ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ
Навчально-науковий інститут заочно-дистанційного навчання
Форма навчання заочна
Кафедра комп'ютерних наук та інформаційних технологій

Допускається до захисту
Завідувач кафедри
_____ Олена ОЛЬХОВСЬКА
(підпис)

«___» _____ 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА
на тему
РОЗРОБКА ТРЕНАЖЕРУ З ТЕМИ «ФОРМАЛЬНІ ГРАМАТИКИ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ ПРОГРАМУВАННЯ»

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

Виконавець роботи Фролов Данило Олегович
_____ «____»_____ 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
String	Це тип даних в мові програмування Visual Basic, який використовується для представлення текстових рядків. Він дозволяє зберігати послідовність символів, таких як літери, цифри та спеціальні символи.
Dim	Це ключове слово в мові програмування Visual Basic, яке використовується для оголошення змінних. Воно вказує компілятору, що змінна буде використовуватися в програмі і який тип даних вона буде зберігати.
Integer	Це тип даних в мові програмування Visual Basic, який використовується для зберігання цілих чисел без десяткової частини.
Close	Це метод або дія, яка використовується для закриття об'єкта або ресурсу в мові програмування Visual Basic.

ВСТУП

В сучасному світі, дистанційне навчання стає все більш популярним і важливим аспектом освіти. Завдяки стрімкому розвитку технологій і доступності Інтернету, студенти тепер мають можливість отримати якісну освіту, не виходячи зі зручності свого домівки чи офісу. Дистанційне навчання відкриває двері до незліченних можливостей для освіти та саморозвитку.

Однією з головних причин необхідності дистанційного навчання є глобалізація та мобільність. Зараз люди можуть здобувати освіту або покращувати свої навички, не залежно від місця проживання чи розташування університету.

Крім того, дистанційне навчання забезпечує гнучкість і доступність освіти. Студенти можуть самостійно планувати свій навчальний графік, пристосовуючи його до своїх потреб і розкладу. Вони можуть навчатися власним темпом і повертатися до матеріалу, якщо потрібно. Крім того, дистанційне навчання часто є вигіднішим з фінансової точки зору, оскільки не вимагає витрат на переїзд, проживання в гуртожитку або витрат на дорогу до учбового закладу.[2]

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

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

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

Кваліфікаційна робота складається з чотирьох розділів, кожен з яких зосереджений на різних аспектах розробки навчального тренажеру. У першому розділі наведено постановку завдання, яке передує розробці тренажеру. Другий розділ присвячений формальним граматикам і детально розглядає їх властивості та особливості. Третій розділ містить теоретичний матеріал, пов'язаний з тематикою тренажеру, а також розгорнутий опис алгоритму роботи та блок-схем тренажеру. У четвертому розділі описується процес програмної реалізації тренажеру та надається інструкція для користувача.

1. ПОСТАНОВКА ЗАВДАННЯ

Основним завданням кваліфікаційної роботи є розробка програмного забезпечення для впровадження тренажеру з теми "Формальні граматики" у дистанційному навчальному курсі "Теорія програмування".

У рамках цього проекту передбачається виконання наступних завдань:

1. Ознайомитись з матеріалом, пов'язаним з "Формальними граматиками".

2. Розробити алгоритм роботи тренажеру.

3. Створити блок-схему алгоритму роботи тренажеру.

4. Реалізувати програмні елементи тренажеру.

5. Описати результати програмної реалізації готових елементів тренажеру.

У процесі програмування елементів програмного забезпечення тренажеру необхідно приділити увагу таким вікнам:

1. Початкове вікно тренажеру, де буде відображатись інформація про тематику тренажеру, кнопка, що розпочинає тренування, а також відомості про розробника та наукового керівника проекту з фаху.

2. Інші вікна тренажеру, які міститимуть питання з теми тренажеру у форматі тесту, на які користувач повинен дати правильну або неправильну відповідь. Основним завданням курсового проекту з фаху є розробка програмного забезпечення для впровадження елементів тренажеру з теми "Формальні граматики" у дистанційному навчальному курсі "Теорія програмування".

У рамках цього проекту передбачається виконання наступних завдань:

1. Ознайомитись з матеріалом, пов'язаним з "Формальними граматиками".

2. Розробити алгоритм роботи тренажеру.

3. Створити блок-схему алгоритму роботи тренажеру.

4. Реалізувати програмно тренажер.

5. Описати результати програмної реалізації тренажеру.

У процесі програмування програмного забезпечення тренажеру необхідно приділити увагу таким вікнам:

1. Початкове вікно тренажеру, де буде відображатись інформація про тематику тренажеру, кнопка, що розпочинає тренування, а також відомості про розробника та наукового керівника проекту з фаху.

2. Інші вікна тренажеру, які міститимуть питання з теми тренажеру у форматі тесту, на які користувач повинен дати правильну або неправильну відповідь.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1. Синтаксичний розбір формальних граматик

Виведення ланцюжка за допомогою правил граматики може бути заданий не тільки у вигляді синтаксичного дерева [4,5,9]. Якщо пронумерувати правила граматики, то послідовність номерів використовуваних правил також задає виведення. Визначення. Послідовність номерів правил граматики Г, застосування яких дозволяє побудувати виведення розглянутого ланцюжка σ з початкового символу граматики, називається синтаксичним розбором σ. Наприклад, у граматиці Г1.9: 18 R ={ 1. E →E +T 2. E →T 3. T →T *P 4. T →P Vт = { i, +, * , (,) }, VA = {E, T, P}, 5. P →(E) 6. P → i }, правила якої пронумеровані, виведення E ⇒ E +T ⇒ T +T ⇒ T *P +T ⇒ P *P +T ⇒ i * P +T ⇒ i * i +T ⇒ i * i +P ⇒ i * i + i має синтаксичний розбір [1, 2, 3, 4, 6, 6, 4, 6].

Якщо в процесі побудови виведення з'являються проміжні ланцюжки, що містять кілька нетермінальних символів, то можна продовжувати виведення, заміняючи кожний з ланцюжків. Таким чином, ті самі правила можуть бути використані при виводі ланцюжка у будь-якому порядку. Наприклад, виведення ланцюжка i + i в граматиці Г1..9 може бути отриманий десятьма різними способами.

Серед різного виведення найбільший інтерес становлять наступні два типи виведення. Визначення. Якщо при побудові виведення ланцюжка α при кожнім застосуванні правила заміняється найлівіший нетермінальний символ, то таке виведення називається лівим, або лівостороннім виведенням α. Якщо при побудові виведення α, завжди заміняється найправіший нетермінальний символ проміжного ланцюжка, то виведення називається правим, або правостороннім виведенням α. Наприклад вище наведене виведення ланцюжка i * i + i в граматиці Г1.9 є лівостороннім виведенням. Слід зазначити, що різному виведенню 19 ланцюжка i+i в

граматиці Г1..9 відповідає те ж саме синтаксичне дерево. Аналогічна ситуація має місце і при виводі ланцюжка і * і + і.

Визначення. Ланцюжок мови L(Г) називається неоднозначним, якщо для його виведення існує більш ніж одне синтаксичне дерево. Якщо граматика Г породжує неоднозначний ланцюжок, то вона називається неоднозначною. Неоднозначність може існувати не тільки в штучних мовах. Добре відомо, що в природних мовах можуть бути пропозиції, що допускають неоднозначне написання. Наприклад, "Пальто забруднило вікно". У цій фразі не ясно, що є підметом, а що доповненням. Іншим прикладом служить англійська фраза: "They are flying planes", що може бути зрозуміла подвійно: “Вони пілотують літак” або “Це літаки, що летять”. Властивість неоднозначності є вкрай небажаною для штучних мов, оскільки вона не дозволяє однозначно відновити дерево виведення за заданим ланцюжком мови. У загальному випадку можна зробити такий висновок: 1) кожному ланцюжку, виведеному в граматиці, може відповідати одне або кілька синтаксичних дерев; 2) кожному синтаксичному дереву можуть відповідати різні виведення; 3) кожному синтаксичному дереву відповідають єдиний правий і єдиний лівий виводи. Крім того, варто підкреслити, що та сама мова може бути отримана за допомогою різних граматик. Визначення. Дві граматики – Г1 і Г2 – називаються еквівалентними, якщо вони породжують ту саму мову, тобто L(Г1) = L(Г2).

