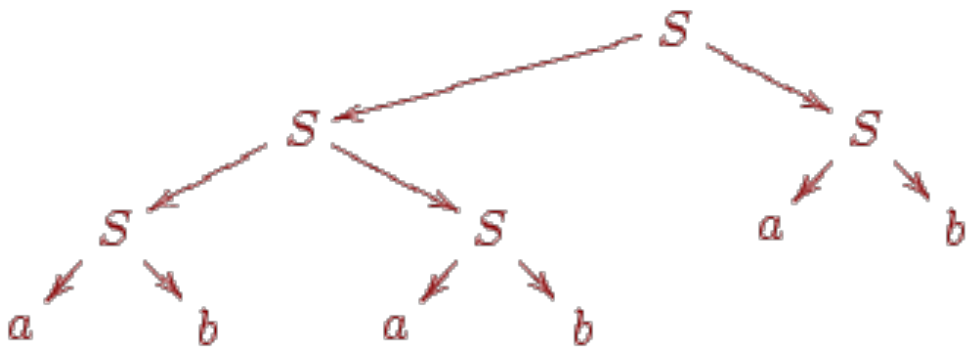
	Цифра	правило 1
5	1 + *Цифра* – *Цифра*	Застосовано правило 4
6	1 + 2 – *Цифра*	Застосовано правило 4
7	1 + 2 – 9	Застосовано правило 4

Порядок застосування правил довільний. Граматика визначає лише правила, які дозволяється використовувати у поточній ситуації, і не дає вказівок, які правила мають бути застосовані. Наприклад, до першого речення (аксіоми) можна застосувати лише правила 1—3, та не можна застосувати правило 4. Починаючи з 4 речення можливе застосування лише правила 4.

Процес виводу слова в формальній граматиці може бути графічно представлений у вигляді *синтаксичного дерева*, кореневою вершиною якого є мета граматики, в проміжних вершинах знаходяться нетермінали, а «листям» є термінали. Розглянемо наступну граматику:

$$S \to SS,$$
$$S \to ab,$$
$$S \to aSb.$$

Виведенню $S \Rightarrow SS \Rightarrow Sab \Rightarrow SSab \Rightarrow abSab \Rightarrow ababab$ відповідає наступне синтаксичне дерево:



Крона цього дерева виводу - *ababab*.

З математичної точки зору ***формальна мова*** $L(G)$ - це множина рядків символів в алфавіті *T*, які можуть бути отримані із аксіоми S шляхом кінцевого числа застосувань правил *P* граматики *G*.

Набір правил послідовно визначає всі нетермінальні символи формальної граматики так, що кожен такий символ може бути зведений до комбінації термінальних символів шляхом послідовного (рекурсивного) вживання правил.

Теорія формальних граматик сформульована у 50-ті роки американським лінгвістом Хомським (Чомскі).

3.2. Алгоритм роботи тренажеру з теми «Формальні мови» дистанційного навчального курсу «Теорія програмування»

Початок алгоритму блоку завдань.

1. Що таке формальна мова?

a) Мова, яку використовують для офіційного спілкування.

b) Мова, яку використовують в математиці та комп'ютерних науках.

c) Мова, яку використовують у літературних творах.

Правильна відповідь: b) Мова, яку використовують в математиці та комп'ютерних науках.

2. Яка характеристика формальної мови описує її синтаксис?

a) Граматика.

b) Семантика.

c) Стилістика.

Правильна відповідь: a) Граматика.

3. Що таке граматика формальної мови?

a) Правила, що визначають коректну структуру мови.

b) Значення, які приписуються символам мови.

c) Засоби вираження почуттів та емоцій у мові.

Правильна відповідь: a) Правила, що визначають коректну структуру мови.

4. Які типи формальних мов існують?

a) Регулярні, контекстні, безконтекстні та рекурсивно-перераховувані мови.

b) Українська, англійська, французька та іспанська мови.

c) Письмова, усна, технічна та наукова мови.

Правильна відповідь: a) Регулярні, контекстні, безконтекстні та рекурсивно-перераховувані мови.

5. Які операції можна виконувати з формальними мовами?

a) Об'єднання, перетин та доповнення.

b) Додавання, віднімання та множення.

c) Запити, сортування та фільтрація.

Правильна відповідь: a) Об'єднання, перетин та доповнення.

6. Що таке автомат в теорії формальних мов?

a) Математична модель, яка визначає обчислювальну поведінку формальних мов.

b) Список слів, що належать до певної формальної мови.

c) Інструкція для виконання певних операцій над символами мови.

Правильна відповідь: a) Математична модель, яка визначає обчислювальну поведінку формальних мов.

7. Що таке рекурсивно-перераховувана мова?

a) Мова, яка містить нескінченну кількість слів.

b) Мова, для якої існує алгоритм, який може перерахувати всі її слова.

c) Мова, яка не має синтаксичних правил.

Правильна відповідь: b) Мова, для якої існує алгоритм, який може перерахувати всі її слова.

8. Яка особливість контекстно-вільних граматик у теорії формальних мов?

a) Правила граматики можуть мати контекстні обмеження.

b) Граматика може визначати лише регулярні мови.

c) Граматика містить тільки нетермінали.

Правильна відповідь: a) Правила граматики можуть мати контекстні обмеження.

9. Що таке регулярний вираз у теорії формальних мов?

a) Спосіб опису мови за допомогою виразів, які містять символи та оператори.

b) Вираз, що описує мову, в якій всі слова мають однакову довжину.

c) Кодування символів мови відповідно до стандарту.

Правильна відповідь: a) Спосіб опису мови за допомогою виразів, які містять символи та оператори.

10. Що таке детермінований скінченний автомат (ДСА) в теорії формальних мов?

a) Автомат, у якого є тільки одна початкова конфігурація.

