ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«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 _____ О.ОЛЬХОВСЬКА

(підпис)

«_____»_____2022 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО ДИПЛОМНОЇ РОБОТИ

на тему

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ ТРЕНАЖЕРА З ТЕМИ «НОРМАЛЬНІ АЛГОРИТМИ» АНГЛОМОВНОГО ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ АЛГОРИТМІВ»

зі спеціальності 122 «Комп’ютерні науки»

освітня програма «Комп’ютерні науки»

ступеня магістра

Виконавець роботи Пилипченко Віталій Сергійович_____ «___»_____20__р.

(підпис)

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

_____ «___»_____20__р.

(підпис)

ПОЛТАВА 2022 р.

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ

«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

ЗАТВЕРДЖУЮ

Завідувач кафедри_____О.ОЛЬХОВСЬКА

(підпис)

«____» вересня 2022 р.

ЗАВДАННЯ ТА КАЛЕНДАРНИЙ ГРАФІК

ВИКОНАННЯ ДИПЛОМНОЇ РОБОТИ

Здобувач вищої освіти зі спеціальності 122 «Комп'ютерні науки»

Освітня програма «Комп'ютерні науки»

Прізвище, ім'я, по батькові Пилипченко Віталій Сергійович

1. **Тема «Програмне забезпечення для тренажера з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів»»**

затверджена наказом ректора № ___-Н від «___» _____ 2022 р.

Термін подання студентом дипломної роботи _____ «_____»_____ 2022 р.

2.Вихідні дані до дипломної роботи _____

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

_____

4. Перелік графічного матеріалу (з точним визначенням кількості блок-схем, іншого графічного матеріалу) _____

5. Консультанти розділів дипломної роботи

Розділ	Прізвище, ініціали, посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

6. Календарний графік виконання дипломної роботи

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

Дата видачі завдання «__» _____ 20___ р.

Здобувач вищої освіти _____

(підпис)

Науковий керівник _____ _____

(підпис) (науковий ступінь, вчене звання, ініціали та прізвище)

Результати захисту дипломної роботи

Дипломна робота оцінена на _____

(балів, оцінка за національною шкалою, оцінка за ECTS)

Протокол засідання ЕК № ____ від «____» _____ 2022 р.

Секретар ЕК___ _____

(підпис) (ініціали та прізвище)

Затверджую
Зав. кафедрою _____
к.ф.-м.н. О. ОЛЬХОВСЬКА

«____» _____ 2022 р.

Погоджено
Науковий керівник _____
доцент, к.ф.-м.н. О. ЧЕРНЕНКО

«____» _____ 2021 р.

План

Дипломної роботи здобувача вищої освіти ступеня магістра
спеціальності 122 Комп'ютерні науки
освітня програма 122 Комп'ютерні науки
Прізвище, ім'я, по батькові Пилипченко Віталій Сергійович
На тему «Програмне забезпечення для тренажера з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів»»

Вступ

1. Постановка задачі
   1.1 Постановка задачі розробки тренажера
2. ІНФОРМАЦІЙНИЙ ОГЛЯД
   2.1 Огляд робіт
   2.2 Позитивні аспекти оглянутих робіт
   2.3 Вади розробок з оглянутих робіт
3. ТЕОРЕТИЧНА ЧАСТИНА
   3.1 Алгоритм роботи тренажера
   3.2 Блок-схема програми-тренажера

4 ПРАКТИЧНА ЧАСТИНА

4.1 Опис процесу програмної реалізації

Висновки

Здобувач вищої освіти _____ В. ПИЛИПЧЕНКО

«____» _____ 2022 р.

Реферат

Записка: 44с., 27 рис.,1 табл., 15 джерел.

Актуальність теорії алгоритмів обумовлюється тим, що це окремий розділ в математиці, що вивчає загальні властивості алгоритмів, та утворює теоретичний фундамент для обчислювальних наук. Алгоритми простежуються в математиці протягом всього часу її існування. Програма тренажер – вид комп'ютерної програми або онлайн-сервісів, призначених для покращення навиків та навчання студентів в оволодінні навчальним матеріалом. Дистанційне навчання в наш час користується високим попитом, через зручність у використанні, віддаленим зв'язком з викладачем, та можливістю використання в будь-який час, саме тому розробка даного тренажера є актуально.

Предмет розробки – реалізація програмного забезпечення для тренажеру з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів» для навчання розв'язку задач з використанням нормальних алгоритмів Маркова.

Мета розробки – алгоритмізація та розробка програмного забезпечення для в тренажеру англомовного дистанційного курсу «Теорія алгоритмів» з теми «Нормальні алгоритми».

Методи розробки – для реалізації елементів тренажеру було використано JavaScript, Html, Css, Scss, Bootstrap, Fullpage js. Тренажер розроблений в середовищі PhpStorm.

Ключові слова: НАВЧАЛЬНИЙ ТРЕНАЖЕР, НОРМАЛЬНІ АЛГОРИТМИ, ТЕОРІЯ АЛГОРИТМІВ, АНГЛОМОВНИЙ ДИСТАНЦІЙНИЙ КУРС.

Зміст

Перелік умовних позначень, символів, скорочень і термінів

Html	Мова гіпертекстової розмітки, засобами якої здійснюється розмітка веб-сторінок.
Css	Мова стилю веб-сторінок, що використовується для опису їх зовнішнього вигляду.
Scss	Препроцесор, який слугує для швидкого написання Css стилів
Java Script	Динамічна, об’єктно-орієнтована прототипна мова програмування
IDE	Комплексне програмне рішення для розробки програмного забезпечення.
Bootstrap	Це Html/Css фреймворк для створення сайтів.
Fullpage	Повноекранний веб-плагін.
МП	Марківська підстановка.
Плагін	Додаток, незалежно скомпільований програмний модуль, що динамічно підключається до основної програми, призначений для розширення або використання її можливостей.
Фреймворк	Інфраструктура програмних рішень, шо полегшує розробку складних систем.

Вступ

Актуальність теорії алгоритмів обумовлюється тим, що це окремий розділ в математиці, що вивчає загальні властивості алгоритмів, та утворює теоретичний фундамент для обчислювальних наук. Алгоритми простежуються в математиці протягом всього часу її існування. Програма тренажер – вид комп'ютерної програми або онлайн-сервісів, призначених для покращення уміння та навчання студентів в опановуванні навчальним матеріалом. Дистанційне навчання в наш час має високий попит, через зручність у використанні, віддаленим зв'язком з викладачем, та можливістю використання в будь-який час, саме тому розробка даного тренажера є актуально.

Мета розробки – алгоритмізація та розробка програмного забезпечення для в тренажеру англомовного дистанційного курсу «Теорія алгоритмів» з теми «Нормальні алгоритми».

Об'єкт роботи – створення програмного забезпечення дистанційного навчання з теми "Нормальні алгоритми" англомовного дистанційного курсу "Теорія алгоритмів"».

Предмет розробки – реалізація програмного забезпечення для тренажера з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів» для навчання розв'язку задач з використанням нормальних алгоритмів Маркова.

Методи розробки – для реалізації елементів тренажера було використано JavaScript, Html, Css, Scss, Bootstrap, Fullpage js. Тренажер розроблений в середовищі PhpStorm.

Обсяг пояснювальної записки: 44 стор., в т.ч. основна частина 44 стор., джерел - 15 назв.

1. ПОСТАНОВКА ЗАДАЧІ

1.1 Постановка задачі розробки тренажера

Основною задачею магістерської роботи є розробка програмного забезпечення для тренажера з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів».

Створення даної програми має відбуватися у середовищі програмування PhpStorm з використанням мови JavaScript. Алгоритм програми повинен містити теоретичний та практичний блоки. Практичний блок повинен містити задачі, які здатний розв’язати користувач, який ознайомився з блоком теорії.

Поставлена мета потребує виконання наступних завдань:

- постановка задачі,
- ознайомлення з літературою, що стосується дистанційного навчання(дистанційних курсів),
- ознайомлення з літературою, що стосується використання «Нормальних алгоритмів»,
- провести інформаційний аналіз,
- провести інформаційний огляд, ознайомлення зі схожими тренажерами,
- дослідити теоретичні відомості щодо теми,
- скласти алгоритм роботи тренажера,
- створити блок-схеми роботи складеного алгоритму,
- розробити тренажер,
- протестувати програму на наявність помилок,
- проаналізувати роботу та написати звіт.

Основні вимоги до тренажера:

- наявність блоку з теорією,
- можливість змінити мову тренажера, розробка простого та зрозумілого інтересу,
- сумісність середовища розробки з системою дистанційного навчання Moodle ПУЕТ,

- можливість зручної взаємодії з тренажером,
- створення алгоритму для перевірки відповідей, у разі помилки повідомити користувача про неї.

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Огляд подібних до теми роботи тренажерів

Огляд схожих робіт допомагає визначити основні переваги та недоліки вже створених раніше тренажерів. Для кращої взаємодії з користувачем та покращення елементів тренажеру, а також для забезпечення кращих результатів навчання студентів, було проведено огляд наявних програм розміщених в системі дистанційного навчання Moodle ПУЕТ.

Під час написання магістерської роботи було проведено огляд наявних тренажерів. Наступні тренажери написані студентами попередніх років спеціальності 122 комп'ютерні науки, а саме Жайворонка Я. І. [8], Гусака Ю. С. [9], Стельніка А. І. [10], Волосевича М. В. [11], Белінської В.В. [12].0

Жайворонок Я. І. [8] розробив програмне забезпечення для тренажера дистанційного навчально курсу «Теорія інформації та кодування» з теми «Коди і виявлення помилок» на мові C#. Тренажер починає роботу з того, що демонструє вікно з загальною інформацією про тренажер, кнопку зміни мови тренажеру, кнопку початку роботи та виходу з тренажеру. При введенні, чи виборі неправильної відповіді, програма демонструє вікно з помилкою.

Гусак Ю.С. [9] розробив тренажер з теми «Моделювання булевих функції за допомогою елементарного персептрону» дисципліни «Нейронно-мережеві технології в інформатиці» на мові C#. Тренажер починає роботу з того, що демонструє вікно з загальною інформацією про тренажер та кнопками початку роботи та виходу з тренажеру. При введенні, чи виборі неправильної відповіді, програма демонструє вікно з помилкою та підказкою.

Стельнік А. І. [10] розробив програмне забезпечення тренажеру з теми «Регулярні мови» дистанційного навчального курсу «Теорія програмування» на мові Java. Тренажер починає роботу аналогічно з тренажером [8]. При введенні, чи виборі неправильної відповіді, програма демонструє вікно з помилкою та пояснення при повторній помилці.