Схема граматики містить правила виведення, що визначають синтаксис мови, або, іншими словами, можливі компоненти і конструкції ланцюжків породжуваної мови. Для задання правил використовуються різні форми опису: символічна, форма Наура-Бекуса, ітераційна форма і синтаксичні діаграми. 22 У роботах, пов'язаних з розглядом загальних властивостей граматик, звичайно застосовують символічну форму завдання правил.

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1. Формальні граматики

Математичні моделі, що використовують подання текстів у вигляді послідовності символів, називають формальними мовами і граматиками Визначення. Кінцева множина символів, неподільних у даному розгляді, називається словником чи алфавітом, а символи, що входять у множину, – буквами алфавіту.

Наприклад, алфавіт A = {2, b, c, +, !} містить 5 букв, а алфавіт B = {00, 01, 10, 11} містить 4 букви, кожна з яких складається з двох символів. Визначення. Послідовність букв алфавіту називається словом чи ланцюжком у цьому алфавіті. Число букв, що входять у слово, називається його довжиною. Наприклад, слово a= 2bc в алфавіті A має довжину l(a) = 3, а слово b = 0000110010 в алфавіті B має довжину l (b) = 5. Якщо заданий алфавіт A, то позначимо A * множину всяких ланцюжків, що можуть бути побудовані з букв алфавіту A. При цьому передбачається, що порожній ланцюжок, який позначимо знаком $, також входить у множину A * . Визначення. Формальною граматикою Г, що породжує множину символів, називається наступна сукупність чотирьох об'єктів: Г = { VT , I, VA , R }, де VT – термінальний алфавіт (словник); букви цього алфавіту називаються термінальними символами; з них будуються ланцюжки породжувані граматикою; VA – нетермінальний, допоміжний алфавіт (словник); букви цього алфавіту використовуються при побудові ланцюжків; вони можуть входити в проміжні ланцюжки, але не повинні входити в результат породження; I – початковий символ граматики $I \in VA$; R – множина правил виведення вигляду, що $\alpha \to \beta$, де α і β – ланцюжки, побудовані з букв алфавіту $V_T \cup V_A$, що називають повним алфавітом (словником) граматики Г. До множини правил граматики можуть також входити правила з порожньою правою частиною вигляду $E \to$. Щоб

уникнути невизначеності через відсутність символу в правій частині правила, домовимося використовувати символ порожнього ланцюжка, записуючи таке правило у вигляді E → $. Щоб встановити правила побудови ланцюжків, породжуваних граматикою, введемо наступні поняття. Визначення. Нехай τ→ γ – правило граматики Г і α → χ'τ χ" – ланцюжок символів, причому χ', χ" ∈ (Vт ∪VA) * . Тоді ланцюжок β→ χ' γ χ " може бути отриманий з ланцюжка α шляхом застосування правила τ → γ . У цьому випадку говорять, що ланцюжок β безпосередньо виведений з ланцюжка α і позначають α ⇒β . Визначення. Якщо задана сукупність ланцюжків Ω = (ϖ0, ϖ1,...,ϖn), за якої існує послідовність безпосередніх виводів: ϖ0 ⇒ ϖ1, ϖ1 ⇒ ϖ2, ... ,ϖ n-1 ⇒ϖ n то таку послідовність називають виведення ϖn з ϖ0 у граматиці Г і позначають ϖ 0 ⇒* ϖn. Визначення. Множина кінцевих ланцюжків термінального алфавіту Vт граматики Г, виведених з початкового символу I, називається мовою, породжуваною граматикою Г, і позначається L(Г). L(Г) = {ϖ ∈ Vт * | ⇒*ϖ }.

Визначення. Якщо мова, породжувана граматикою Г, не містить жодного кінцевого ланцюжка (кінцевого слова), то вона називається порожньою. Твердження. Для того, щоб мова L(Г) не була порожньою, у множині R повинне бути хоча б одне правило вигляду r = χ → ψ, де ψ ∈ Vт * і повинно існувати виведення I ⇒* χ. 1.4 Типи формальних мов і граматик У теорії формальних мов виділяються 4 типи граматик, яким відповідають 4 типи мов. Такий розподіл зветься «граматична структура Хомського». Ці граматики виділяються шляхом накладення обмежень на правила граматики. Граматики типу 0 Граматики типу 0, що називаються граматиками загального вигляду, не мають ніяких обмежень на правила породження.

Граматики типу 1, що називаються також контекстно-залежними граматиками, не допускають використання будь-яких правил. Правила

виведення в таких граматиках повинні мати вигляд: $\chi 1 A \chi 2 \rightarrow \chi 1 w \chi 2$, де $\chi 1$, $\chi 2$ – ланцюжки, можливо порожні, з множини $(Vт \cup Va)^*$, символ $A \in Va$ і ланцюжок $w \in (Vт \cup Va)^*$. Ланцюжки $\chi 1$ і $\chi 2$ залишаються незмінними при застосуванні правила, тому їх називають контекстом (відповідно лівим і правим), а граматику – контекстно-залежною. Граматики типу 1 значно зручніші на практиці, ніж граматики типу 0, оскільки в лівій частині правила заміняється завжди один нетермінальний символ, який можна зв'язати з деяким синтаксичним поняттям, у той час як у граматиці типу 0 можна заміняти відразу кілька символів, у тому числі і термінальних.

Граматики типу 2 називають контекстно-вільними (КВ) граматиками, або безконтекстними граматиками. Правила виведення таких граматик мають вигляд: $A \rightarrow \alpha$, де $A \in Va$ і $\alpha \in (Vт \cup Va)^*$. Очевидно, що ці правила виходять із правил граматики типу 1 за умови $\chi 1 = \chi 2 = \$$. Оскільки контекстні умови відсутні, то правила КВ- граматик виходять простіші, ніж правила граматик типу 1. Саме такі граматики використовують для опису мов програмування.

Граматики типу 3 називають автоматними граматиками (А-граматиками). Правила виведення в таких граматиках мають вигляд: $A \rightarrow a$, або $A \rightarrow aB$, або $A \rightarrow B\ a$, де $a \in Vт$, $A, B \in VA$, причому граматика може мати тільки правила вигляду $A \rightarrow aB$ – правосторонні правила, або тільки вигляду $A \rightarrow Ba$ – лівосторонні правила.

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

Початок алгоритму блоку завдань.

1. Що таке формальна граматика?
   a) Мова програмування
   b) Математична структура для опису мов
   c) Мовний вимір
   Відповідь: b) Математична структура для опису мов
2. Які основні компоненти складають формальну граматику?
   a) Символи, правила, аксіома
   b) Запитання, відповіді, тест
   c) Функції, змінні, оператори
   Відповідь: a) Символи, правила, аксіома
3. Які види символів можуть бути використані в формальній граматиці?
   a) Тільки літери алфавіту
   b) Літери алфавіту та спеціальні символи
   c) Тільки числа
   Відповідь: b) Літери алфавіту та спеціальні символи
4. Що таке нетермінальний символ в формальній граматиці?
   a) Символ, який може бути замінений іншим символом
   b) Символ, який не може бути замінений іншим символом
   c) Символ, який може бути використаний тільки у правилах граматики
   Відповідь: a) Символ, який може бути замінений іншим символом
5. Що таке термінальний символ в формальній граматиці?
   a) Символ, який може бути замінений іншим символом
   b) Символ, який не може бути замінений іншим символом
   c) Символ, який може бути використаний тільки у правилах граматики
   Відповідь: b) Символ, який не може бути замінений іншим символом

6. Що таке правило в формальній граматиці?

a) Послідовність символів, що описують заміну

b) Правило поводження в граматиці

c) Символ, який позначає кінцевий результат граматики

Відповідь: a) Послідовність символів, що описують заміну

7. Як називається процес заміни символів в формальній граматиці?

a) Деривація

b) Редагування

c) Перетворення

Відповідь: a) Деривація

8. Як називається мова, яку можна згенерувати за допомогою формальної граматики?

a) Початкова мова

b) Термінальна мова

c) Виведена мова

Відповідь: c) Виведена мова

9. Що таке контекстно-вільна граматика?

a) Граматика, в якій правила можуть бути застосовані в будь-якому контексті

b) Граматика, в якій правила можуть бути застосовані тільки у визначеному контексті

c) Граматика, в якій правила можуть бути застосовані тільки до термінальних символів

Відповідь: a) Граматика, в якій правила можуть бути застосовані в будь-якому контексті

10. Що таке рекурсивне правило в контекстно-вільній граматиці?