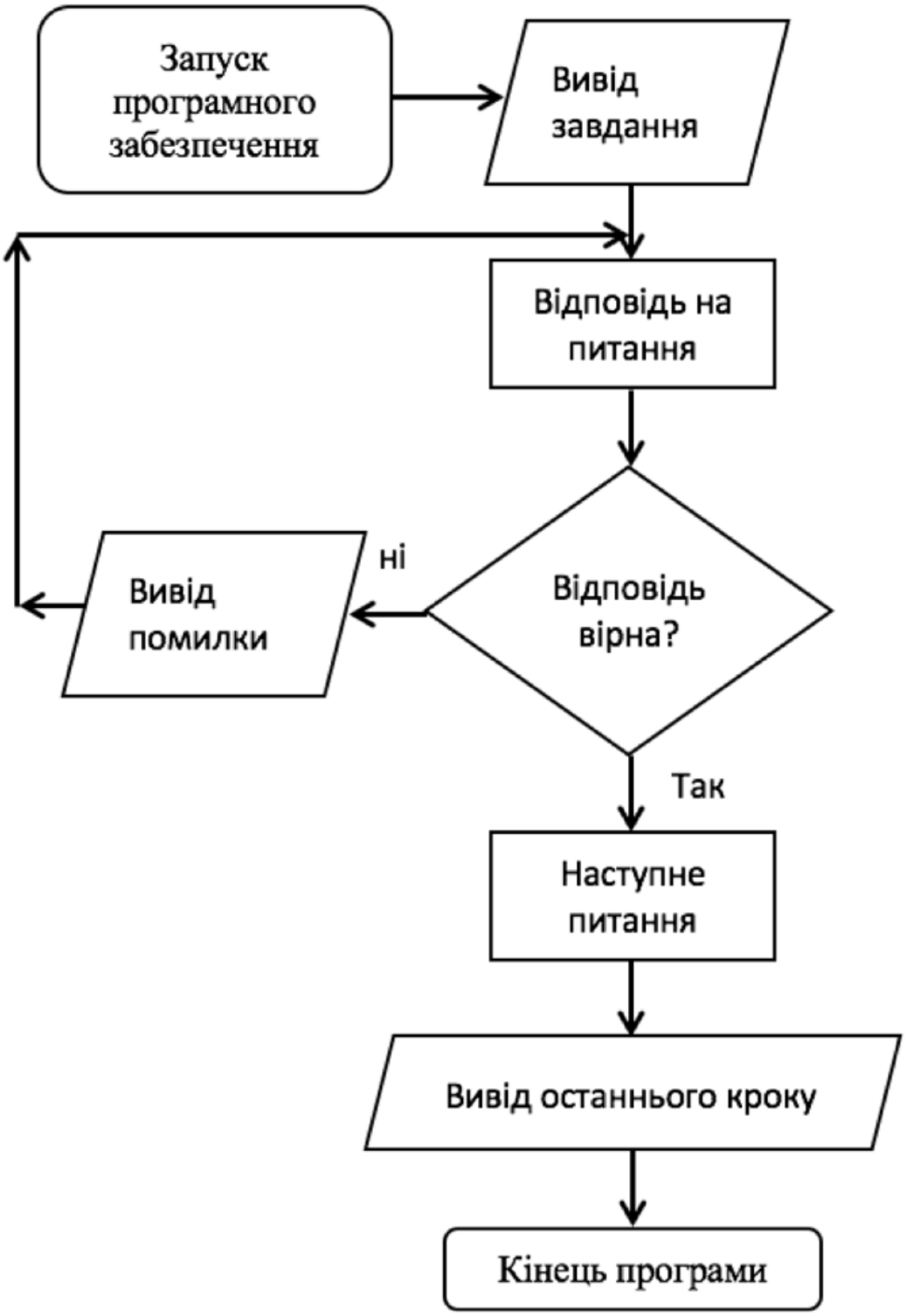
b) Автомат, у якого немає станів з затримкою.

c) Автомат, який може переходити до будь-якого стану за один крок.

Правильна відповідь: b) Автомат, у якого немає станів з затримкою.

Кінець алгоритму блоку завдань.

3.3. Блок-схема тренажеру



4. ПРАКТИЧНА ЧАСТИНА

4.1 Опис програмної реалізації

Програмні компоненти тренажеру програмно реалізовані з використанням мови програмування Visual Basic.(рис. 4.1)

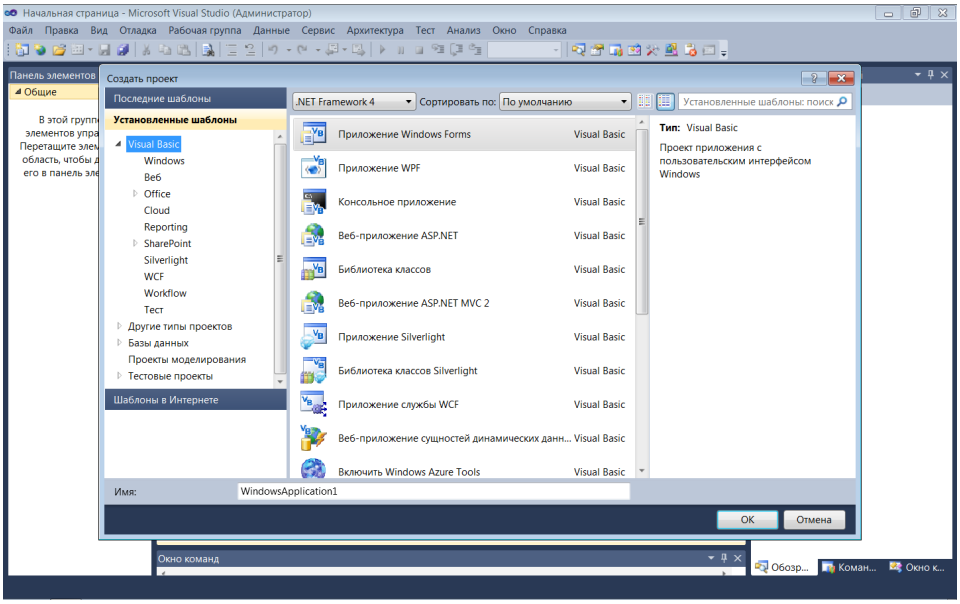


Рисунок 4.1 – програмне середовище Visual Studio

Після створення нового проекту в середовищі розробки, наступним кроком є розробка початкової форми тренажеру, яка виступає в ролі стартового екрану. (рис. 4.2)

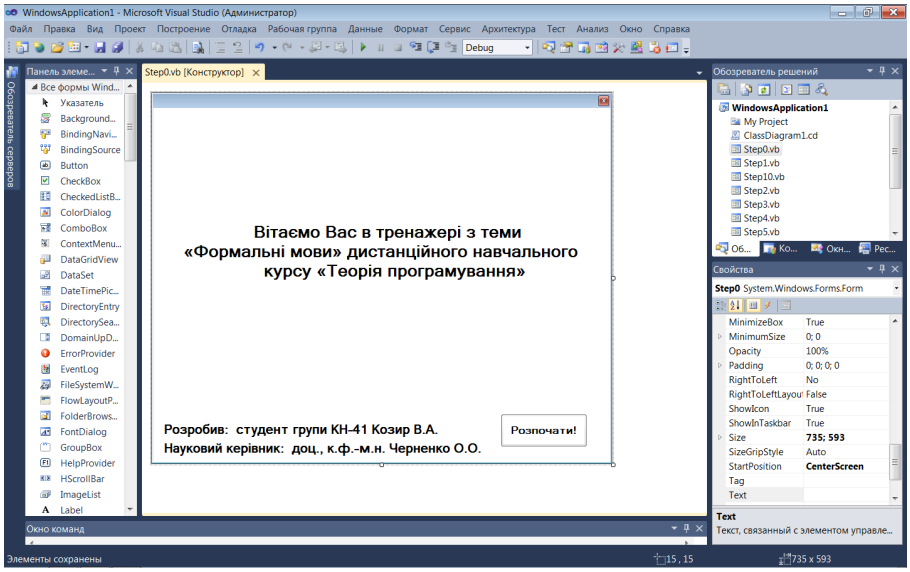


Рисунок 4.2 – стартовий екран.