Волосевич М. В. [11] розробив тренажер з теми «Марківські підстановки» дистанційного навчального курсу «Теорія алгоритмів» та розробка його

програмного забезпечення на мові Java. Тренажер починає роботу аналогічно з тренажером [8]. При введенні, чи виборі неправильної відповіді, програма демонструє вікно з помилкою [8-10].

Белінська В. В. [12] розробила програмне забезпечення з теми «Розподіли дискретних випадкових величин та їх числові характеристики» дистанційного навчального курсу «Теорія ймовірностей та математична статистика» на мові C#. Тренажер починає роботу аналогічно з тренажером [8]. При введенні, чи виборі неправильної відповіді, програма демонструє вікно з помилкою [8-11].

Кривошей О. С.[13] розробив тренажер на тему «Оптимізація перевезень сільгосппродукції: програмна реалізація тренажера (моделювання, розв’язування) дистанційного курсу «Проектне навчання з курсу «Методи оптимізації та дослідження операцій»»» на мові JavaScript, з використанням JavaScript бібліотеки «Sweet Alert 2» та JavaScript фреймворків «Electron» та «jQuery». Програма починає роботу з того, що користувач отримує загальну інформацію про тренажер, а також умову задачі.. Під час проходження тесту, програма-тренажер перевіряє правильність кожної відповіді користувача. Якщо користувач ввів не правильну відповідь, то він отримує вікно з помилкою[8-12].

2.2 Позитивні аспекти оглянутих робіт

1. Можливість переходу до попереднього питання [8,13].
2. Багатомовність тренажеру [8,9,10,12]
3. Виконання тренажеру у вигляді десктопної програми, це дозволяє користуватися програмою без наявності інтернету [8-13].
4. В тренажерах [8-12] постійно відображається умова задачі.
5. Можливість роботи з тренажером за допомогою мобільного з операційною системою Android [8].
6. Можливість взаємодії з тренажером шляхом введення відповіді з клавіатури [8-13].
7. Тренажери [8-13] допомагають здобути знання з теми, для якої вони розроблені.

8. В процесі розробки тренажеру [9, 13] було використано фреймворк, а не тільки мова програмування.
9. Тренажер [8] містить змістовні підказки.
10. В тренажері [8, 10-12] теоретичний та практичний матеріал розділений за темами.

2.3 Вади розробок з оглянутих робіт

В тренажерах [11,13] немає підказки і в тому випадку, коли користувач не знає правильної відповіді, то він змушений перебирати всі можливі варіанти.

В тренажерах [11, 13] немає вибору мови.

ТЕОРЕТИЧНА ЧАСТИНА

3.1 Алгоритм роботи тренажера

Крок 0.

Програма починає роботу зі стартового вікна, в якому демонструє меню вибору мови тренажера. Після обрання мови програма демонструє вікно з інформацію про тему, виконавця та керівника дипломного проекту.

Крок 1.

Тренажер демонструє вікно теоретичного блоку, який містить визначення, формули та приклади розв’язування завдань з практичного блоку.

Крок 2.

При виборі теоретичного блоку програма демонструє вікно в якому користувач отримує теоретичну інформацію, яка розподілена по блоках. Перший блок з теорією:

Нормальні алгоритми являють собою деякі правила по перетворенню слів в деякому алфавіті, так що вхідні дані та результат алгоритмів є словами в деякому алфавіті. Теорія нормальних алгоритмів була створена радянським математиком А. А. Марковим в кінці 40-55 рр. XX ст. Він висловив думку про можливість запису будь-яких алгоритмів в термінах слів і словникових функцій, які одним словам ставлять у відповідність інші слова деякого алфавіту.

Алгоритм – це сукупність правил, які визначають процес перероблювання допустимих початкових даних у вихідні результати.

Марков розглядав початкові дані та вихідні результати у вигляді слів, які складаються із символів деякого скінченого алфавіту.

Алфавіт – будь-яка скінчена сукупність символів, а букви алфавіту – символи цієї сукупності.

Слово в алфавіті – це кожна впорядкована сукупність букв даного алфавіту (ab, ba – різні слова).

Довжина слова – це число букв у слові.

Слово P = xy має довжину 2.

Слово, що не містить жодної букви, називається пустим, його довжина 0.

Виконання алгоритму починається з перевірки першої продукції. Якщо вона може бути застосована до рядка, то рядок перетворюється. Якщо ж формула не може бути застосована (тобто в рядку не знайдено підрядка, який можна було б замінити), то відбувається перехід до перевірки наступної продукції. У випадку успішного розпізнавання входження та заміни виконання алгоритму припиняється, якщо підстановка була заключною, або продовжується з першої продукції. Тобто після заміни, визначеної звичайною підстановкою, завжди відбувається повернення до початку нормального алгоритму і знову пошук входження першої підстановки у змінене слово.

Алгоритм Маркова завершується в одному з двох випадків: - до рядка не може бути застосована жодна з наявних формул підстановок; - до рядка застосована заключна (термінальна) підстановка. У кожному з цих випадків уважають, що нормальний т алгоритм застосовний до заданого вхідного слова. Якщо ж у процесі виконання нормального алгоритму безмежну кількість разів застосовувані не завершальні формули, то алгоритм незастосовний до заданого вхідного слова.

Крок 3.

Програма демонструє другий блок з теорією:

Ідею нормального алгоритму Маркова (НАМ) розглянемо на прикладі таблиці слів, що перетворюються за допомогою Марківських підстановок.

Для зручності міркувань допускаються порожні слова (в їх складі немає жодної букви). Порожнє слово будемо позначати Λ (читається як “лямба”).Якщо А і В – два алфавіту, причому $A \leq B$, то алфавіт В називається розширенням алфавіту А.

Одне слово може бути складовою частиною іншого слова. Тоді перше слово називається підсловом другого, або входженням в друге.

Крок 4.

Тренажер демонструє третій блок з теорією:

Означення 1. Марківською підстановкою (МП) називається операція над словами, що задається за допомогою впорядкованої пари слів (PP,QQ), суть якої в наступному: в заданому слові RR знаходять перше входження слова PP (якщо таке є) і, не змінюючи інших частин слова RR, замінюють в якому це входження словом

QQ. Отримане слово називають результатом використання МП (PP,QQ) до слова RR. Якщо ж першого входження PP в слово RR немає (і відповідно, PP в RR), то вважається що МП (PP,QQ) незастосована до слова RR.

Окремими випадками Марківських підстановок є підстановки з порожніми словами (ΛΛ,QQ),(PP, ΛΛ),(ΛΛ, ΛΛ) див. таблицю 1.1.

Крок 5.

Тренажер демонструє теоретичний блок з означеннями та прикладами розв'язання завдань у вигляді акордеон меню.

Приклад 1

Хай АА – український алфавіт

а) Розглянемо такі слова

P1P1: машина, P2P2: шина, P3P3: на.

Слово P2P2 є підсловом P1P1, P3P3 є підсловом P1P1 та P2P2

б)

P1P1: параграф, P2P2: граф, P3P3: ра.

P3P3 є підсловом P1P1 і P2P2, причому в P1P1 воно входить двічі.

Особливу цікавість представляє перше входження.

Означення 2. Формулою підстановки (PP,QQ) називається Марківська підстановка, що позначається P→Q.P→Q. Слово PP називається лівою частиною в формулі підстановки, а QQ – правою частиною.

Означення 3. Якщо підстановка записана у вигляді P→.QP→.Q, то вона називається формулою остаточної (завершальної) підстановки.

Зміст назви розшифруємо пізніше.

Означення 4.Упорядкований скінчений список підстановок

P1→(.)Q1P1→(.)Q1

P2→(.)Q2P2→(.)Q2

Pr→(.)QrPr→(.)Qr

в алфавіті АА називається схемою (записом) нормального алгоритму в алфавіті АА. Крапка в дужках може стояти в цьому місці, або бути відсутня. Дана схема визначає алгоритми перетворення слів, що називаються нормальним алгоритмом Маркова.

Таблиця 3.1 – Приклад Марківської підстановки.

	Перетворюване слово	Марківська підстановка	Результат
1	245	4,00	2005
2	функція	функція, tg x	tg x
3	телефон	теле	фон
4	екран	теле (цю підстановку не можна застосувати до слова екран)	Результату немає

Приклад 2.

Нехай над словами з алфавіту V = {a,Ь,c.} задано алгоритм з такими формулами підстановки P1: ab => 'b', P2: ac => c P3 : aa => a. Цей алгоритм вилучає всі входження символа 'a' з рядка, за винятком випадку, коли 'a' є в кінці рядка.

Простежимо роботу алгоритму, якщо вхідний рядок має вигляд 'bacaabaa'. Нехай символ => означає результат перетворення, а підрядок, який підлягає заміні, будемо виділяти жирним шрифтом: 'baca**ab**aa' => 'bac**ab**aa' ==> 'b**ac**baa' => 'bcb**aa**' 'bcba'.

Оскільки далі жодна з формул не може бути застосована, то на цьому робота алгоритму завершується.

Нормальні алгоритми іноді зображають за допомогою граф-схеми. Алгоритми, які задають граф-схемами, складеними винятково з розпізнавачів входження і операторів підстановки, називають нормальними, якщо їхні граф-схеми задовольняють такі умови: 1) кожний розпізнавальний вузол Q і відповідний йому операторний вузол Q –> R об'єднані в один узагальнений вузол, який називають операторно-розпізнавальним; усі узагальнені вузли схеми впорядковані за допомогою нумерації від 1 до n; 2) негативний вихід і~го вузла приєднаний до (i+1)-го вузла (i = 1,n - 1), а негативний вихід п-го вузла — до вихідного вузла граф-схеми; 3) позитивні виходи всіх узагальнених вузлів приєднані або до першого, або до вихідного вузла граф-схеми; у першому випадку підстановку оператора

відповідного вузла називають звичайною, а в другому — заключною; 4) вхідний вузол приєднаний до першого узагальненого вузла.

Приклад 2.1. Алгоритм, зображений на граф-схемі (рис.3.1), є нормальним і виконує перетворення довільного вхідного слова Р в алфавіті А = {x,y}, що містить т значень х та п значень у, у вихідне слово довжини |m - n|, що складається тільки з х (якщо m> n), або тільки з у (якщо n> n). Нагадаємо, що Λ позначено порожнє слово.