a) Правило, яке посилається на саме себе

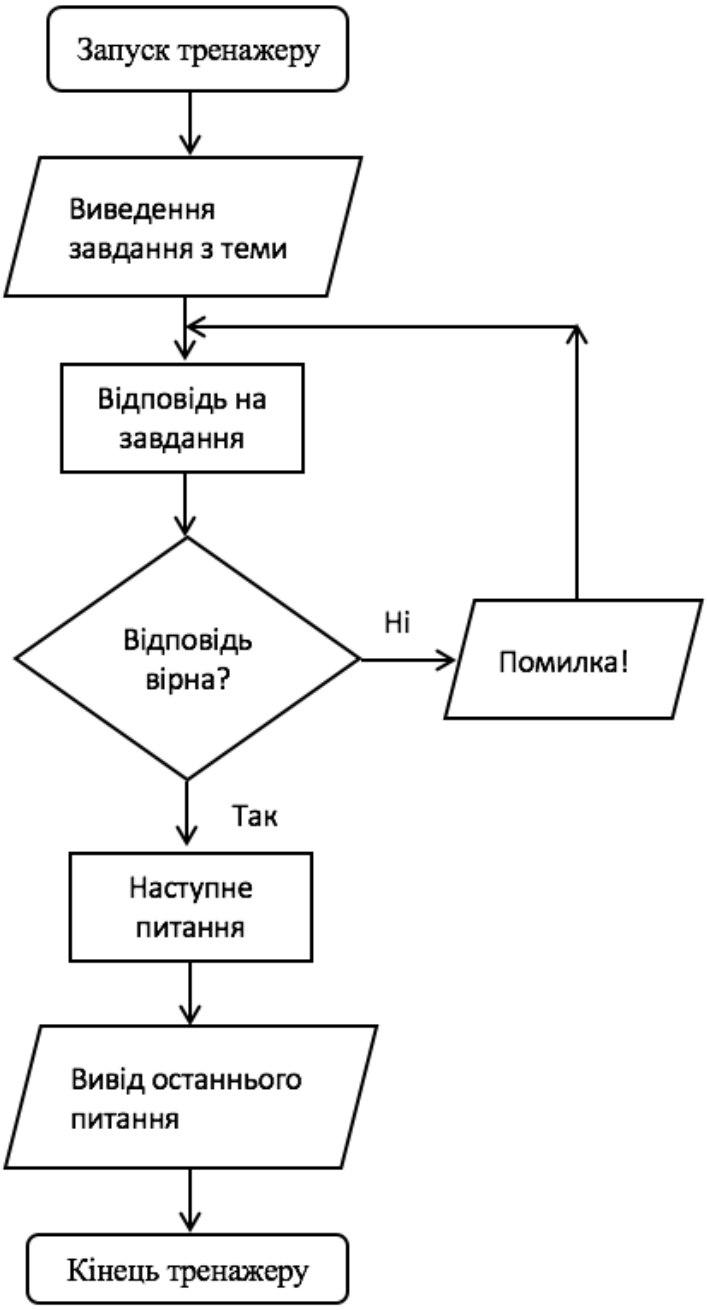
b) Правило, яке не має жодних залежностей

c) Правило, яке може бути застосоване тільки у визначеному контексті

Відповідь: a) Правило, яке посилається на саме себе

Кінець алгоритму блоку завдань.

3.3. Блок-схема тренажеру



4. ПРАКТИЧНА ЧАСТИНА

4.1 Опис програмної реалізації

Для програмної реалізації тренажеру були використані програмні компоненти, розроблені з використанням мови програмування Visual Basic.(рис. 4.1)

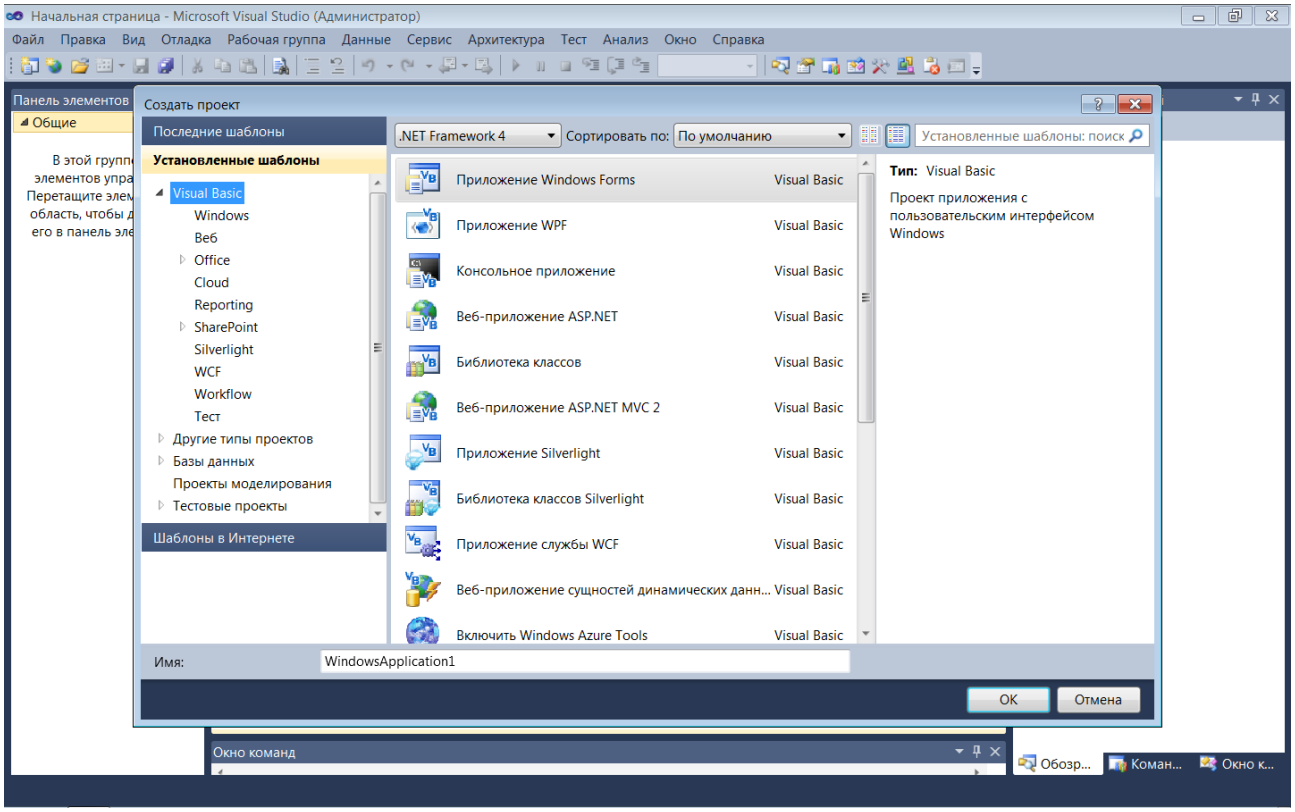


Рисунок 4.1 – налаштування нового проекту в програмному середовищі Visual Studio.

Після запуску нового проекту, наступним етапом є розробка початкової форми тренажеру, яка служить стартовим екраном. (рис. 4.2)