Після стартового екрану, наступним кроком є програмування першого питання з лекційного матеріалу тренажеру. (рис. 4.3)

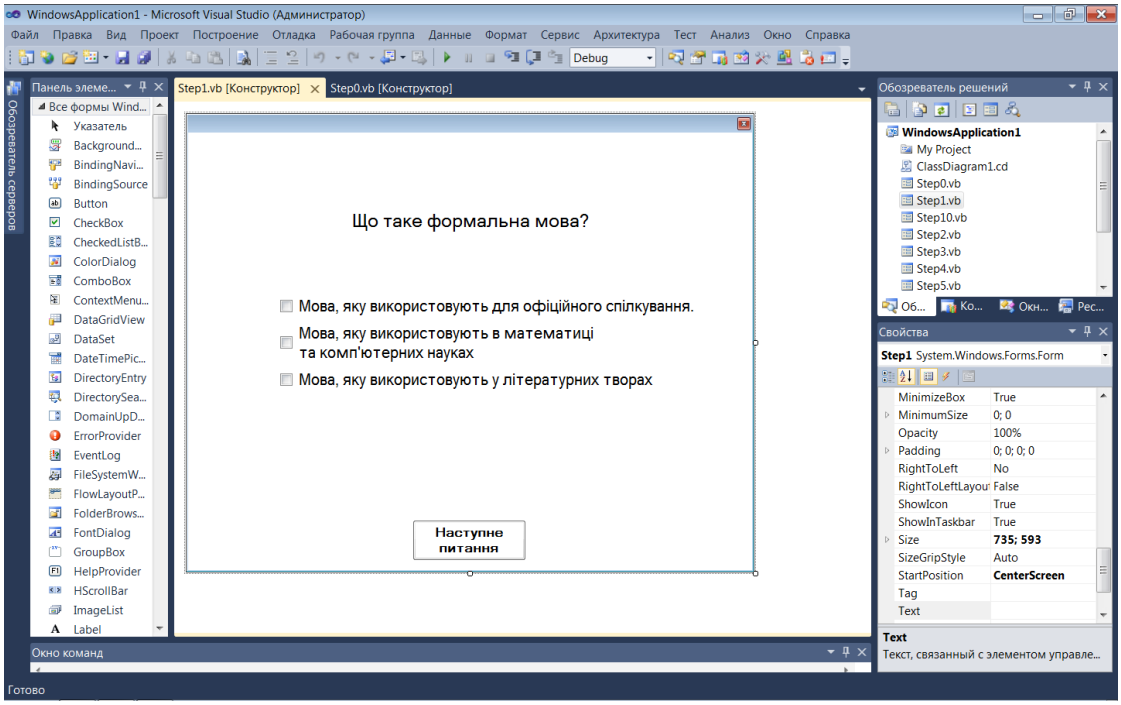


Рисунок 4.3 – перше вікно з лекційним матеріалом.

Після першого вікна з лекційним матеріалом, реалізується наступне схоже за дизайном вікно (рис. 4.4)

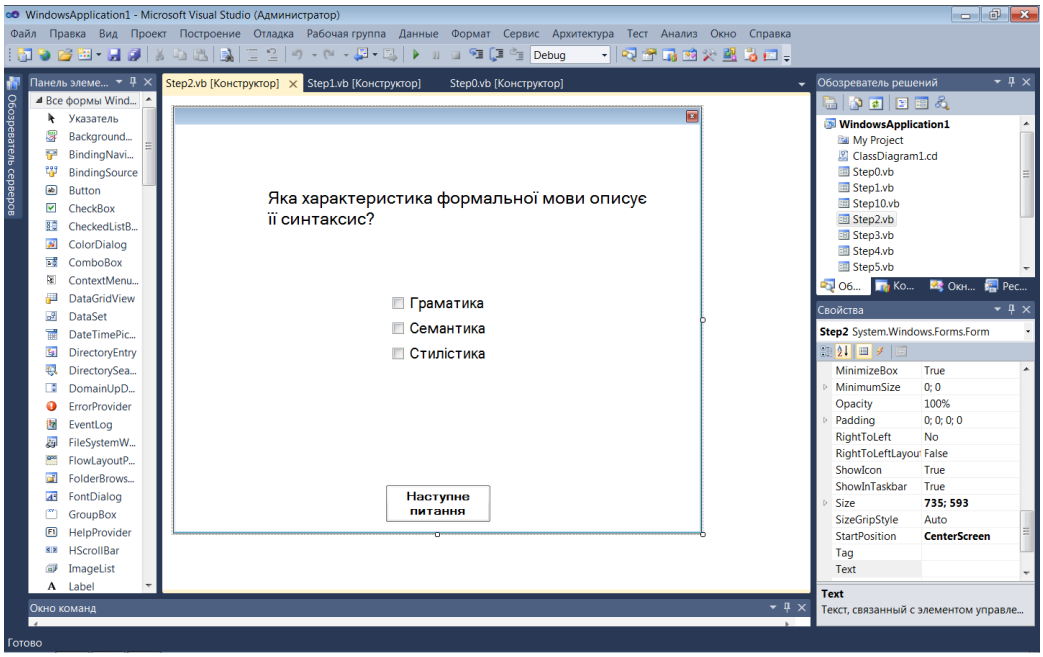


Рисунок 4.4 – друге вікно з лекційним матеріалом.

4.2. Інструкція по використанню навчального тренажеру

Після запуску програми тренажеру користувачу відкривається початковий екран (рис. 4.4)