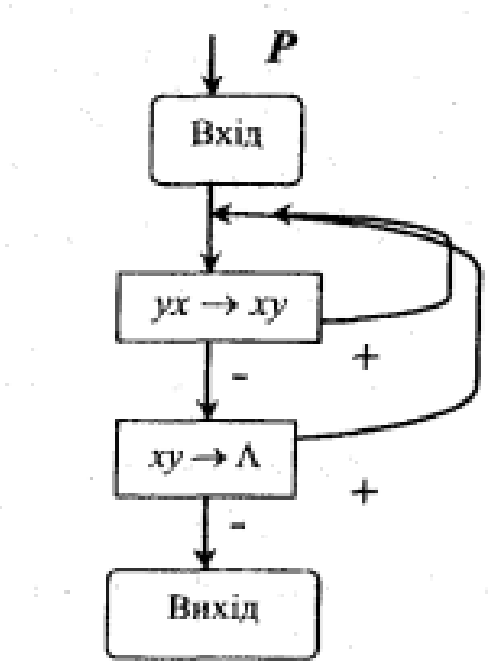

Рис. 3.1.–Граф-схема

Нормальні алгоритми прийнято задавати не граф-схемами, а впорядкованими наборами підстановок, у яких кожна підстановка відповідає операторно-розпізнавальному вузлу. У цьому разі звичайні підстановки записують у вигляді Q —>R, а заключні - Q-* 'R Упорядкований набір підстановок визначеного типу називають схемою заданого алгоритму. Порядок виконання підстановок однозначно визначений умовами 1 - 4 означення нормального алгоритму.

Справді, з огляду на ці умови довільна і-та підстановка схеми алгоритму повинна виконуватися годі й тільки тоді, коли вона є першою із застосовних підстановок, тобто всі підстановки від першої до (і-1)- ї не можуть бути застосовані. Виконання алгоритму починається з першої формули. Послідовність виконання алгоритму залежить від того, чи може бути застосована до рядка чергова формула

підстановки. Процес виконання підстановок закінчується лише тоді, коли жодна з підстановок схеми не може бути застосована до отриманого слова, або коли виконана (перший раз) деяка заключна підстановка.

Крок 6.

Після того, як користувач ознайомився або вивчив теоретичний блок, чи одразу вибрав практичний блок, програма демонструє вікно з першим питанням тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки а) до слова abcddacba.

1. dccddacba
2. abcdddcba
3. aaddacba
4. abcdcba

Якщо користувач вибрав неправильну відповідь, то програма продемонструє вікно з помилкою, в іншому випадку тренажер продемонструє вікно з наступним питанням.

Правильна відповідь (варіант 1).

Крок 7.

Тренажер демонструє друге питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки ж) до слова abcddacba.

1. dccddacba
2. abcdddcba
3. aaddacba
4. abcdda

Правильна відповідь (варіант 4).

Крок 8.

Тренажер демонструє третє питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки в) до слова abcddacba.

1. dccddacba
2. abcbbacba
3. aaddacba
4. abcdda

Правильна відповідь (варіант 2).

Крок 9.

Тренажер демонструє четверте питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки е) до слова abcddacba.

1. dccddacba
2. abcbbacba
3. ddacba
4. abcdda

Правильна відповідь (варіант 3).

Крок 10.

Тренажер демонструє вікно з п’ятим питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки и) до слова abcddacba.

1. dccddacba
2. abcbbacba
3. ddacba
4. abcdacdba

Правильна відповідь (варіант 4).

Крок 11.

Тренажер демонструє вікно з шостим питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки л) до слова acdbaacd на першому входженні.

1. dcdbaacd
2. abcbbacba
3. bdcdbbdbdcd
4. abcdacdba

Правильна відповідь (варіант 1).

Крок 12.

Тренажер демонструє вікно з сьомим питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки л) до слова acdbaacd на першому входженні.

1. dcdbaacd
2. abcbbacba
3. bdcdbbdbdcd
4. abcdacdba

Правильна відповідь (варіант 1).

Крок 13.

Тренажер демонструє вікно з восьмим питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який результат підстановки л) до слова acdbaacd на другому входженні.

1. abcbbacba
2. dcdbadcd
3. bdcdbbdbdcd
4. abcdacdba

Правильна відповідь (варіант 2).

Крок 14.

Тренажер демонструє вікно з дев'ятим питання тестового типу.

Хай для слів в алфавіті A = {a, b, c, d} задані наведені Марківські підстановки: а) ab → dc ; г) ac → dc ; ж) cba → Λ ; к) b → a ; б) bc → a ; д) cb → d ; з) da → Λ ; л) a → bd . 18 в) dd → bb ; е) abc → Λ ; и) dac → acd ;

Який фінальний результат підстановки л) до слова acdbaacd.

4. dccddacba
5. abcbbacba
6. bdcdbbdbdcd
4. abcdacdba

Правильна відповідь (варіант 3).

В наступних кроках користувач повинен ввести відповідь на питання у відповідне поле. Якщо користувач ввів у відповідне поле неправильну відповідь, то програма продемонструє вікно з помилкою, в іншому випадку продемонструє наступне питання.

Крок 15.

Програма демонструє вікно з питанням та полем для вводу відповіді.

Нехай нормальний алгоритм заданий такими формулами підстановок: 1) ху => ух ; 2) ух =>Λ. Тоді на вхідному слові Р - ххуухух одержимо ххуухух => хухухух =>

уххухух => ухуххух => уууххух => уухххух => уухухххх => ууухххх => уухххх => ухх => х. Тобто Q = х і цей алгоритм реалізує віднімання від кількості х кількість у.

Введіть результат підстановки 1 до слова ххуухух на першому входженні.

Правильна відповідь (хухухух).

Крок 16.

Тренажер демонструє вікно з питанням та полем для вводу відповіді.

1) ху => ух ; 2) ух =>Λ.

Введіть результат підстановки 1 до слова ххуухух на другому входженні.

Правильна відповідь (ухххухух).

Крок 17.

Тренажер демонструє вікно з питанням та полем для вводу відповіді.

1) ху => ух ; 2) ух =>Λ.

Введіть результат підстановки 1 до слова ххуухух на третьому входженні.

Правильна відповідь (ухухххух).

Крок 18.

На екрані відкривається вікно з питанням та полем для вводу відповіді.

1) ху => ух ; 2) ух =>Λ.

Введіть результат роботи алгоритму до слова ххуухух.

Правильна відповідь (х).

Крок 19.

Тренажер демонструє вікно з питанням та полем для вводу відповіді

Нормальний алгоритм в алфавіті A = {a, b, 1} задається схемою: a $\rightarrow$1, b $\rightarrow$1 . Застосуйте його до слова: bababbaa, та введіть результат у відповідне поле.

Рішення: даний алгоритм спочатку послідовно замінює всі входження букви a в даному слові на букву 1, і потім також послідовно замінює всі входження букви b на букву 1. Наявні в слові букви 1 залишає без зміни. Таким чином, дане слово переробляється в слово, що складається з такої кількості одиниць, скільки всього букв містило дане слово.

Процес перероблення даного слова виглядає так: abaabbb => 1baabbb =>1b1abbb => 1b11bbb => 1111bbb => 11111bb 111111b => 1111111.

До отриманого слова жодна з підстановок даної схеми вже не застосовна. Отже, робота алгоритму завершена.

Відповідь (1111111).

Крок 20.

На екрані відкривається вікно з питанням та полем для вводу відповіді.

Нормальний алгоритм в алфавіті A = {a, b} задається схемою: ba → ab, ab → Λ . Застосуєте його до слова: a) bbaabab. Виявіть закономірність в роботі алгоритму, та введіть результат роботи алгоритму у відповідне поле.

Рішення: спочатку можливе число разів працює перша підстановка схеми: bbaabab => bababab => abbabab => ababbab => aabbbab => aabbabb => aababbb => aaabbbb. Перша підстановка себе вичерпала і до отриманого слова більш не застосовна. Але тепер працює друга підстановка: aaabbbb => aabbb => abb => b.

Відповідь (b).

Крок 21.

Програма демонструє вікно з питанням та полем для вводу відповіді.

Нормальний алгоритм в алфавіті A = {a, b, 1} задається схемою: a →1,b →1, 11→ Λ Застосуєте його до слова abaabbb. Рішення: алгоритм попереднього завдання є складовою частиною алгоритму справжнього завдання. Тому спочатку дане слово перероблюється в слово, що складається з такої кількості одиниць, скільки всього букв містить дане слово.

Потім в отриманому слові з одиниць починають викреслювати одиниці по дві штуки. У результаті або не залишиться жодної одиниці (пусте слово, якщо початкове слово містило парне число букв), або залишається одна одиниця.

Наприклад, для даного слова процес перероблення виглядає так (з урахуванням роботи алгоритму з попереднього завдання): abaabbb => 1111111 => 11111 => 111 => 1.

До отриманого слова жодна з підстановок даної схеми вже не застосовна. Отже, робота алгоритму завершена.

Правильна відповідь (1)

Крок 22.

Програма демонструє вікно з питанням та полем для вводу відповіді.

Нормальний алгоритм в алфавіті A = {a, b, 1} задається схемою: a →1,b →1, 11→ Λ Застосуєте його до слова bbbabaaab. Рішення: алгоритм попереднього завдання є складовою частиною алгоритму справжнього завдання. Тому спочатку дане слово перероблюється в слово, що складається з такої кількості одиниць, скільки всього букв містить дане слово.

Потім в отриманому слові з одиниць починають викреслювати одиниці по дві штуки. У результаті або не залишиться жодної одиниці (пусте слово, якщо початкове слово містило парне число букв), або залишається одна одиниця.

Наприклад, для даного слова процес перероблення виглядає так (з урахуванням роботи алгоритму з попереднього завдання): bbbabaaab => 111111111 => 1111111 => 11111 => 111 => 1.

До отриманого слова жодна з підстановок даної схеми вже не застосовна. Отже, робота алгоритму завершена.

Правильна відповідь (1)

Крок 23.

Програма демонструє вікно з питанням та полем для вводу відповіді.

Нормальний алгоритм в алфавіті A = {f, c, l,} задається схемою: f →cl, c → Λ, l→ 1, 11→ Λ Застосуєте його до слова ffclc. Рішення: алгоритм поступово замінює всі літери f на cl. Потім в отриманому слові всі літери c перетворюються в порожнє слово.

Потім в отриманому слові всі літери l заміняються на 1, після чого починають викреслювати одиниці по дві штуки. У результаті або не залишиться жодної одиниці (пусте слово, якщо початкове слово містило парне число букв), або залишається одна одиниця.

Наприклад, для даного слова процес перероблення виглядає так (з урахуванням роботи алгоритму з попереднього завдання): ffclc => clclclc => lll => 111 => 1.

До отриманого слова жодна з підстановок даної схеми вже не застосовна. Отже, робота алгоритму завершена.

Правильна відповідь (1)

Крок 24.

Після того, як користувач ввів правильну відповідь на останнє запитання тренажер демонструє фінальне вікно програми.