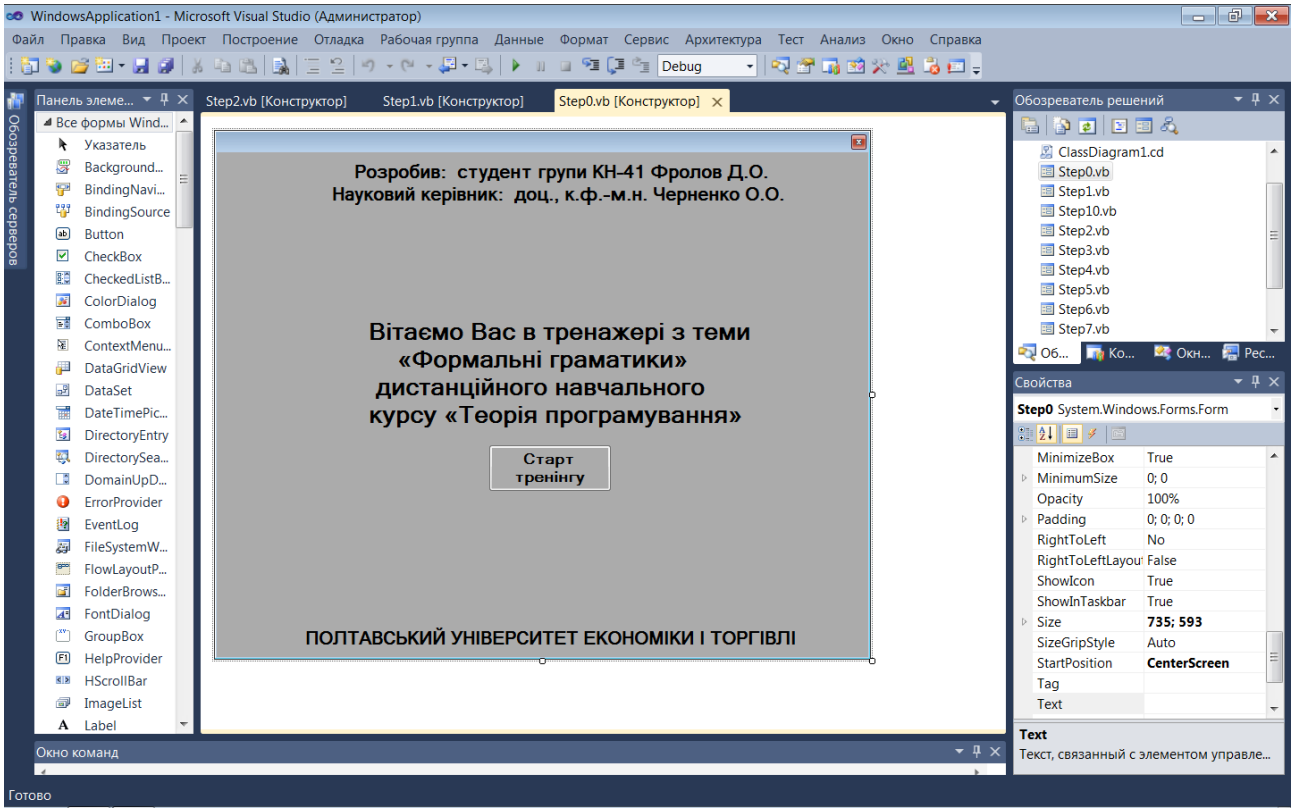


Рисунок 4.2 – початковий екран тренажеру.

Після відображення початкового екрану, наступним кроком є реалізація програмного коду для першого питання, яке базується на теоретичному матеріалі тренажеру. (рис. 4.3)

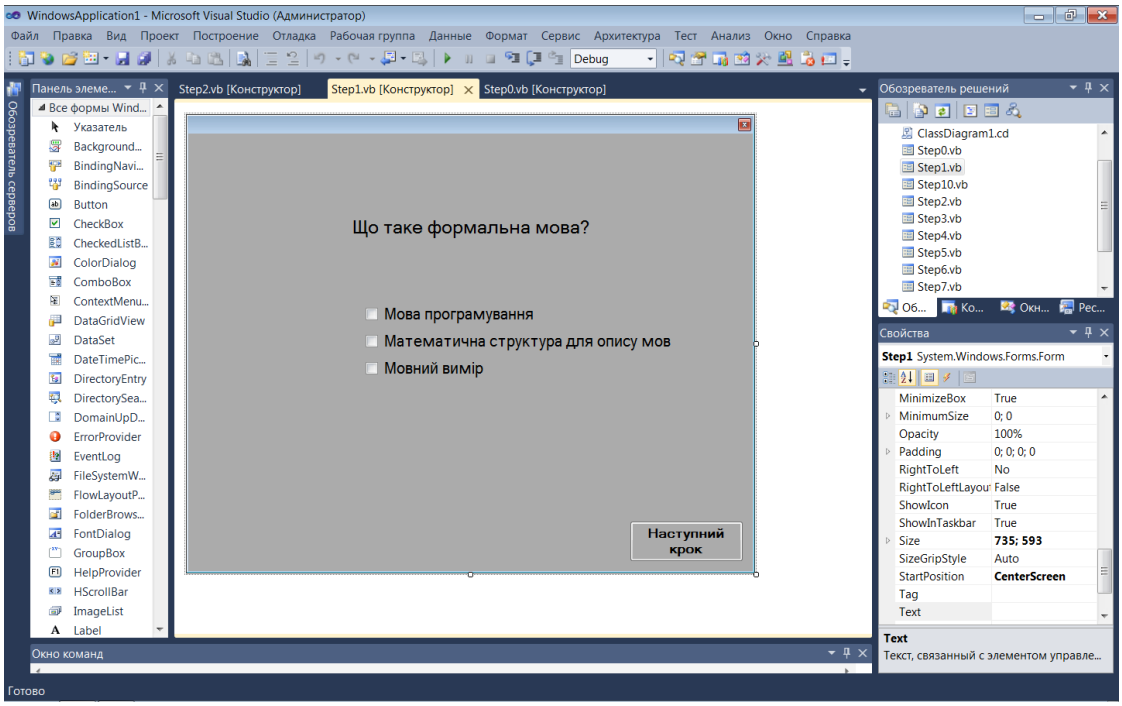


Рисунок 4.3 – перше вікно з теоретичним матеріалом.

Після першого вікна з теоретичним матеріалом, наступним кроком є створення подібного за дизайном вікна. (рис. 4.4)

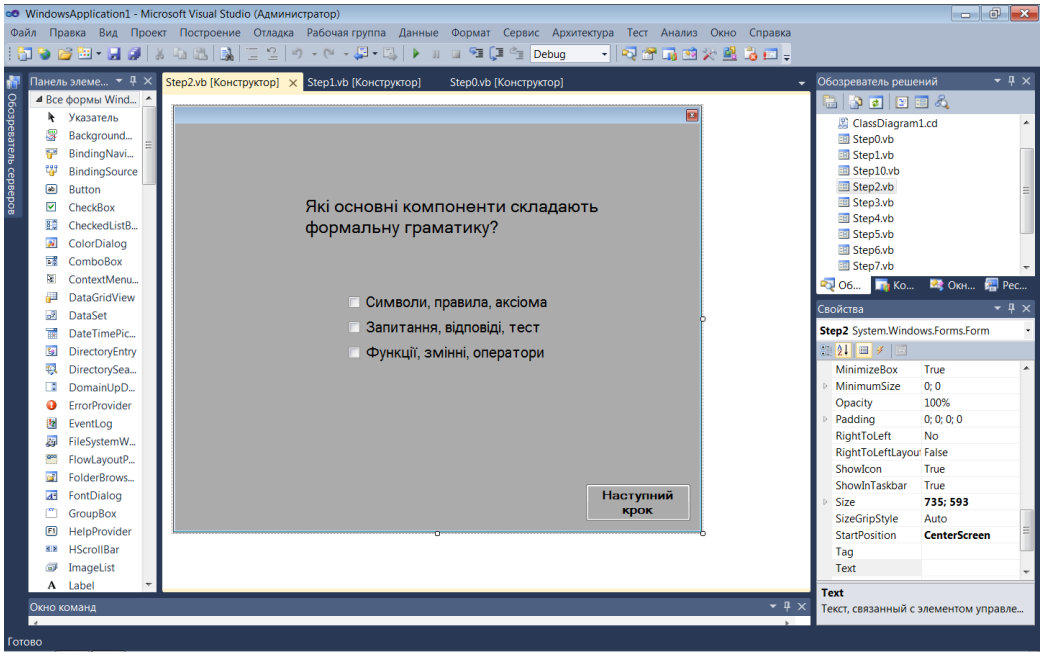


Рисунок 4.4 – подібне вікно з теоретичним питанням.