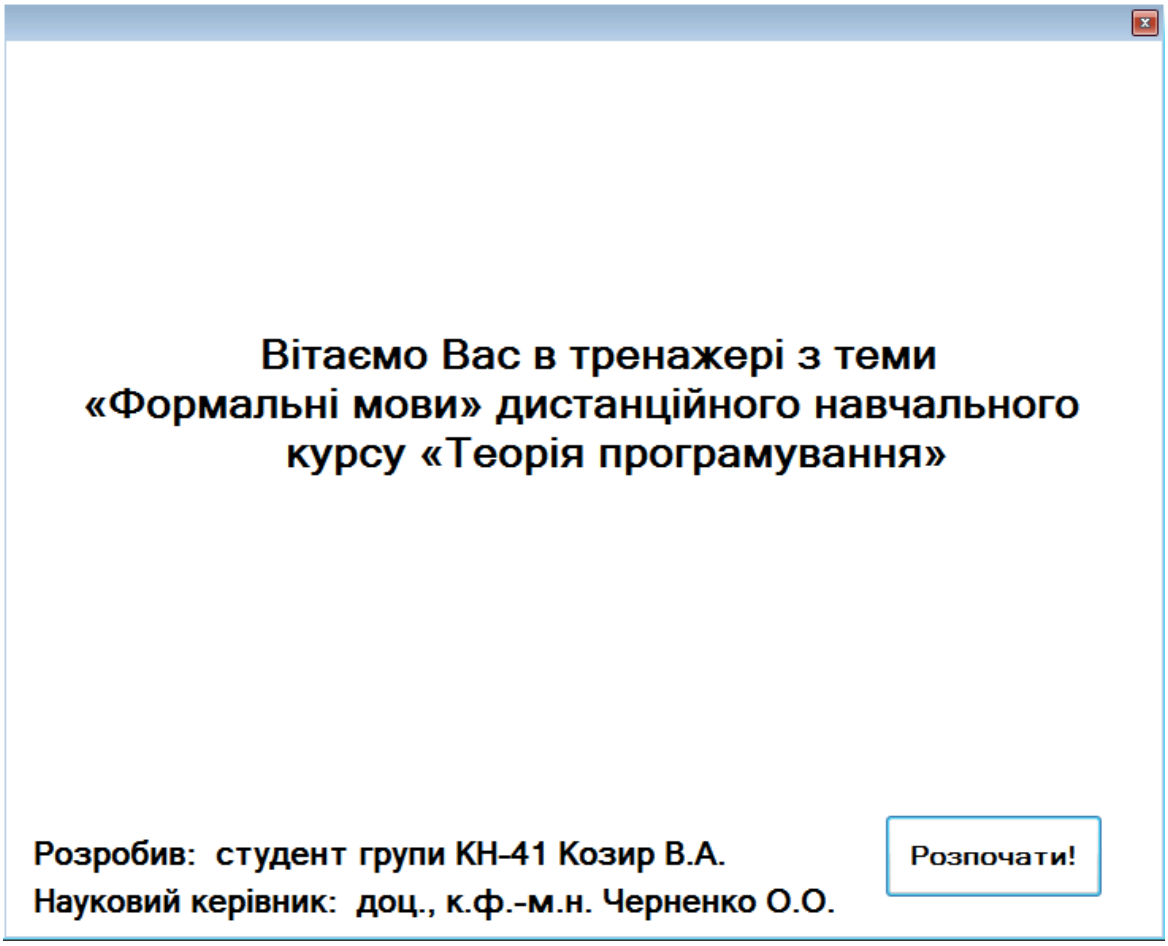


Рисунок 4.4 – головний стартовий екран тренажеру.

Далі користувач натискає кнопку на формі, користувачу автоматично виведеться перше питання, на яке він може відповісти, натиснувши будь-який варіант відповіді, що надано для цього питання. (рис. 4.5)

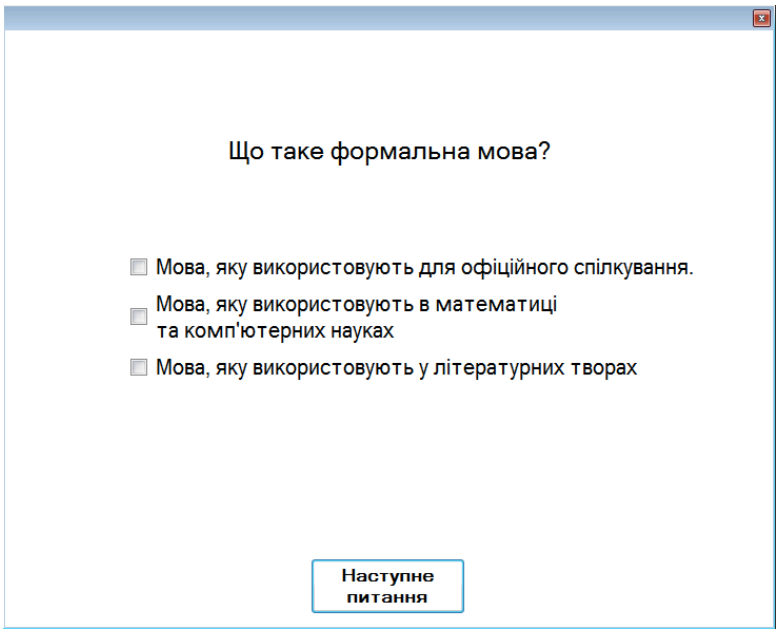


Рисунок 4.5 – перше завдання з лекційного матеріалу.

Після того як користувач обрав свій варіант відповіді, програмне забезпечення перевіряє цю відповідь, додатково користувач інформується вікном (рис. 4.6)

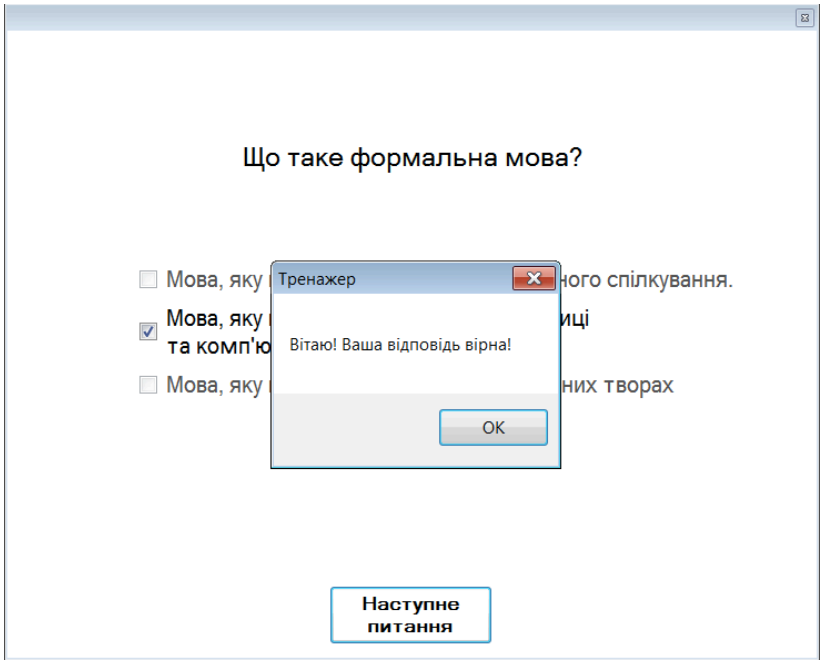


Рисунок 4.6 – сповіщення в разі правильної відповіді.

В випадку якщо користувач відповідає на питання неправильно, виводиться наступне попередження (рис. 4.7)