3.2 Блок-схема програми тренажера

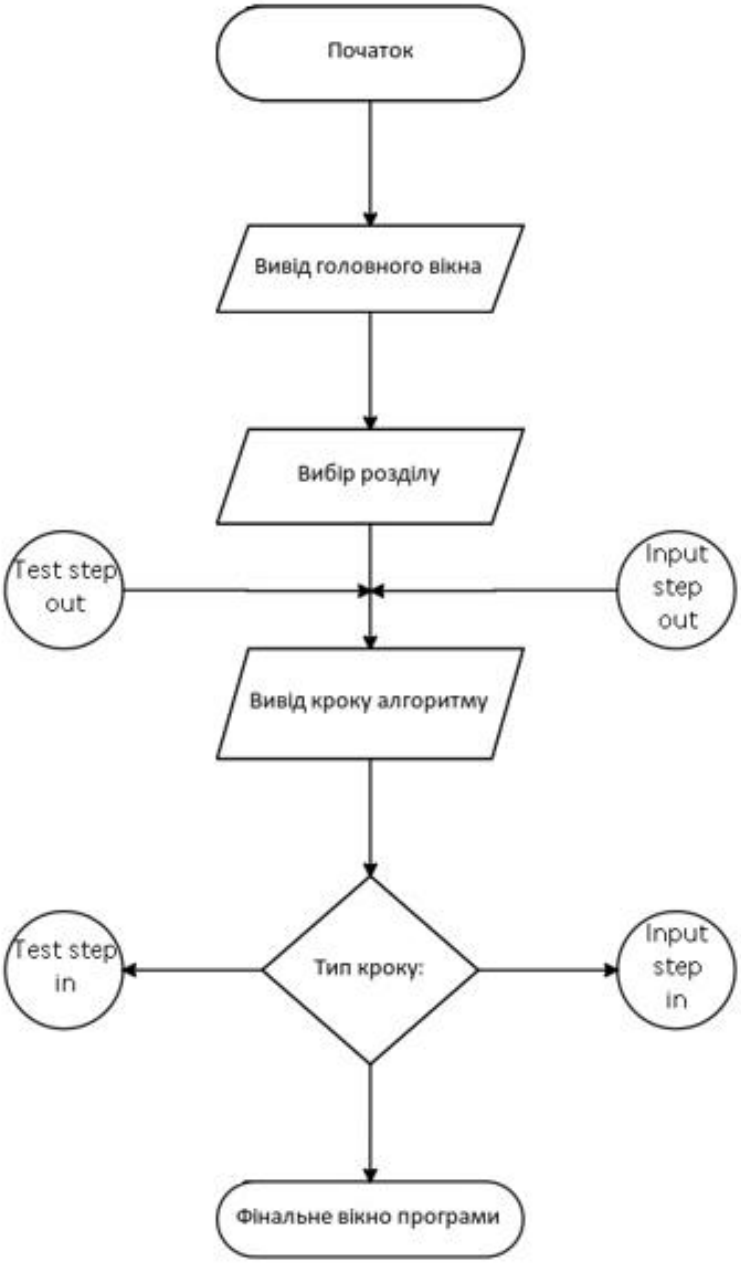

Рисунок 3.2 Блок-схема загальної роботи тренажеру

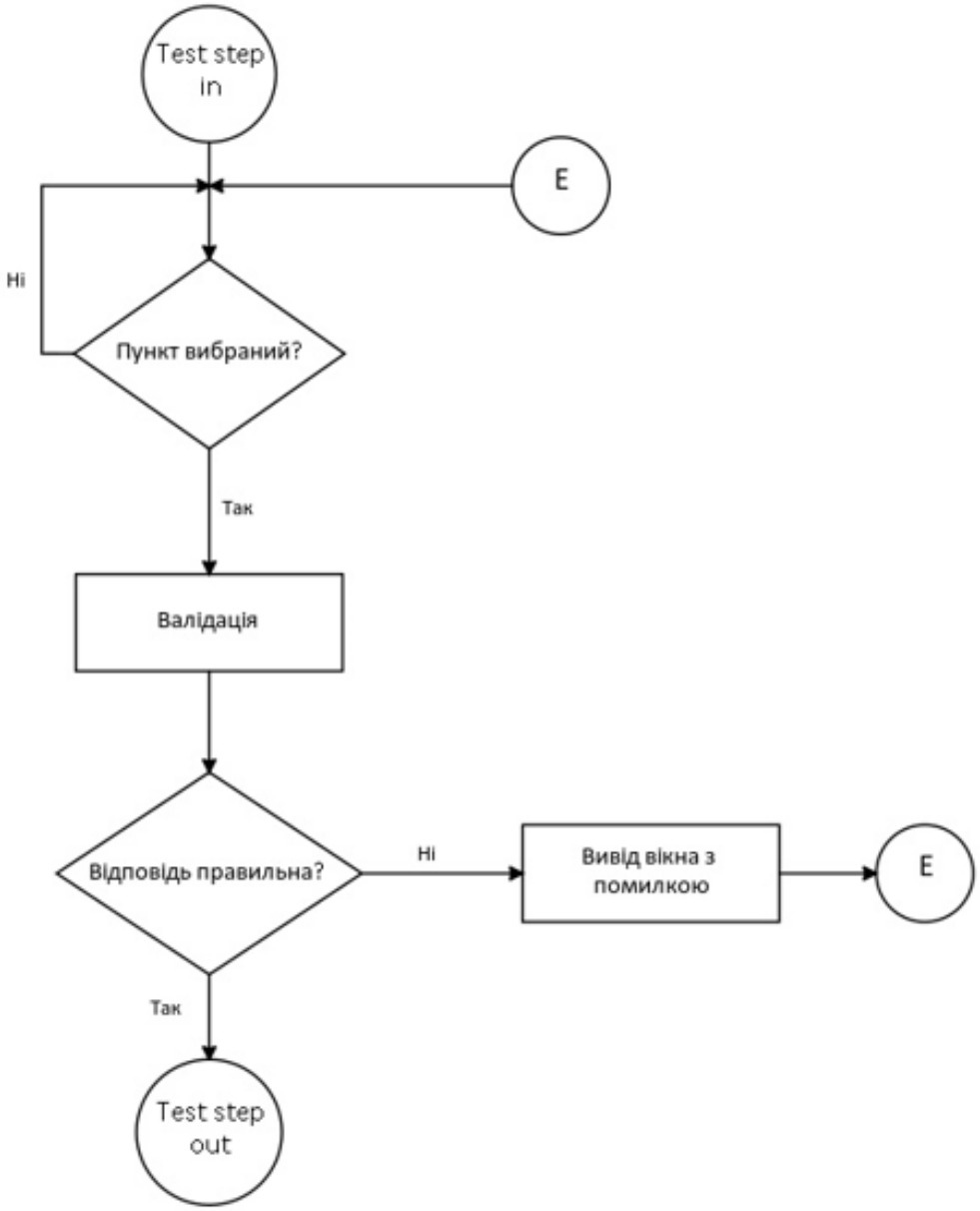

Рисунок 3.3 Блок-схема роботи тренажеру для тестових кроків

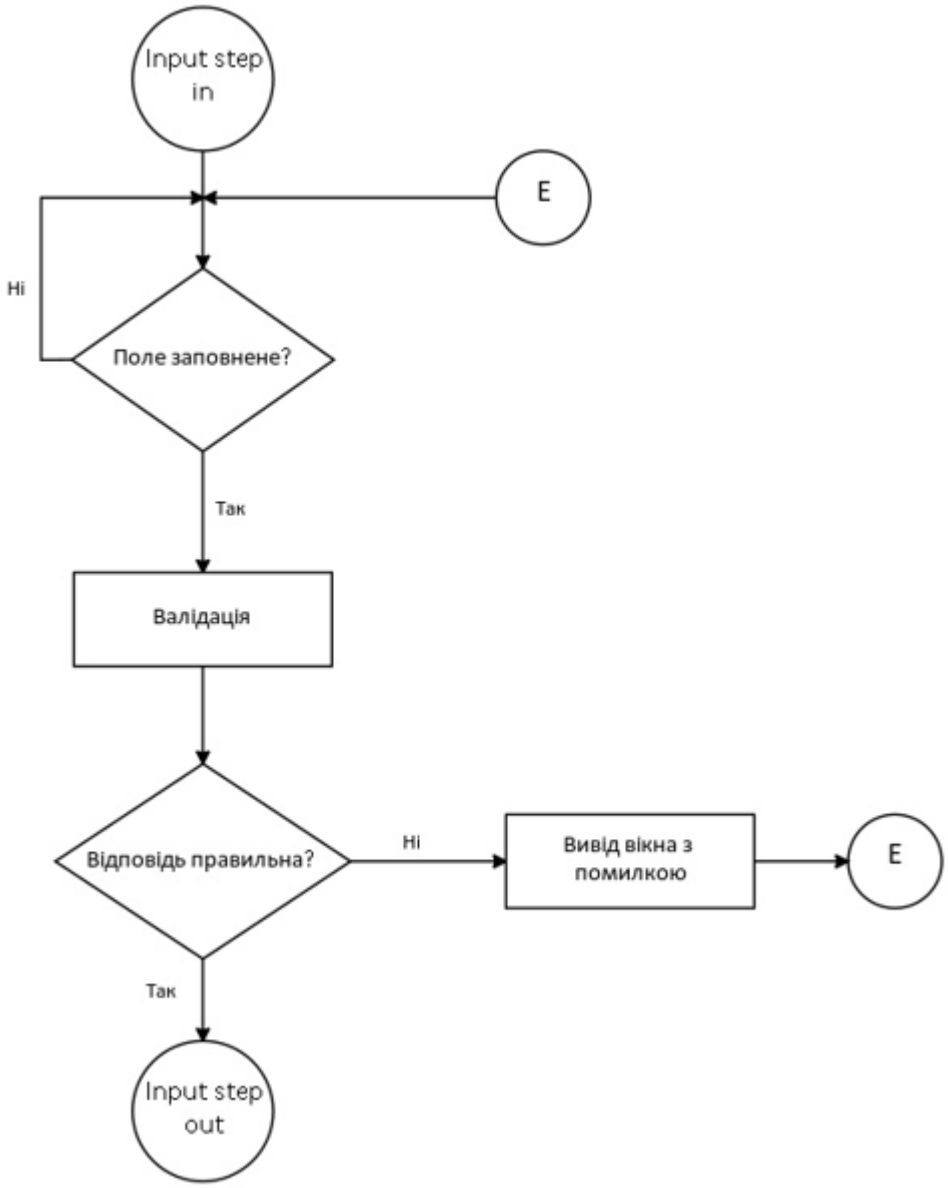

Рисунок 3.4 Блок-схема роботи тренажеру для кроків введення відповіді користувачем

ПРАКТИЧНА ЧАСТИНА

4.1. Опис процесу програмної реалізації.