4.2. Інструкція по використанню навчального тренажеру

Після запуску тренажеру, користувачу автоматично відображається початковий екран. (рис. 4.4)

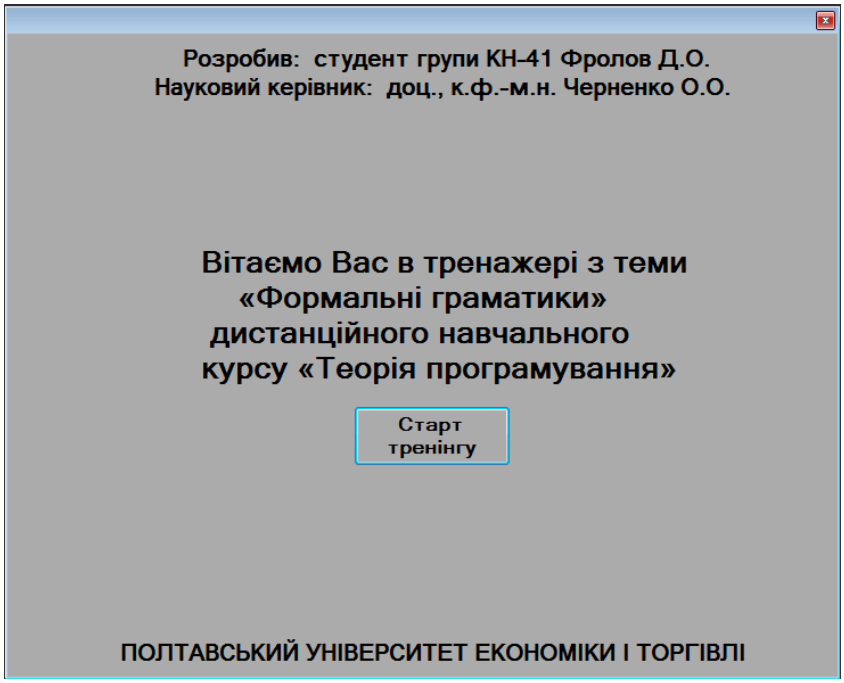


Рисунок 4.4 – початковий екран тренажеру.

Після натискання користувачем кнопки на формі, тренажер автоматично відображає перше питання, на яке користувач може відповісти, вибравши один з наданих варіантів відповіді. (рис. 4.5)

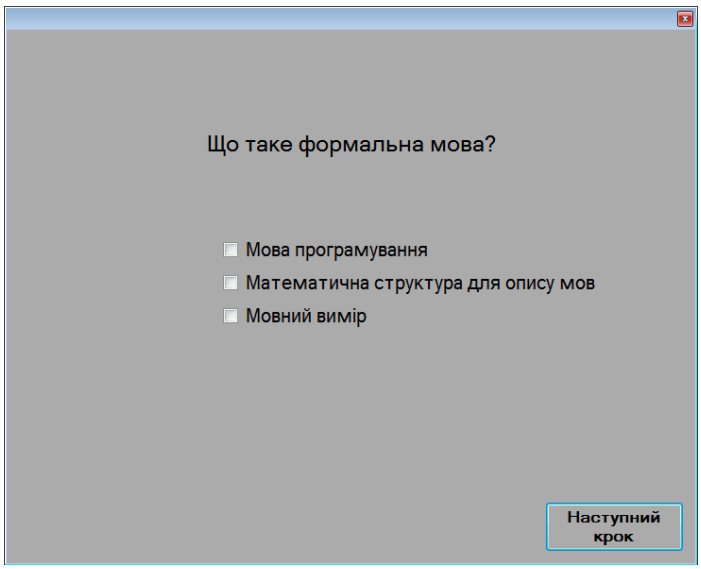


Рисунок 4.5 – перше завдання з теоретичного матеріалу.

Після того, як користувач обрав свій варіант відповіді, програма перевіряє цю відповідь і відображає вікно з додатковою інформацією для користувача. (рис. 4.6)

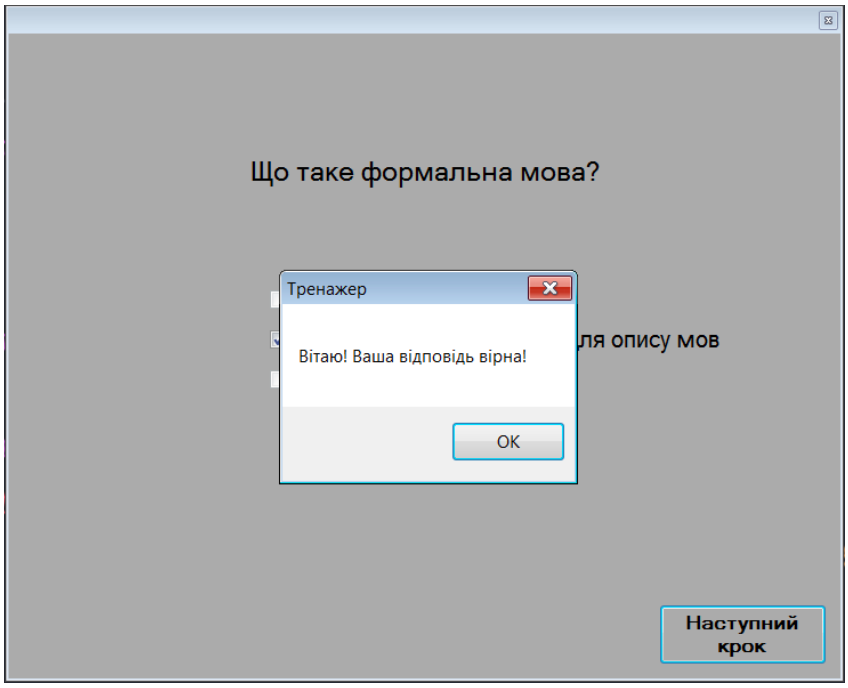


Рисунок 4.6 – вікно правильної відповіді.

Якщо користувач надає неправильну відповідь на питання, з'являється наступне попередження. (рис. 4.7)

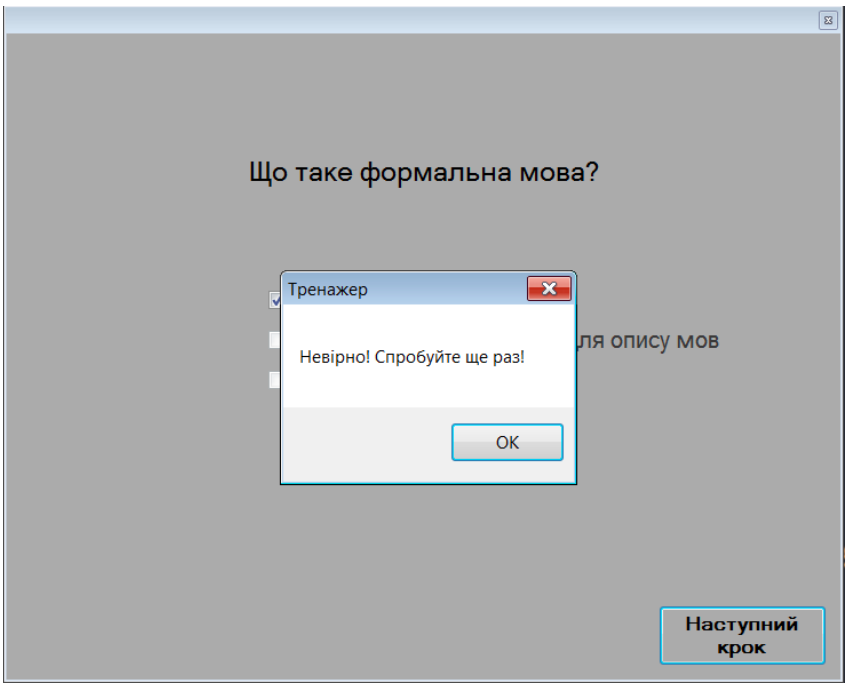


Рисунок 4.7 – вікно неправильної відповіді.

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

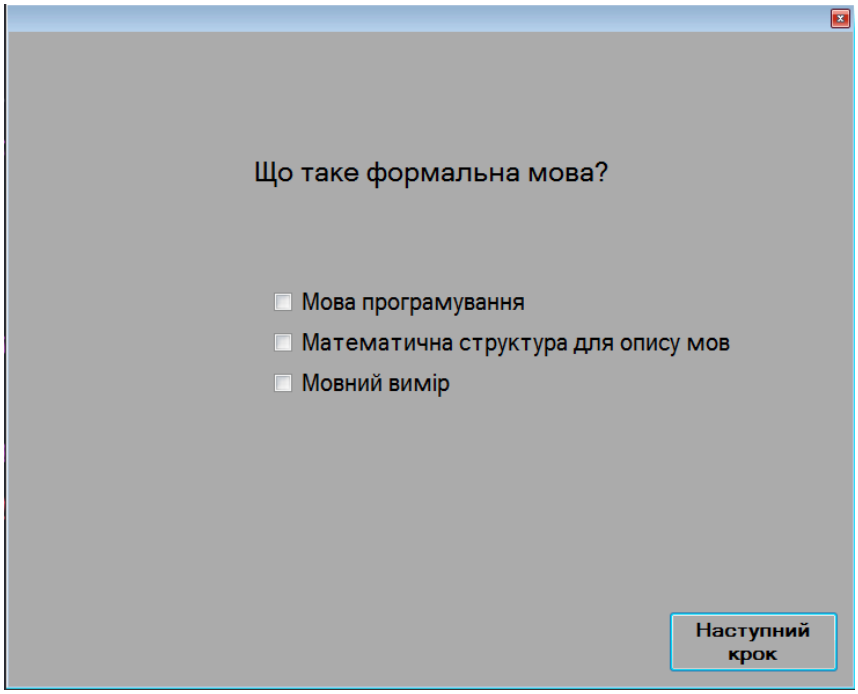


Рисунок 4.8 – вікно питання номер 2.