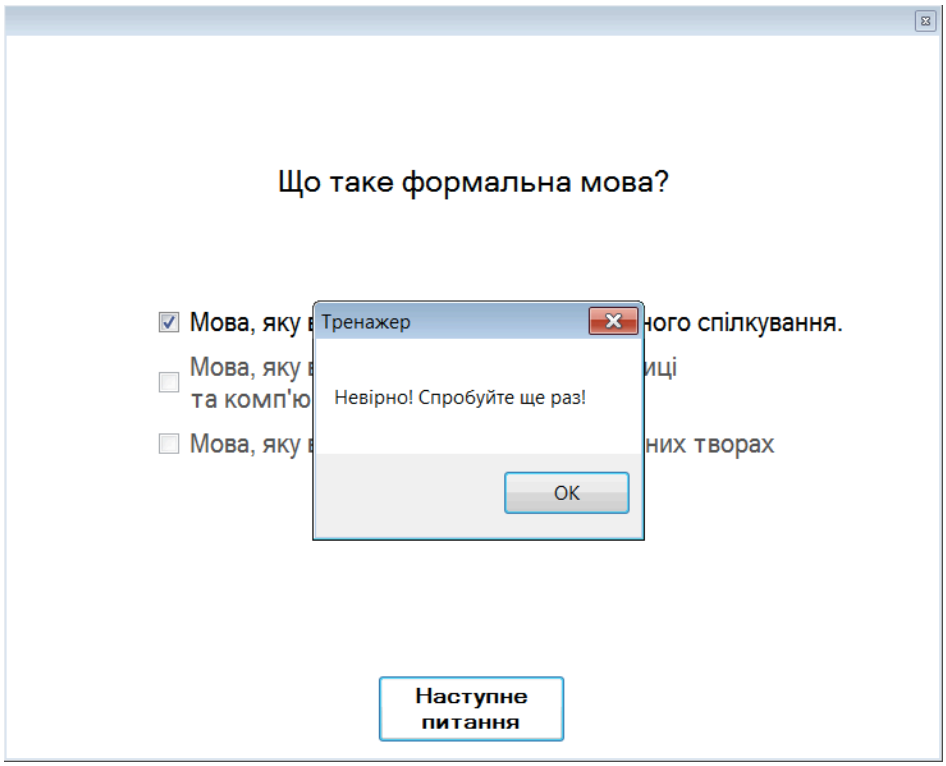


Рисунок 4.7 – сповіщення в разі помилкової відповіді.

Після успішної відповіді, користувач переходить до наступного питання, що стосується теми. (рис. 4.8)

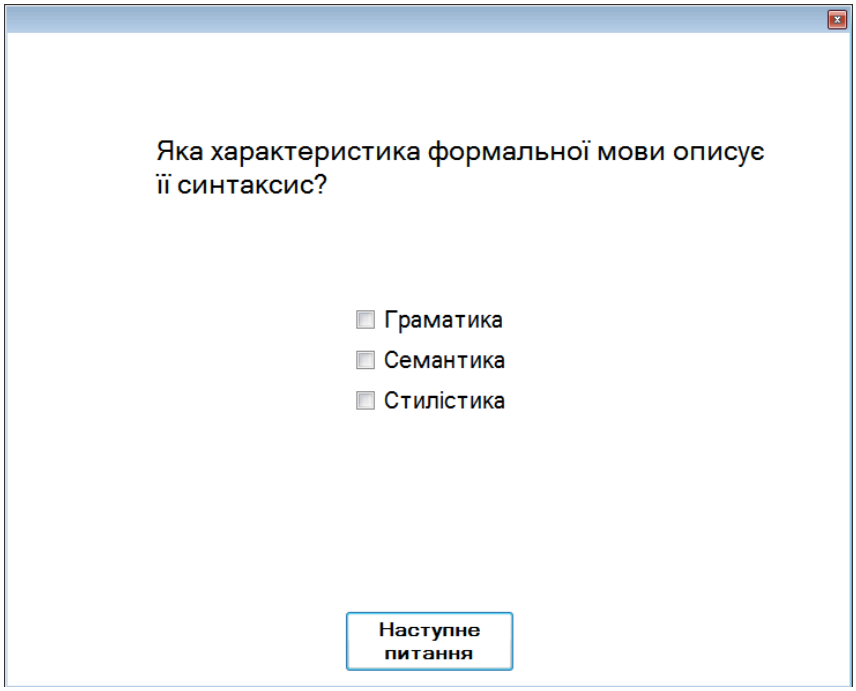


Рисунок 4.8 – вікно питання номер 2.

Після коректної відповіді, наступним кроком буде питання 3 (рис. 4.9)

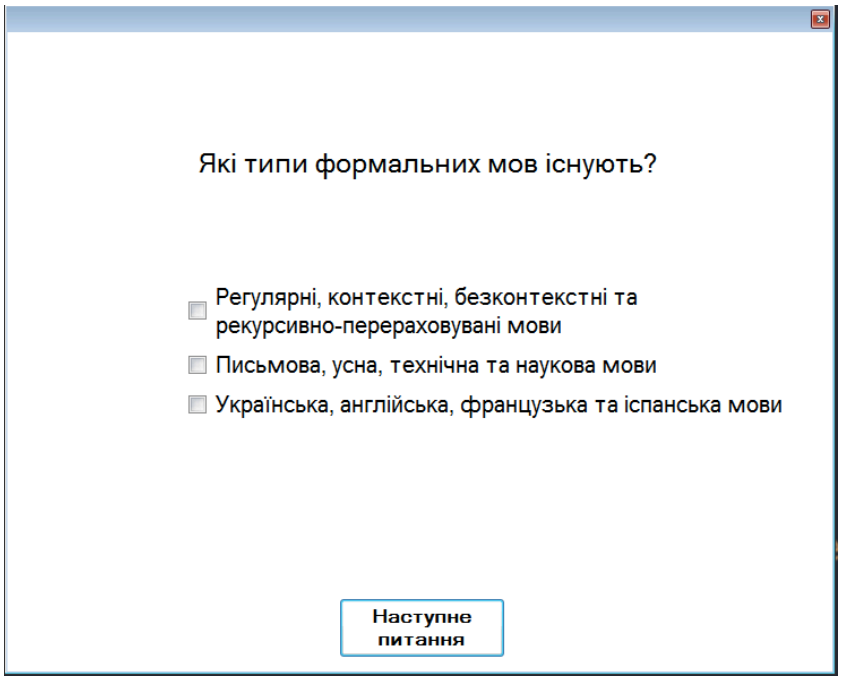


Рисунок 4.9 – вікно питання номер 3.

Після 3 питання, користувачу буде запропоновано 4 питання з теми у відповідному порядку. (рис. 5.0)

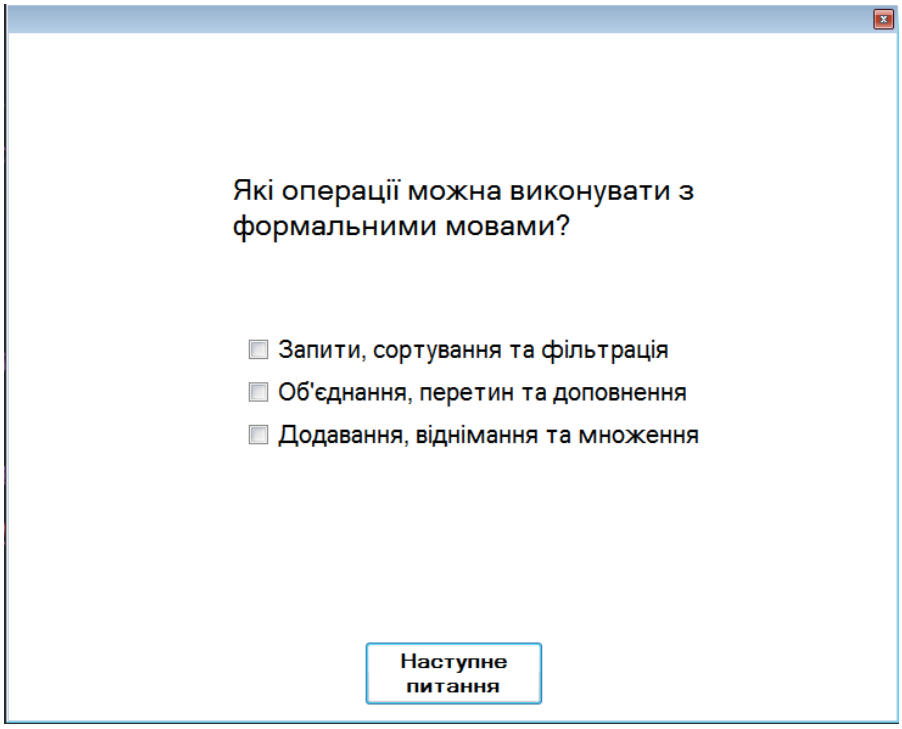


Рисунок 5.0 – вікно питання номер 4.