Для створення тренажера використано :PhpStorm, Html, Css, Scss, JS, Bootstrap, Fullpage. Для цього було створено проект у середовищі розробки (IDE) PhpStorm. PhpStorm обрано в якості редактору коду тому, що цей редактор має такі переваги:

- глибоко аналізує структуру коду,
- підтримує автодоповнення коду,
- попереджує про помилки.

В PhpStorm можна працювати з такими сучасними технологіями:

- HTML 5,
- Css,
- Sass,
- Less,
- Stylus,
- CoffeScript,
- Emmet,
- JavaScript.

При цьому також доступні ре факторинг, відладка та юніт-тестування. Завдяки функції Live Edit всі зміни які були внесені в програму можливо відразу переглянути в браузері. Використовуючи PhpStorm розробник отримує інструмент для форматування та швидкого виправлення коду, а також можливість писати акуратний код, який легко підтримувати.

Мова програмування JavaScript у 21 столітті почала виступати в якості однієї з найпопулярніших. Це мова, яка у своїй основі містить скрипт.

Працювати зі скриптами може кожен браузер, що в свою чергу робить цю мову кросплатформенною. JavaScript – це мультипарадигменна мова. Вона підтримує декілька стилів одночасно:

- функціональна,
- об’єктно-орієнтована,

- імперативна.

JavaScript має досить широке поширення. Використовуючи JS можна створювати різноманітні програмні продукти. JS активно бере участь в:

- фронтенд-розробці,
- створенні мобільного ПЗ,
- бекенд-програмуванні,
- написанні ПЗ десктопного типу.

До переваг JS можна віднести наступні моменти:

- незамінність в веб-розробці. JS – це основна технологія для клієнт-серверних моделей,
- наявність повної інтеграції з верстkою сторінок та серверною частиною,
- раціональність використання та простота,
- швидкість роботи. JS може зробити фінальний продукт більш швидким та зручним,
- продуктивність,
- комфортність використання користувацьких інтерфейсів,
- наявність особистої потужної екосистеми. Особливо помітно в останні роки. Приклад – поява великої кількості корисних фреймворків, котрі підходять для будь-якого випадку.

Незважаючи на значні переваги у використанні, JS має також і свої недоліки:

- відсутність можливості читання і завантаження документів,
- відсутність віддаленого доступу. Повноцінно для мережевого ПЗ JS не використовується,
- нестрога типізація,
- вільна трактовка типів даних.

В підсумку можна сказати, що в 2022 році у JavaScript нема достойних конкурентів. Це означає, що мова поки що єдина в своєму роді. JS – розробники користуються попитом як у великих компаніях так і у невеликих фірмах. JS швидко вивчається, та без всяких проблем справляється з усіма

задачами які поставлені перед розробником. В найближчі роки це 100% лідер веб-розробки та програмування для інтернету.

Після цього було створено початкову розмітку, завантажено та підключено до проекту фреймворк Bootstrap та плагін Fullpage js (рис.4.1).

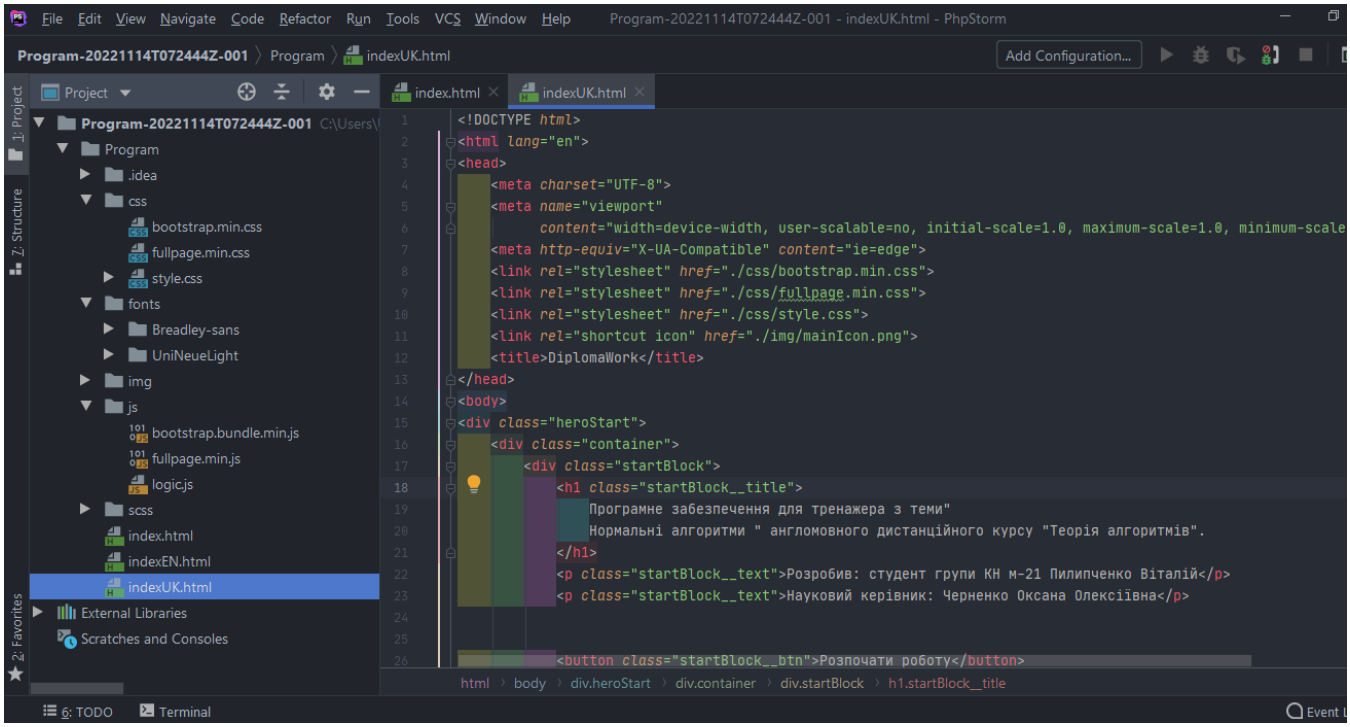

Рисунок 4.1 – Створення проекту

Наступним кроком було створення розмітки меню, для вибору мови тренажера (рис.4.2).

Наступним кроком було створення розмітки блоків стартового вікна програми (рис.4.3), меню, теоретичного блоку, практичного блоку, блоку помилки, фінального блоку.

```
</head>

<body>
<div class="selectLang">
    <a class="selectLang__link selectLang__link--En" href="indexEN.html">EN</a>
    <a class="selectLang__link selectLang__link--Uk" href="indexUK.html" >UK</a>
</div>

<script src="./js/bootstrap.bundle.min.js"></script>
<script src="./js/fullpage.min.js"></script>
<script src="js/logic.js"></script>
<script> new fullpage('#fullpage', {
//options here
```

Рисунок 4.2 – Розмітка меню для вибору мови

```
    <link rel="shortcut icon" href="./img/mainIcon.png">
    <title>DiplomaWork</title>
</head>
<body>
<div class="heroStart">
    <div class="container">
        <div class="startBlock">
            <h1 class="startBlock__title">
                Програмне забезпечення для тренажера з теми"
                Нормальні алгоритми " англомовного дистанційного курсу "Теорія алгоритмів".
            </h1>
            <p class="startBlock__text">Розробив: студент групи КН м-21 Пилипченко Віталій</p>
            <p class="startBlock__text">Науковий керівник: Черненко Оксана Олексіївна</p>


            <button class="startBlock__btn">Розпочати роботу</button>
        </div>
    </div>
</div>
```

Рисунок 4.3 – Розмітка стартового вікна тренажеру

Для функціонування програми вся логіка описана у файлі «logic.js» (рис.4.4). На рисунку 4.5. зображена функція для перевірки тестових питань.

Коли користувач натискає кнопку відповісти, то на кнопці спрацьовує подія «click», яка в свою чергу викликає функцію «checkRadio». У тому випадку, коли користувач не вибрав варіант відповіді, кнопка являється неактивною (рис.4.6).

Перевірка питань, в яких користувач вводить варіант відповіді у відповідне поле, перевірка введених даних проходить наступним чином. Коли користувач ввів відповідь у відповідне поле, при натисканні на кнопку відповісти, спрацьовує подія

«click», яка в свою чергу викликає функцію «checkExcel» (рис.4.7).

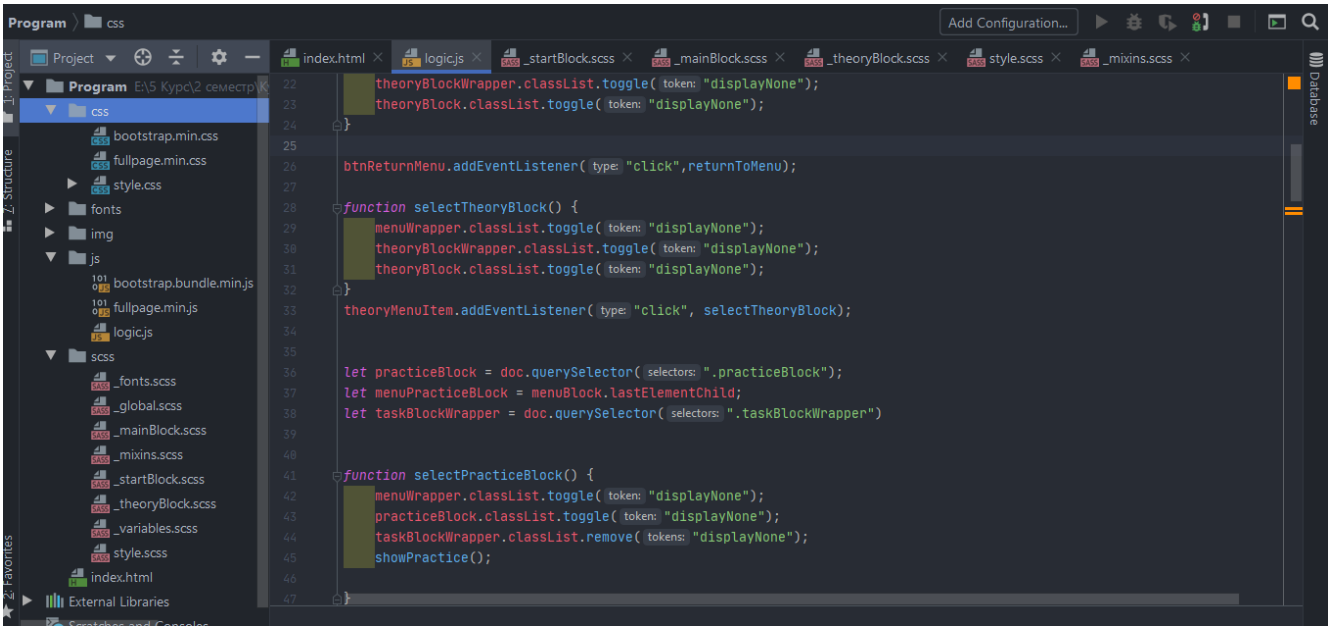

Рисунок 4.4 – Логіка тренажеру

```
function checkRadio() {
    for (let i = 0; i < radioArr.length; i++) {
        if (radioArr[i].value === "1" && radioArr[i].checked === true && counter < 4) {
            closeActivePractice();
            counter++;
            showPractice();
            nextForm();
            newRadio();
        } else if (radioArr[i].value === "1" && radioArr[i].checked === true && counter === 4) {
            closeActivePractice();
            counter++;
            showPractice();
        } else if (radioArr[i].value !== "1" && radioArr[i].checked === true) {
            showError();
        }
    }
}
```