Після правильної відповіді, наступним питанням буде питання 3 (рис. 4.9)

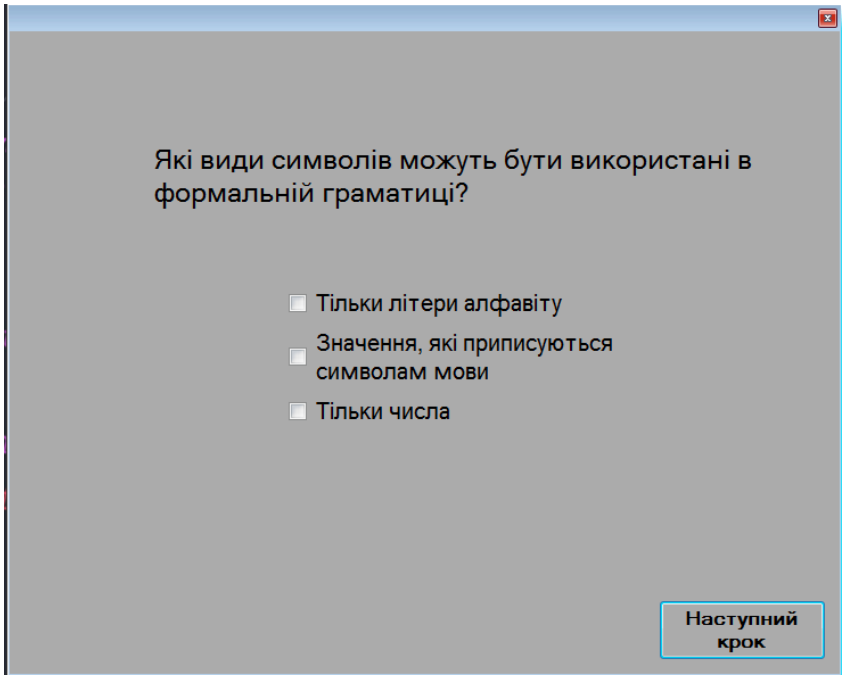


Рисунок 4.9 – вікно питання номер 3.

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

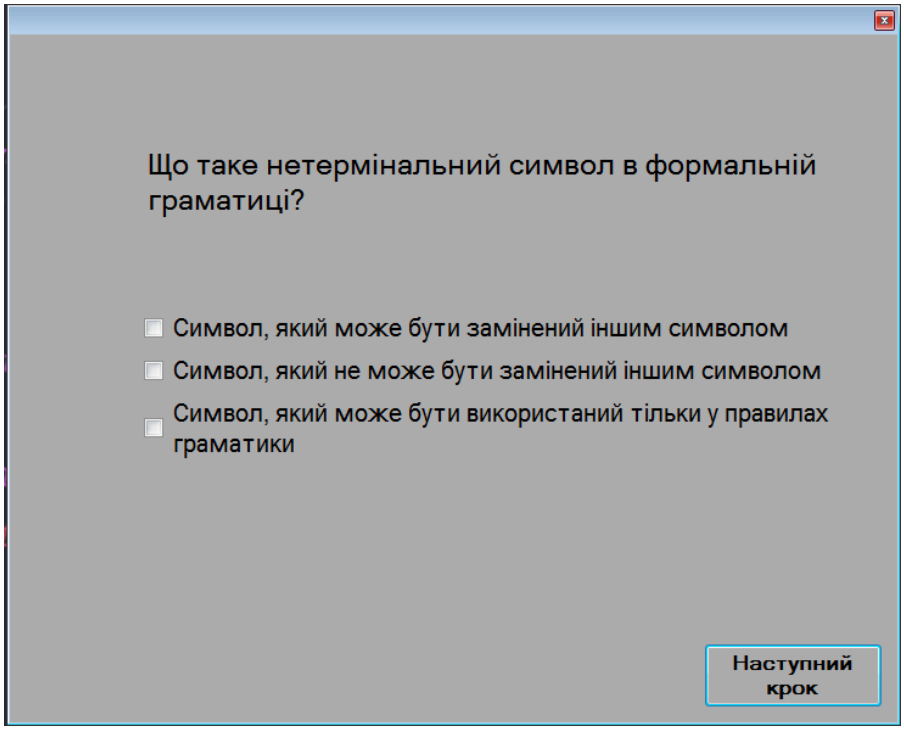


Рисунок 5.0 – вікно питання номер 4.

Після 4 питання, користувачу буде запропоновано 5 питання(рис. 5.1)

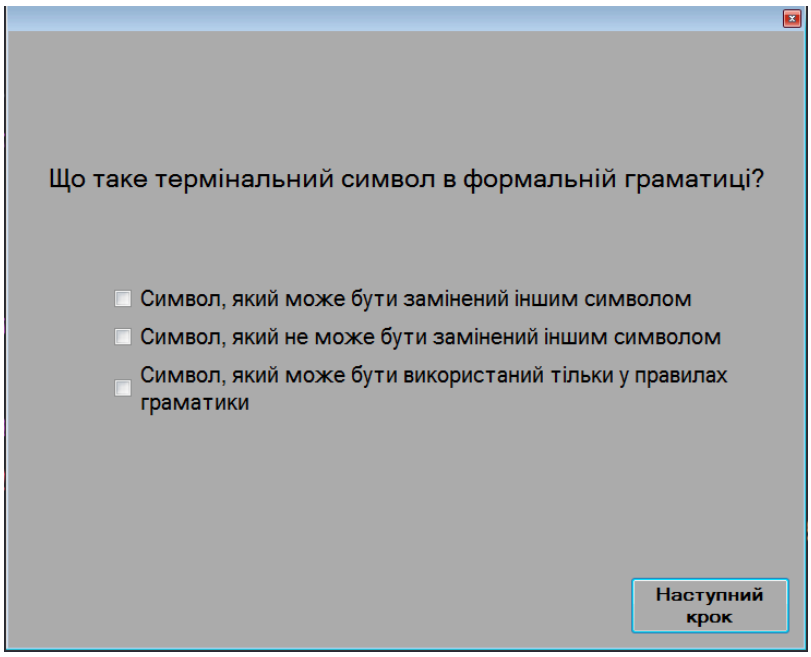


Рисунок 5.1 – вікно питання номер 5.

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

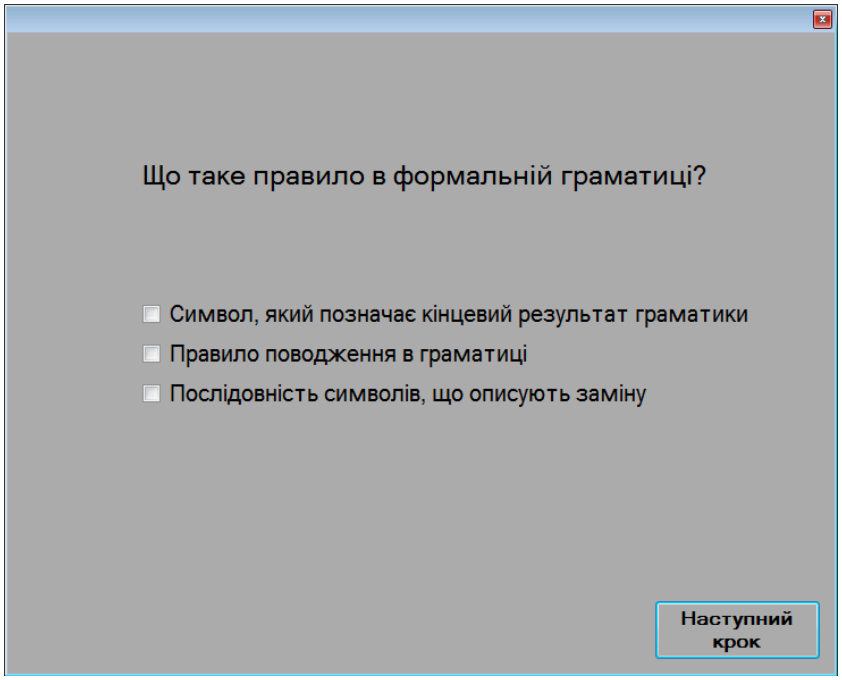


Рисунок 5.2 – вікно питання номер 6.