Після 4 питання, користувачу буде запропоновано 5 питання з теми у відповідному порядку (рис. 5.1)

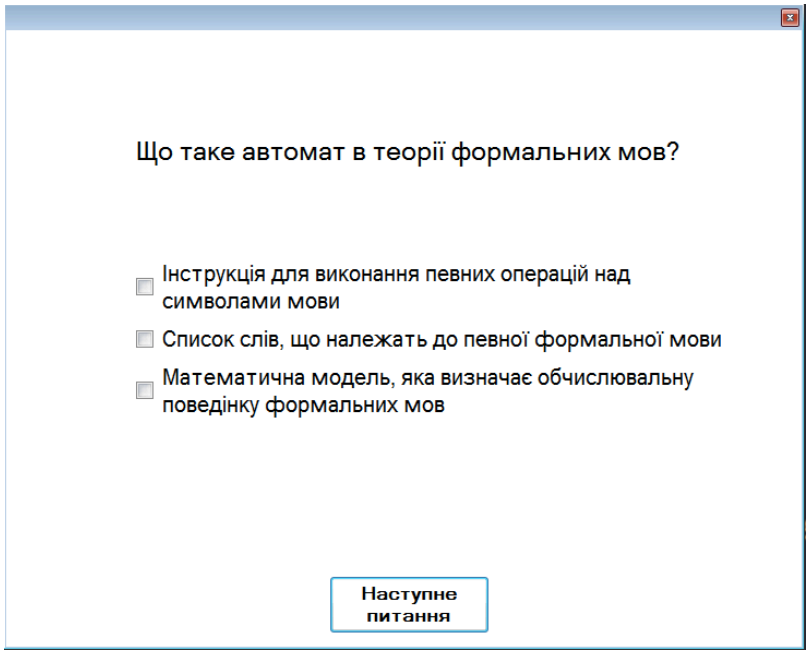


Рисунок 5.1 – вікно питання номер 5.

Після 5 питання, по порядку відкривається питання номер 6 (рис. 5.2)

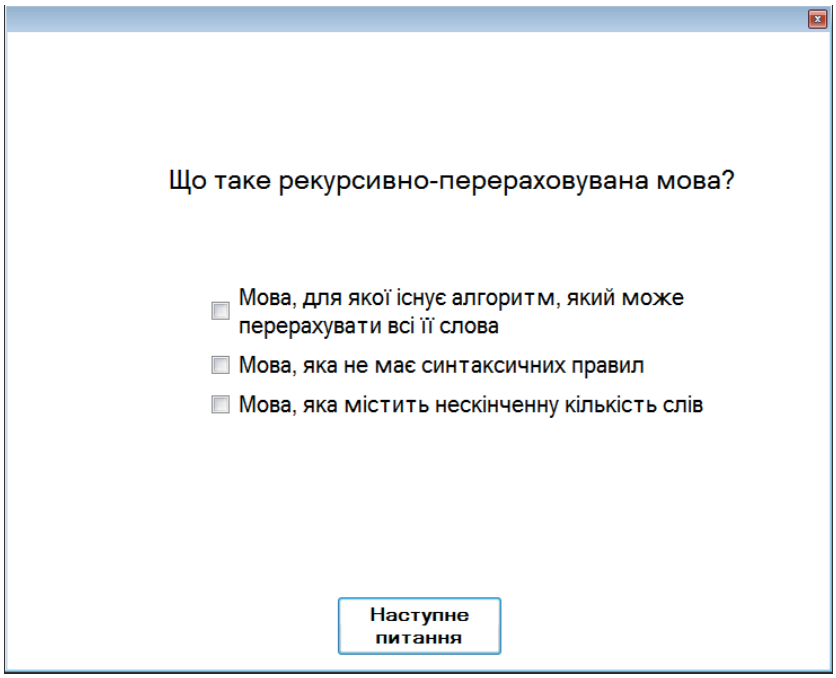


Рисунок 5.2 – вікно питання номер 6.

Потім користувачеві запропоновано питання 7 (рис. 5.2)

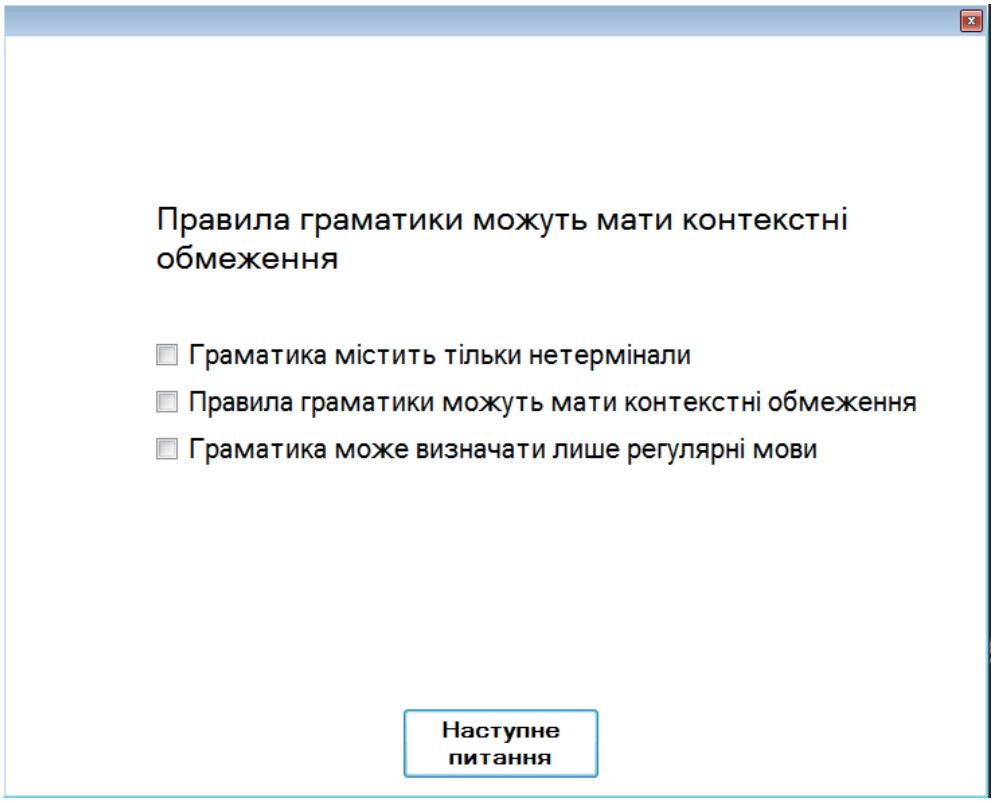


Рисунок 5.2 – вікно питання номер 7.