Рисунок 4.5 – Функція для перевірки питання з вибором варіанту відповіді

```
//Перевірка тестовых питань
btn1.addEventListener( type: "click", checkRadio);
btn2.addEventListener( type: "click", checkRadio);
btn3.addEventListener( type: "click", checkRadio);
btn4.addEventListener( type: "click", checkRadio);
btn5.addEventListener( type: "click", checkRadio);
```

Рисунок 4.6 – Виклик функції перевірки тестових питань

```
function checkExcel() {
    if (inputExcel[excelCounter].value === excelArrTrue[excelCounter]) {
        closeActivePractice();
        counter++;
        excelCounter++;
        showPractice();
    } else {
        showError();
    }
}
btn6.addEventListener( type: "click", checkExcel);
btn7.addEventListener( type: "click", checkExcel);
btn8.addEventListener( type: "click", checkExcel);
btn9.addEventListener( type: "click", checkExcel);
```

Рисунок 4.7 – Функція для перевірки питань введених користувачем

Для стилізації програми використано файли стилів Scss, в яких описані стилі для всіх блоків програми (рис.4.8).

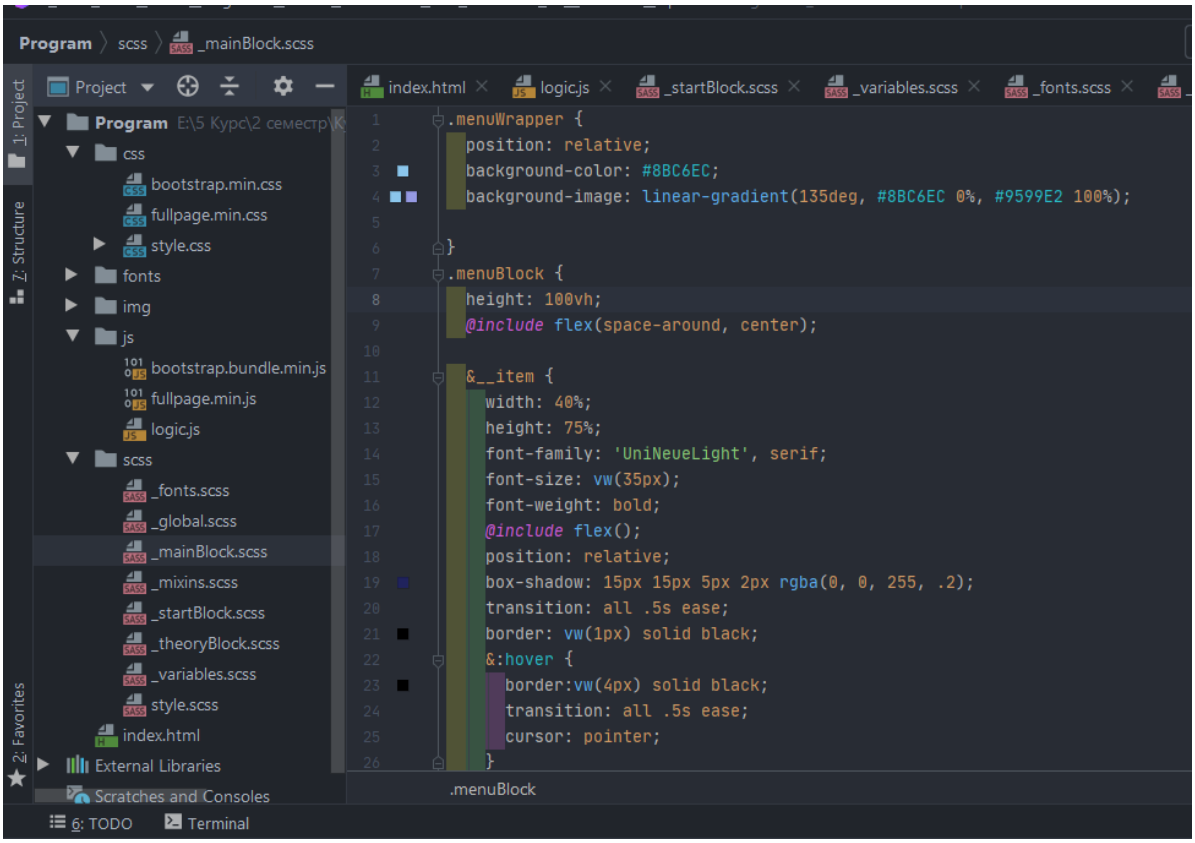

Рисунок 4.8 – Файли стилів

Після відкриття програми користувач побачить стартове меню програми (рис.4.9), в якому пропонується обрати мову тренажера.

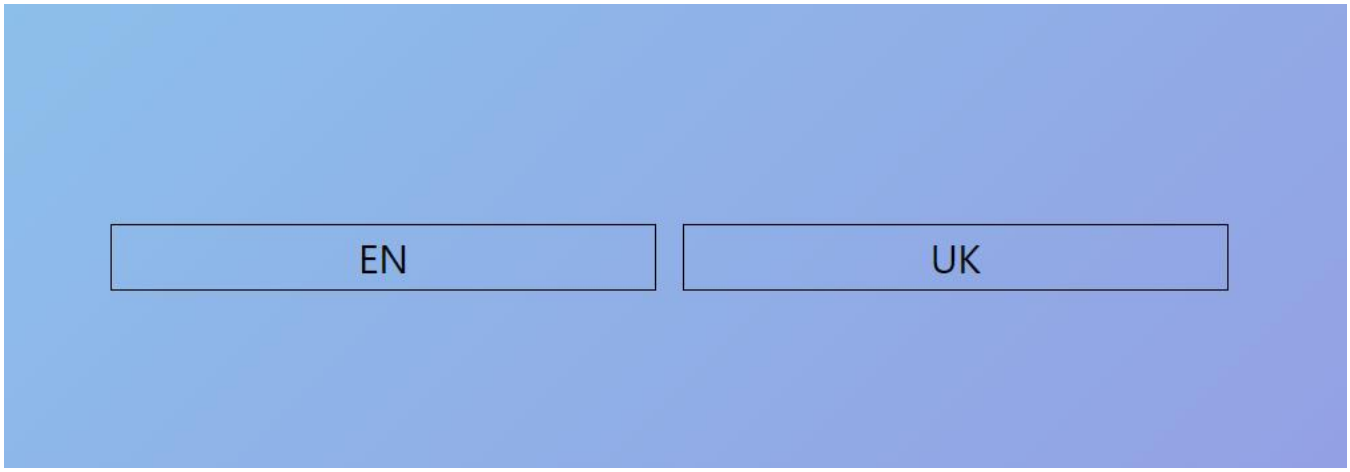

Рисунок 4.9 – Стартове меню

Рисунок 4.10 демонструє вікно в якому подається інформація про тренажер, розробника та наукового керівника.

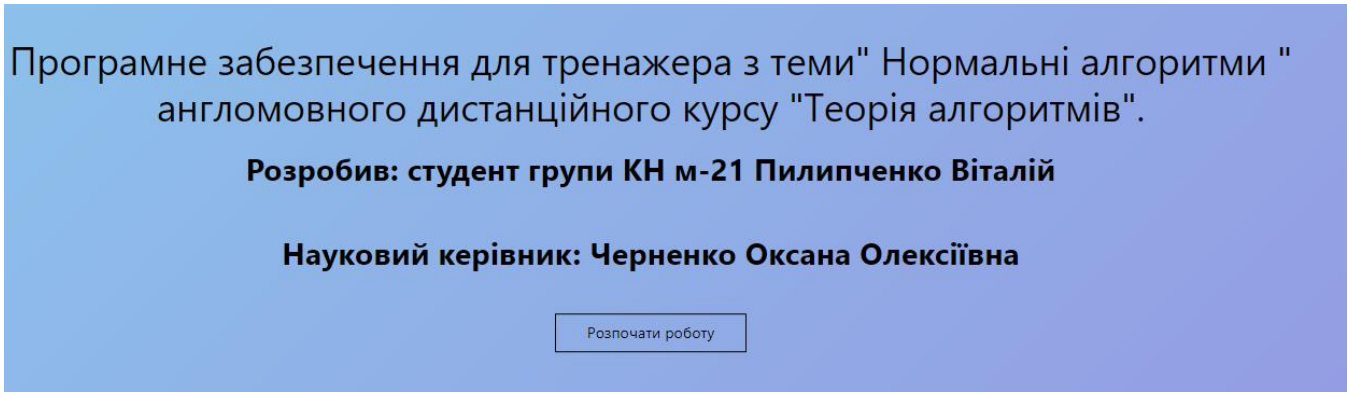

Рисунок 4.10 – Стартове вікно

Рисунок 4.11 демонструє стартове вікно англійською.