Потім користувачеві запропоновано питання 7 (рис. 5.2)

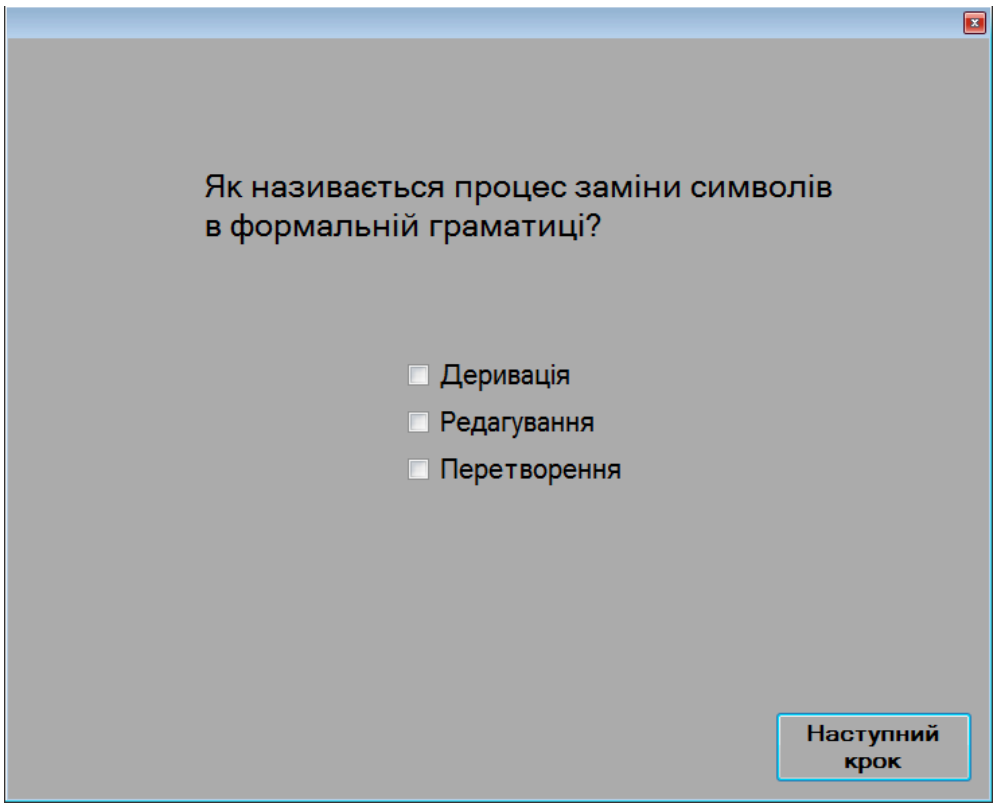


Рисунок 5.2 – вікно питання номер 7.

Далі відкривається питання номер 8 (рис. 5.3)

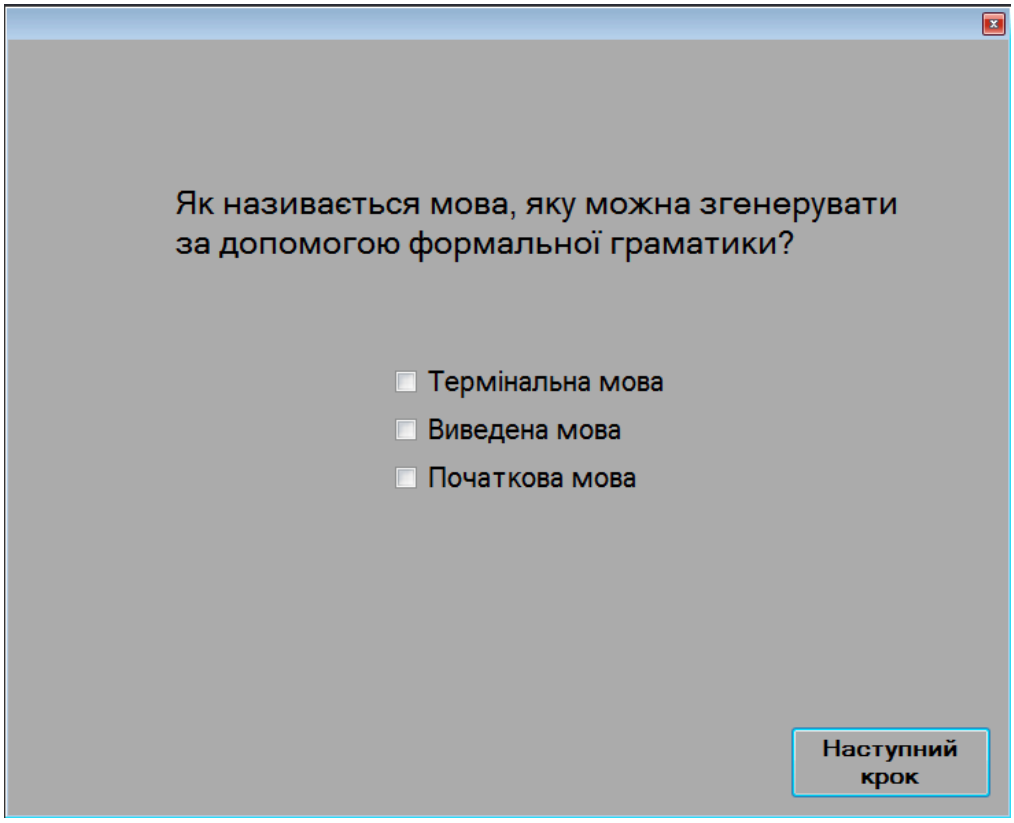


Рисунок 5.3 – вікно питання номер 8.

Аналогічно, користувач продовжує відповідати на всі питання, які задає тренажер. Після десятого питання, коли всі відповіді надані, користувач натискає кнопку "ОК", і автоматично відкривається підсумкове вікно тренажеру. Це вікно містить привітання та кнопку, яка дозволяє користувачеві завершити тренінг. (рис. 5.4)

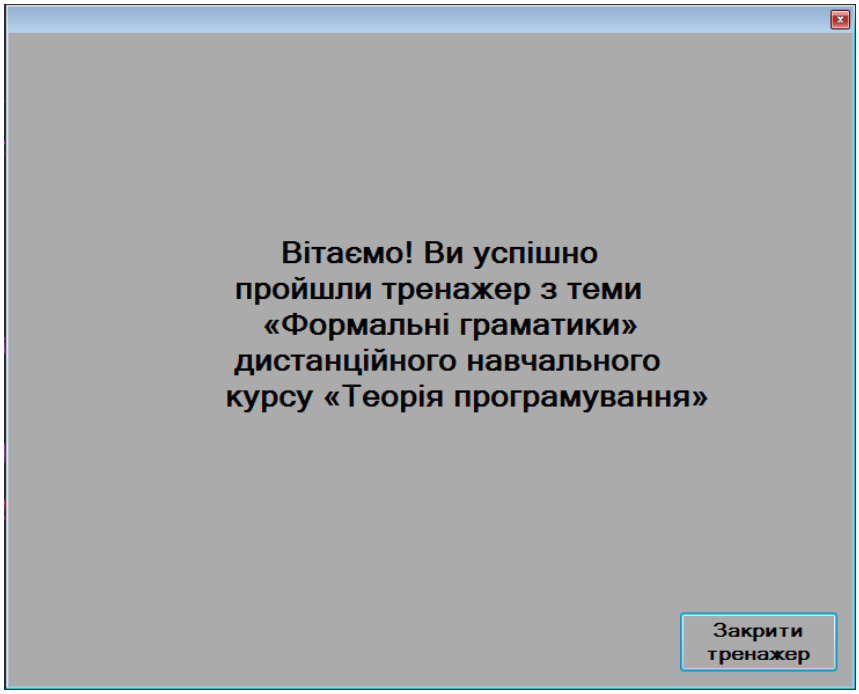


Рисунок 5.4 – останнє вікно додатку тренажеру.