Далі відкривається питання номер 8 (рис. 5.3)

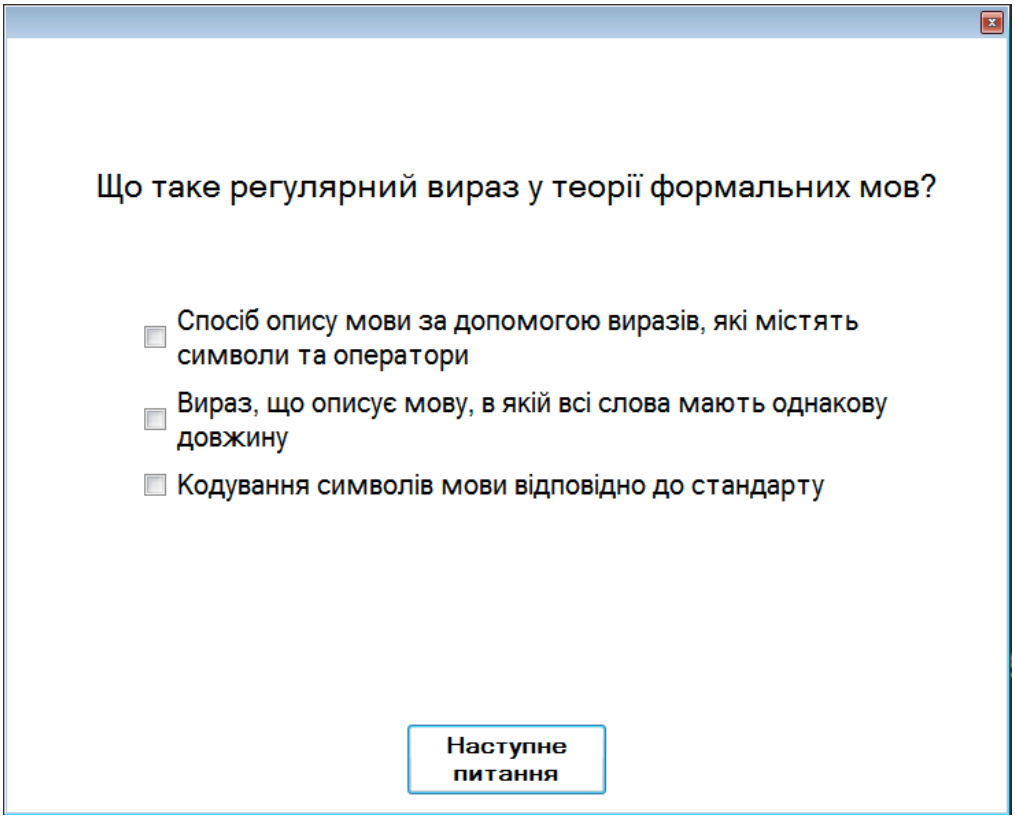


Рисунок 5.3 – вікно питання номер 8.

Подібним чином користувач продовжує відповідати на всі питання тренажеру. Після десятого питання, користувач натискає кнопку "ОК" і відкривається підсумкове вікно тренажеру, яке містить привітання та кнопку для завершення тренінгу. (рис. 5.4)

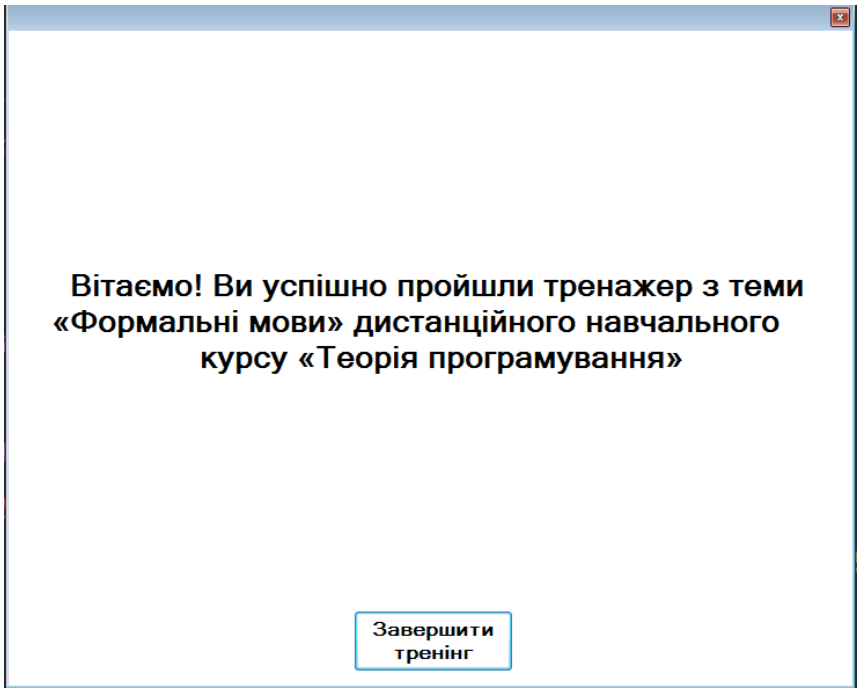


Рисунок 5.4 – підсумкове вікно тренажеру.

4.3. Обґрунтування вибору програмних засобів

Visual Basic є однією з найпоширеніших мов програмування у світі. Основною причиною його популярності є те, що він дозволяє програмістам швидко та легко створювати програми Windows.

Витоки Visual Basic знаходяться в мові програмування, створеній у 1964 році Джоном Кемені та Томасом Курцем. BASIC (універсальний символічний інструкційний код для початківців) спочатку був інтерпретованою мовою, яка була розроблена, щоб спростити процес програмування та зробити програмування більш доступним для всього світу. Використовуючи цю філософію, Microsoft інтегрувала інтерпретатор BASIC у свою операційну систему MS-DOS. Незважаючи на широке розповсюдження та відносну простоту, BASIC не міг конкурувати з більш швидкими скомпільованими мовами, такими як C або C++. Таким чином, BASIC зазвичай використовувався для тривіальних або освітніх цілей, тоді як «справжні» програми зазвичай розроблялися іншими мовами.

ВИСНОВКИ

Отже, дистанційне навчання має велику важливість у сучасному освітньому контексті. На підставі аналізу його переваг можна зробити наступні висновки:

1. Гнучкість та доступність: Дистанційне навчання надає студентам можливість вибирати свій власний розклад та місце навчання. Воно дозволяє людям займатися освітою без обмежень географічного розташування та робочих обов'язків.

2. Технологічний прогрес: Дистанційне навчання сприяє використанню сучасних технологій та інновацій у навчальному процесі. Інтерактивні платформи, відеоуроки, використання віртуальної реальності та інші інструменти покращують якість навчання та стимулюють активну участь студентів.

3. Самоорганізація та самодисципліна: Дистанційне навчання розвиває важливі навички самоорганізації, самостійності та самодисципліни. Студентам потрібно бути відповідальними за своє навчання, планувати час та виконувати завдання.