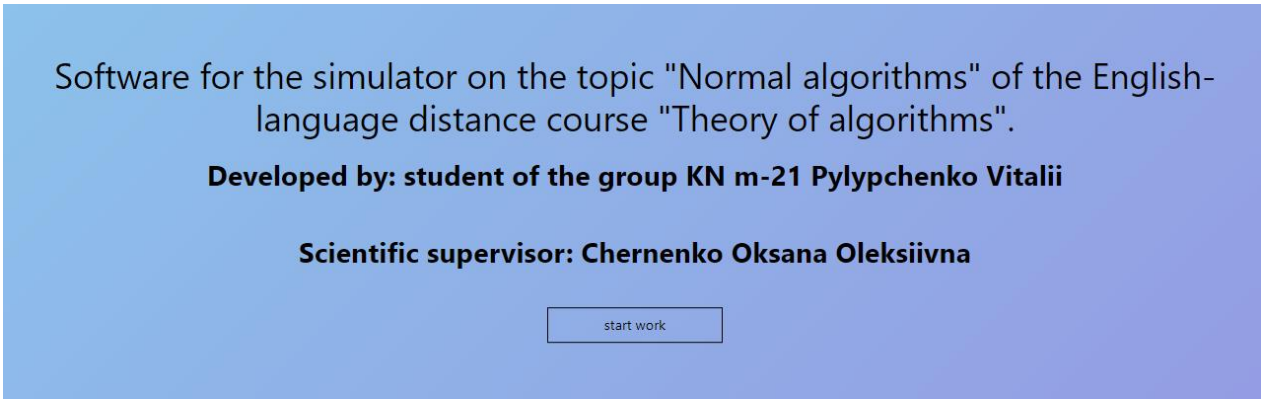

Рисунок 4.11 – Стартове вікно англійською

Натиснувши кнопку «Розпочати роботу» програма демонструє вікно меню

(рис.4.12), в якому можна обрати теоретичний, або практичний блок програми.

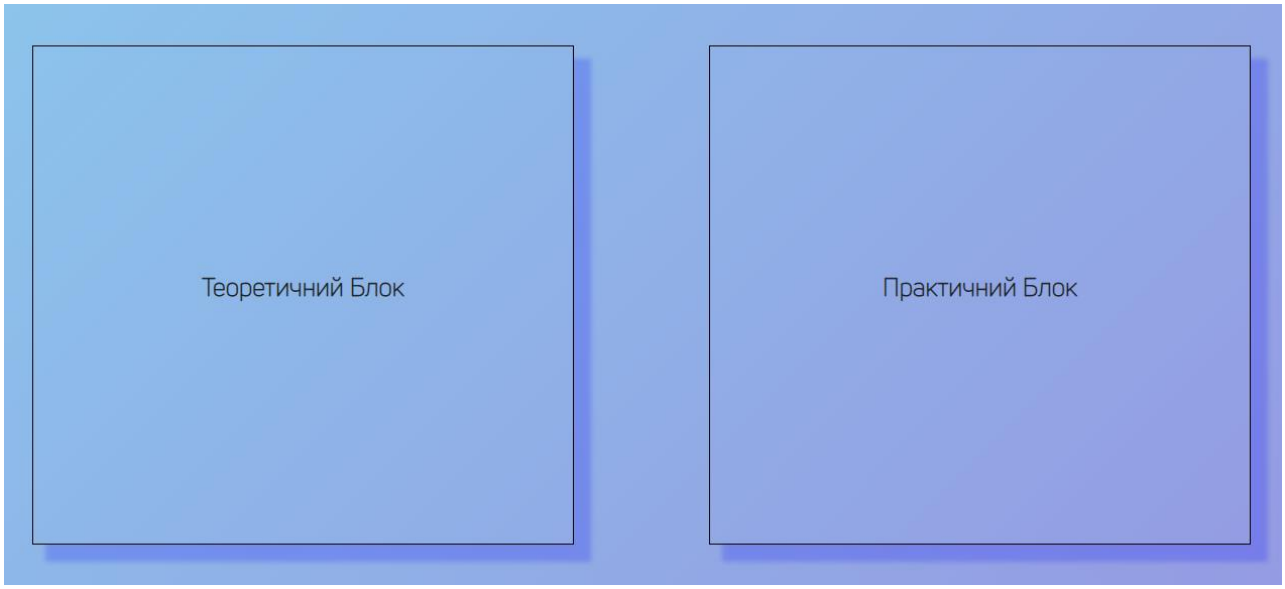

Рисунок 4.12 – Головне меню

Натиснувши на теоретичний блок, програма демонструє вікно з теорією (рис. 4.13 – 4.15). Використовуючи скрол можна прокрутити теоретичні блоки.

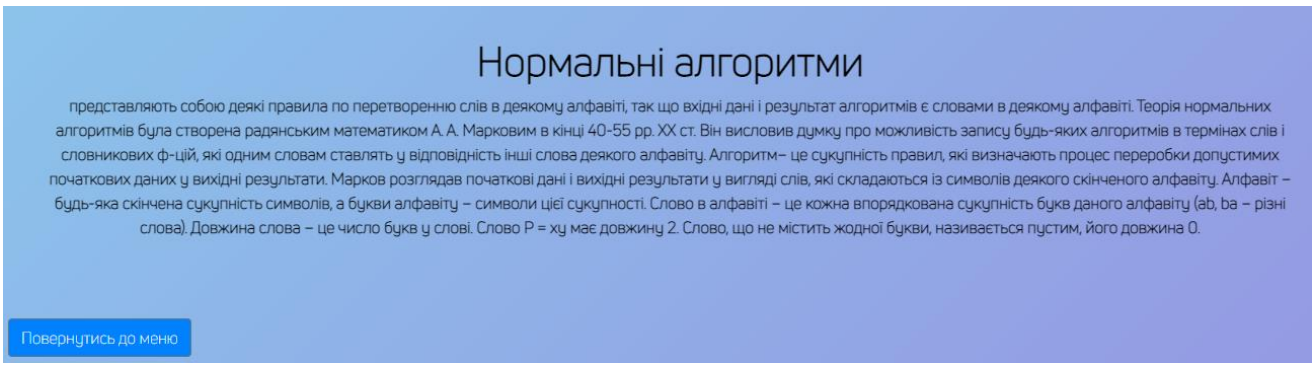

Рисунок 4.13 – Вікно з теорією

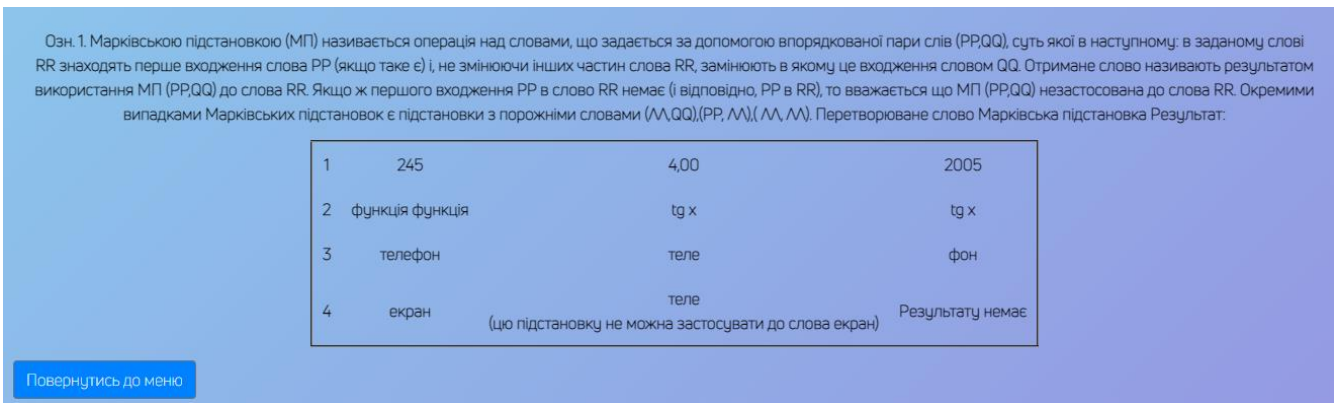

Рисунок 4.14 – Вікно з теорією

На рисунку 4.16 зображено теоретичний блок англійською.

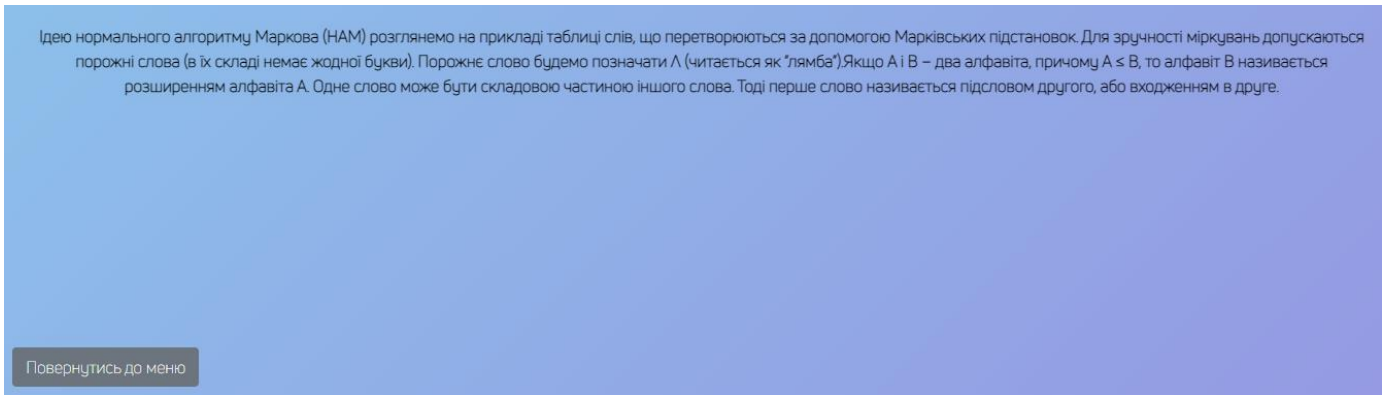

Рисунок 4.15 – Вікно з теорією

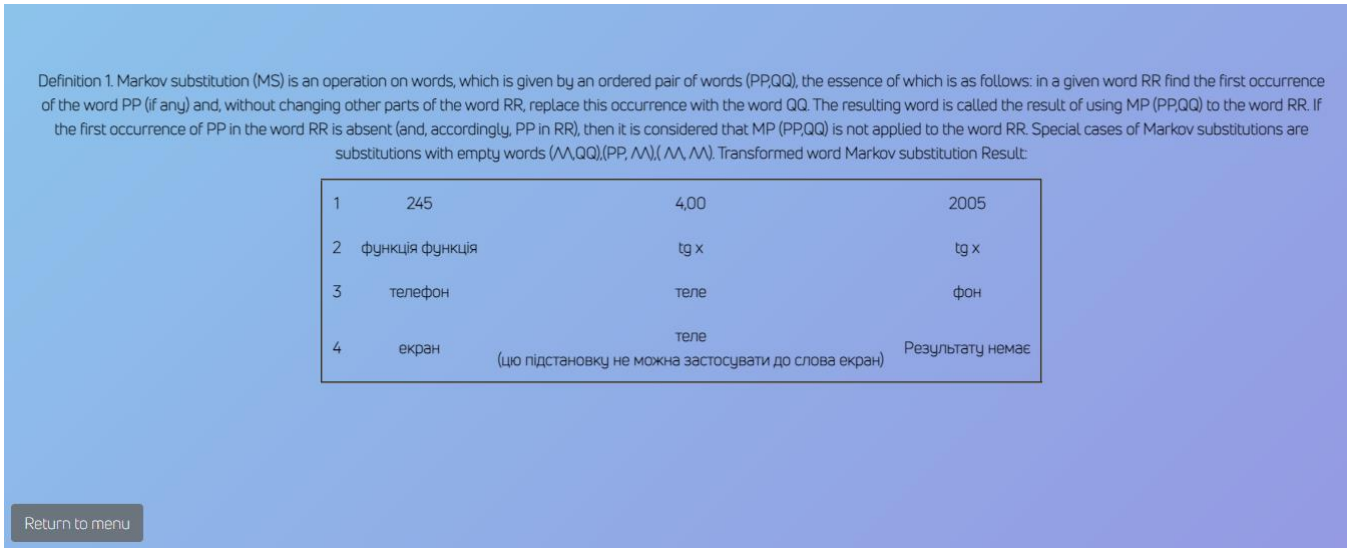

Рисунок 4.16 – Вікно з теорією англійською

Рисунок 4.17 демонструє теоретичний блок з означеннями та прикладами розв'язання завдань у вигляді акордеон меню.