4.3. Обґрунтування вибору програмних засобів

Visual Basic визначається як універсальний символічний інструкційний код для початківців. Це особлива мова, яка використовується користувачами для гнучкого середовища, у якому вони взаємодіють з комп’ютером. Це особлива мова, яка використовується користувачами для створення гнучкого середовища, у якому вони легко взаємодіють з комп’ютером, це найкраща мова програмування та найпростіша у використанні. Деякі характеристики Visual Basic включають:

1. Він простий для розуміння і може використовуватися для будь-якого програмування.
2. Вона забезпечує пряму взаємодію з користувачем комп’ютера, як жодна інша програма

3. Він зручний для користувача, оскільки надає повідомлення про помилки, які виникли
4. Він також забезпечує швидке реагування користувачам програм з мінімальними вимогами.
5. Від користувача не вимагається знання апаратного забезпечення чи операційної системи.
6. Це вимагає мінімальної пам'яті комп'ютера.
7. Програмування Visual Basic є найпростішим і найдешевшим для програмістів у використанні на будь-якому комп’ютері та доступним для тих користувачів, які спільно використовують комп’ютер. Зазвичай це тактичний продукт, хоча це система програмування четвертого покоління. зазвичай використовується в Microsoft.

ВИСНОВКИ

Отже, дистанційне навчання та впровадження програм тренажерів є важливим і потужним інструментом у сучасному освітньому середовищі. Воно забезпечує гнучкість, доступність і ефективність в навчанні. Для студентів це відкриває нові можливості для отримання знань, незалежно від місця проживання і географічних обмежень. Дистанційне навчання також сприяє самостійності, саморозвитку і розвитку навичок цифрової грамотності. Однак, воно також вимагає від студентів високої самодисципліни, самоорганізації і вміння працювати в онлайн-середовищі.

В цілому, дистанційне навчання є важливим елементом сучасної освіти, який відкриває нові можливості і розширює горизонти навчання для студентів. Використання інформаційних технологій і онлайн-ресурсів дає змогу побудувати гнучку і інтерактивну освітню систему, що відповідає потребам сучасного суспільства.

Завдання кваліфікаційної роботи було успішно виконано. Результатом є розроблений інноваційний навчальний тренажер, який забезпечує зручне та ефективне навчання студентів з теми "Формальні граматики" 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. Необхідність дистанційного навчання [Електронний ресурс]. – Режим доступу: https://life.pravda.com.ua/society/2020/07/2/241517/
3. Формальні граматики [Електронний ресурс]. – Режим доступу: https://repository.kpi.kharkov.ua/server/api/core/bitstreams/5847efdd-6ff5-4f7f-9de8-6f6555ad4cc0/content
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

```
Public Class Step1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
        If CheckBox1.Checked = True And CheckBox2.Checked = False Then
            MsgBox("Вітаю! Ваша відповідь вірна!")
            Step2.Show()
        Else
            MsgBox("Невірно! Спробуйте ще раз!")
            CheckBox1.Checked = False
            CheckBox2.Checked = False
            CheckBox3.Checked = False
        End If
    End Sub

    Private Sub Step1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        Step0.Hide()
    End Sub

    Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox1.CheckedChanged
        If CheckBox1.Checked = True Then
            CheckBox2.Enabled = False
            CheckBox3.Enabled = False
        End If
        If CheckBox1.Checked = False Then
            CheckBox2.Enabled = True
            CheckBox3.Enabled = True
        End If
    End Sub

    Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox2.CheckedChanged
        If CheckBox2.Checked = True Then
            CheckBox1.Enabled = False
            CheckBox3.Enabled = False
        End If
        If CheckBox2.Checked = False Then
            CheckBox1.Enabled = True
            CheckBox3.Enabled = True
        End If
    End Sub

    Private Sub CheckBox3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox3.CheckedChanged
        If CheckBox3.Checked = True Then
            CheckBox1.Enabled = False
            CheckBox2.Enabled = False
        End If
        If CheckBox3.Checked = False Then
            CheckBox1.Enabled = True
```

```
        CheckBox2.Enabled = True
    End If
  End Sub
End Class

Public Class Step2

  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
    If CheckBox3.Checked = True And CheckBox2.Checked = False Then
      MsgBox("Вітаю! Ваша відповідь вірна!")
      Step3.Show()
    Else
      MsgBox("Невірно! Спробуйте ще раз!")
      CheckBox1.Checked = False
      CheckBox2.Checked = False
      CheckBox3.Checked = False
    End If
  End Sub

  Private Sub Step2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
    Step1.Hide()
  End Sub

  Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox1.CheckedChanged
    If CheckBox1.Checked = True Then
      CheckBox2.Enabled = False
      CheckBox3.Enabled = False
    End If
    If CheckBox1.Checked = False Then
      CheckBox2.Enabled = True
      CheckBox3.Enabled = True
    End If
  End Sub

  Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox2.CheckedChanged
    If CheckBox2.Checked = True Then
      CheckBox1.Enabled = False
      CheckBox3.Enabled = False
    End If
    If CheckBox2.Checked = False Then
      CheckBox1.Enabled = True
      CheckBox3.Enabled = True
    End If
  End Sub

  Private Sub CheckBox3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox3.CheckedChanged
    If CheckBox3.Checked = True Then
      CheckBox1.Enabled = False
```

```
        CheckBox2.Enabled = False
    End If
    If CheckBox3.Checked = False Then
        CheckBox1.Enabled = True
        CheckBox2.Enabled = True
    End If
  End Sub
End Class

Public Class Step3

  Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
    If CheckBox3.Checked = True And CheckBox2.Checked = False Then
        MsgBox("Вітаю! Ваша відповідь вірна!")
        Step4.Show()
    Else
        MsgBox("Невірно! Спробуйте ще раз!")
        CheckBox1.Checked = False
        CheckBox2.Checked = False
        CheckBox3.Checked = False
    End If
  End Sub

  Private Sub Step2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
    Step2.Hide()
  End Sub

  Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox1.CheckedChanged
    If CheckBox1.Checked = True Then
        CheckBox2.Enabled = False
        CheckBox3.Enabled = False
    End If
    If CheckBox1.Checked = False Then
        CheckBox2.Enabled = True
        CheckBox3.Enabled = True
    End If
  End Sub

  Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox2.CheckedChanged
    If CheckBox2.Checked = True Then
        CheckBox1.Enabled = False
        CheckBox3.Enabled = False
    End If
    If CheckBox2.Checked = False Then
        CheckBox1.Enabled = True
        CheckBox3.Enabled = True
    End If
  End Sub
```

```
    Private Sub CheckBox3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox3.CheckedChanged
        If CheckBox3.Checked = True Then
            CheckBox1.Enabled = False
            CheckBox2.Enabled = False
        End If
        If CheckBox3.Checked = False Then
            CheckBox1.Enabled = True
            CheckBox2.Enabled = True
        End If
    End Sub
End Class

Public Class Step4

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
        If CheckBox3.Checked = True And CheckBox2.Checked = False Then
            MsgBox("Вітаю! Ваша відповідь вірна!")
            Step5.Show()
        Else
            MsgBox("Невірно! Спробуйте ще раз!")
            CheckBox1.Checked = False
            CheckBox2.Checked = False
            CheckBox3.Checked = False
        End If
    End Sub

    Private Sub Step2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        Step3.Hide()
    End Sub

    Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox1.CheckedChanged
        If CheckBox1.Checked = True Then
            CheckBox2.Enabled = False
            CheckBox3.Enabled = False
        End If
        If CheckBox1.Checked = False Then
            CheckBox2.Enabled = True
            CheckBox3.Enabled = True
        End If
    End Sub

    Private Sub CheckBox2_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox2.CheckedChanged
        If CheckBox2.Checked = True Then
            CheckBox1.Enabled = False
            CheckBox3.Enabled = False
        End If
        If CheckBox2.Checked = False Then
            CheckBox1.Enabled = True
```

```
            CheckBox3.Enabled = True
        End If
    End Sub

    Private Sub CheckBox3_CheckedChanged(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles CheckBox3.CheckedChanged
        If CheckBox3.Checked = True Then
            CheckBox1.Enabled = False
            CheckBox2.Enabled = False
        End If
        If CheckBox3.Checked = False Then
            CheckBox1.Enabled = True
            CheckBox2.Enabled = True
        End If
    End Sub
End Class

Public Class StepEnd

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
        End
    End Sub

    Private Sub StepEnd_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load
        Step10.Hide()
    End Sub
End Class
```