Завдання кваліфікаційної роботи успішно виконано. Розроблено інноваційний навчальний тренажер, який забезпечує зручне і ефективне навчання студентів з теми "Формальні мови" на дистанційному навчальному курсі "Теорія програмування".

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Черненко О. О. Курсовий проєкт із фаху: методичні рекомендації щодо оформлення пояснювальних записок до курсового проєкту для студентів спеціальності 122 Комп'ютерні науки освітня програма «Комп'ютерні науки» ступеня бакалавра, магістра / О. О. Черненко. – Полтава : ПУЕТ, 2022. – 58 с. – 1 електрон. опт. диск (CVD-ROM).
2. Дистанційне навчання [Електронний ресурс]. – Режим доступу: https://life.pravda.com.ua/society/2020/07/2/241517/
3. Формальні мови [Електронний ресурс]. – Режим доступу: https://studfile.net/preview/3746924/page:5/
4. Microsoft Visual Studio [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio
5. Microsoft Visual Basic [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Visual_Basic
6. Системи дистанційного навчання: огляд, аналіз, вибір. [Електронний ресурс]. – Режим доступу: http://ena.lp.edu.ua/bitstream/ntb/10662/1/14.pdf
7. Бібліографічний запис. Бібліографічний опис. Загальні вимоги та правила складання: ДСТУ 7.1-2006. – [Чинний від 2007-07-01]. – К. : Держспоживстандарт України, 2007. – 47 с.
8. Основи програмування мовою Visual BASIC 6.0 [Електронний ресурс]. – Режим доступу: https://kursoviks.com.ua/bd_kompyuterni/article_post/65-lektsiya-osnovi-programuvannya-movoyu-visual-basic-6-0
9. Середовище розробки Visual Basic [Електронний ресурс]. – Режим доступу: http://nikolay.in.ua/navchaemos/visual-basic/osnovni-ponyattya/38-seredovishche-rozrobki-visual-basic
10. Вбудовані функції Visual Basic [Електронний ресурс]. – Режим доступу: http://nikolay.in.ua/navchaemos/visual-basic/osnovni-ponyattya/129-vbudovani-funktsiji-visual-basic

11. Програмування в середовищі Visual Basic [Електронний ресурс]. – Режим доступу: http://lib.kart.edu.ua/bitstream/123456789/6106/1/%D0%9A%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82%20%D0%BB%D0%B5%D0%BA%D1%86%D1%96%D0%B9.pdf

12. Microsoft Visual Basic 2010 Step by Step [Електронний ресурс]. – Режим доступу: https://books.google.com.ua/books?hl=uk&lr=&id=Ap9CAwAAQBAJ&oi=fnd&pg=PT18&dq=visual+basic&ots=n5NfMNuX8q&sig=8OFEk2tHiXjaKynsUtD6SgQKabY&redir_esc=y#v=onepage&q=visual%20basic&f=false

ДОДАТОК А. КОД ПРОГРАМИ

```
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()> _
Partial Class Step0
    Inherits System.Windows.Forms.Form

    <System.Diagnostics.DebuggerNonUserCode()> _
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub
    Private components As System.ComponentModel.IContainer
    <System.Diagnostics.DebuggerStepThrough()> _
    Private Sub InitializeComponent()
        Me.Button1 = New System.Windows.Forms.Button()
        Me.Label1 = New System.Windows.Forms.Label()
        Me.Label2 = New System.Windows.Forms.Label()
        Me.Label3 = New System.Windows.Forms.Label()
        Me.SuspendLayout()
        '
        'Button1
        '
        Me.Button1.BackColor =
System.Drawing.SystemColors.ControlLightLight
        Me.Button1.Cursor = System.Windows.Forms.Cursors.Hand
        Me.Button1.Font = New System.Drawing.Font("Microsoft Sans Serif",
10.2!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
CType(204, Byte))
        Me.Button1.Location = New System.Drawing.Point(554, 488)
        Me.Button1.Name = "Button1"
        Me.Button1.Size = New System.Drawing.Size(138, 53)
        Me.Button1.TabIndex = 0
        Me.Button1.Text = "Розпочати!"
        Me.Button1.UseVisualStyleBackColor = False
        '
        'Label1
        '
        Me.Label1.AutoSize = True
```

```
        Me.Label1.Font = New System.Drawing.Font("Microsoft Sans Serif",
16.0!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
CType(204, Byte))
        Me.Label1.Location = New System.Drawing.Point(42, 181)
        Me.Label1.Name = "Label1"
        Me.Label1.Size = New System.Drawing.Size(643, 93)
        Me.Label1.TabIndex = 1
        Me.Label1.Text = "            Вітаємо Вас в тренажері з теми " &
Global.Microsoft.VisualBasic.ChrW(13) &
Global.Microsoft.VisualBasic.ChrW(10) & "«Формальні мови»
дистанційного нав" & _
            "чального" & Global.Microsoft.VisualBasic.ChrW(13) &
Global.Microsoft.VisualBasic.ChrW(10) & "            курсу «Теорія
програмування»      "
        '
        'Label2
        '
        Me.Label2.AutoSize = True
        Me.Label2.Font = New System.Drawing.Font("Microsoft Sans Serif",
12.0!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
CType(204, Byte))
        Me.Label2.Location = New System.Drawing.Point(12, 500)
        Me.Label2.Name = "Label2"
        Me.Label2.Size = New System.Drawing.Size(449, 25)
        Me.Label2.TabIndex = 2
        Me.Label2.Text = "Розробив:  студент групи КН-41 Козир В.А."
        '
        'Label3
        '
        Me.Label3.AutoSize = True
        Me.Label3.Font = New System.Drawing.Font("Microsoft Sans Serif",
12.0!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
CType(204, Byte))
        Me.Label3.Location = New System.Drawing.Point(12, 530)
        Me.Label3.Name = "Label3"
        Me.Label3.Size = New System.Drawing.Size(518, 25)
        Me.Label3.TabIndex = 3
        Me.Label3.Text = "Науковий керівник:  доц., к.ф.-м.н. Черненко О.О."
        '
        'Step0
        '
        Me.AutoScaleDimensions = New System.Drawing.SizeF(8.0!, 16.0!)
        Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font
        Me.BackColor = System.Drawing.SystemColors.ControlLightLight
        Me.ClientSize = New System.Drawing.Size(729, 565)
```

```
        Me.Controls.Add(Me.Label3)
        Me.Controls.Add(Me.Label2)
        Me.Controls.Add(Me.Label1)
        Me.Controls.Add(Me.Button1)
        Me.FormBorderStyle =
System.Windows.Forms.FormBorderStyle.FixedToolWindow
        Me.Name = "Step0"
        Me.StartPosition =
System.Windows.Forms.FormStartPosition.CenterScreen
        Me.ResumeLayout(False)
        Me.PerformLayout()

    End Sub
    Friend WithEvents Button1 As System.Windows.Forms.Button
    Friend WithEvents Label1 As System.Windows.Forms.Label
    Friend WithEvents Label2 As System.Windows.Forms.Label
    Friend WithEvents Label3 As System.Windows.Forms.Label

End Class
```