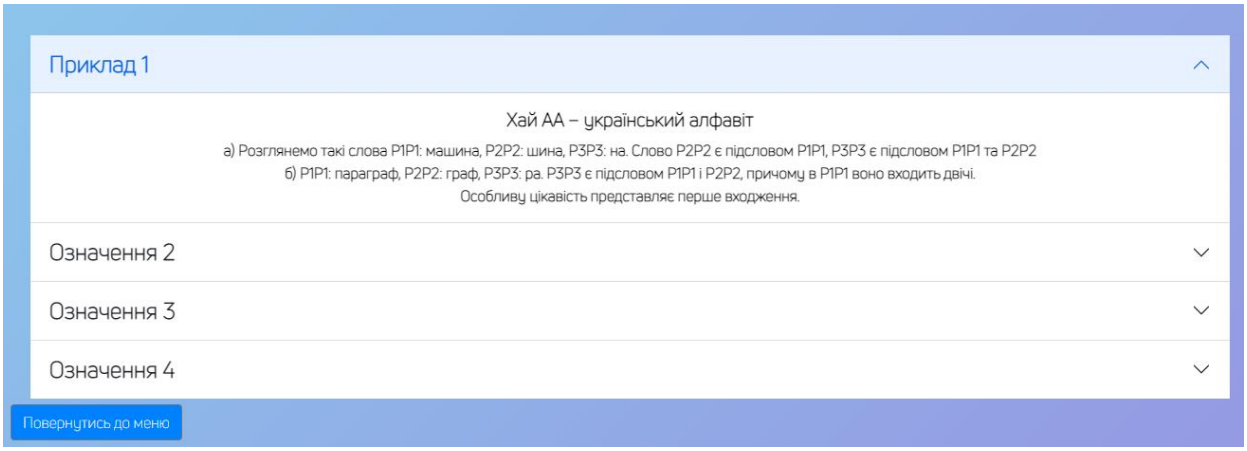

Рисунок 4.17 – Акордеон меню

Натиснувши на кнопку «Повернутись до меню» в нижньому лівому куті блоку з практикою, користувач потрапляє в головне меню програми.

Якщо користувач обирає практичний блок, то програма демонструє вікно з першим практичним завданням тестового типу (рис. 4.18).

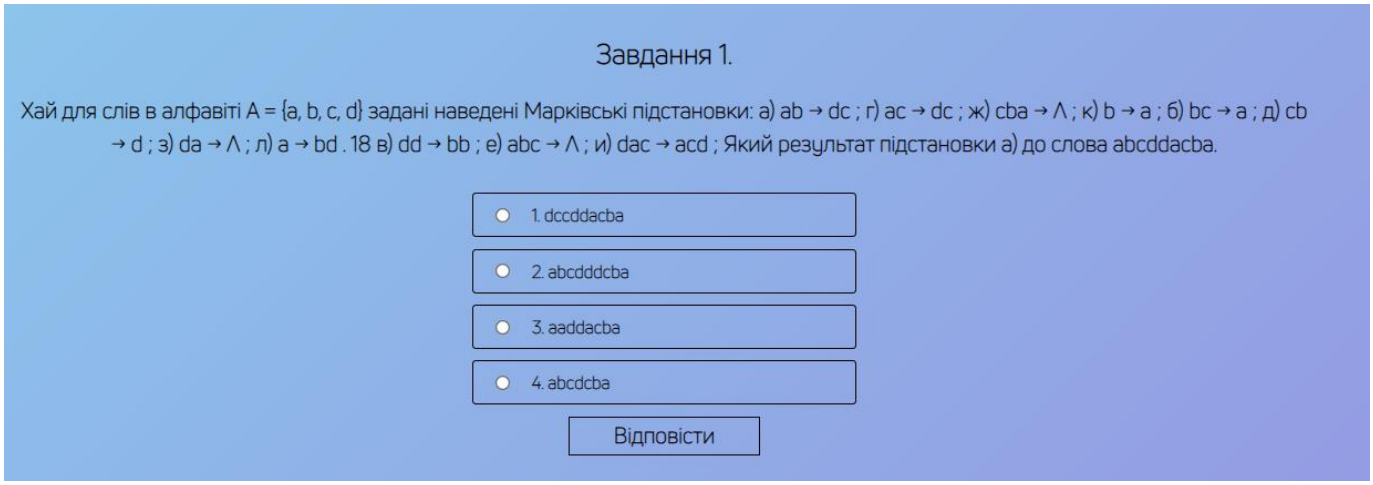

Рисунок 4.18 – Завдання тестового типу

У разі вибору правильного варіанту відповіді, користувач отримує наступне завдання, в іншому випадку програма демонструє вікно з помилкою (рис.4.19).

Рисунок 4.19 – Вікно з помилкою

На рисунку 4.20 зображено вікно з помилкою англійською.

На рисунку 4.21 демонструється питання в яких необхідно ввести правильний варіант відповіді у відповідне поле.

На рисунку 4.22 демонструється питання в яких необхідно ввести правильний варіант відповіді у відповідне поле на англійській.

При введенні неправильної відповіді вікно з помилкою аналогічне до питань тестового типу.

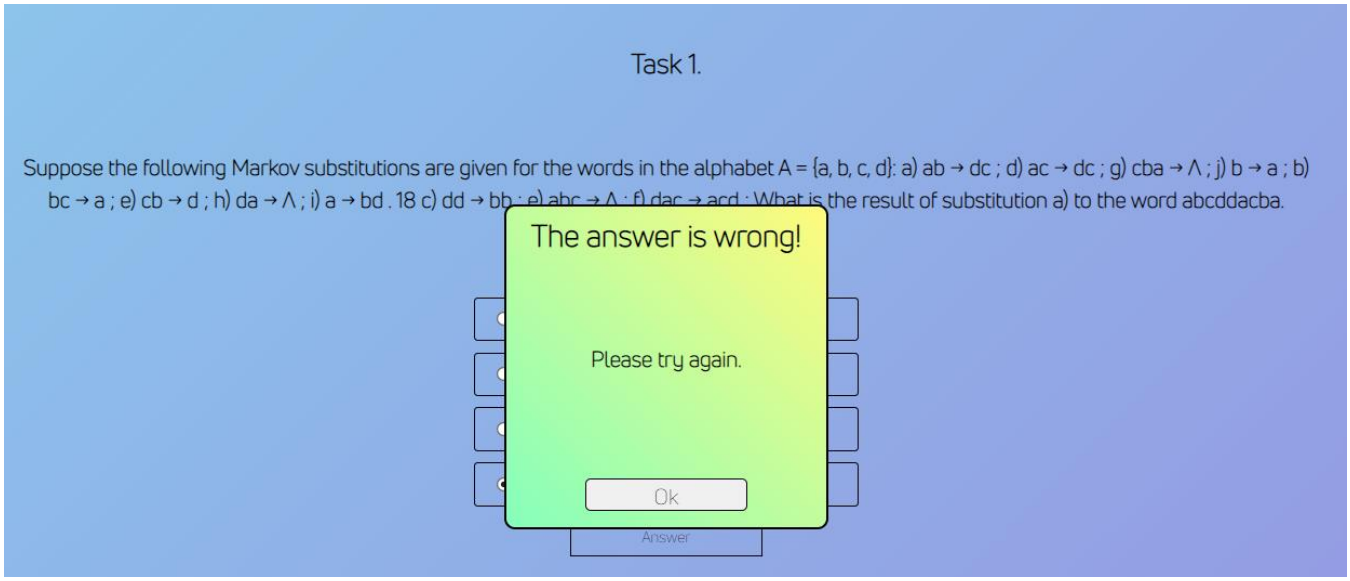

Рисунок 4.20 – Вікно з помилкою англійською

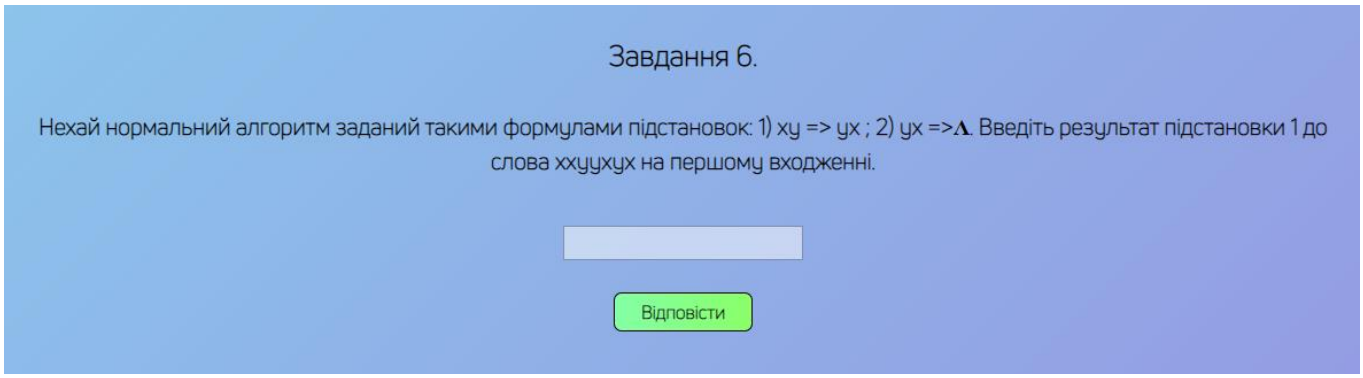

Рисунок 4.21 – Питання з введенням відповіді

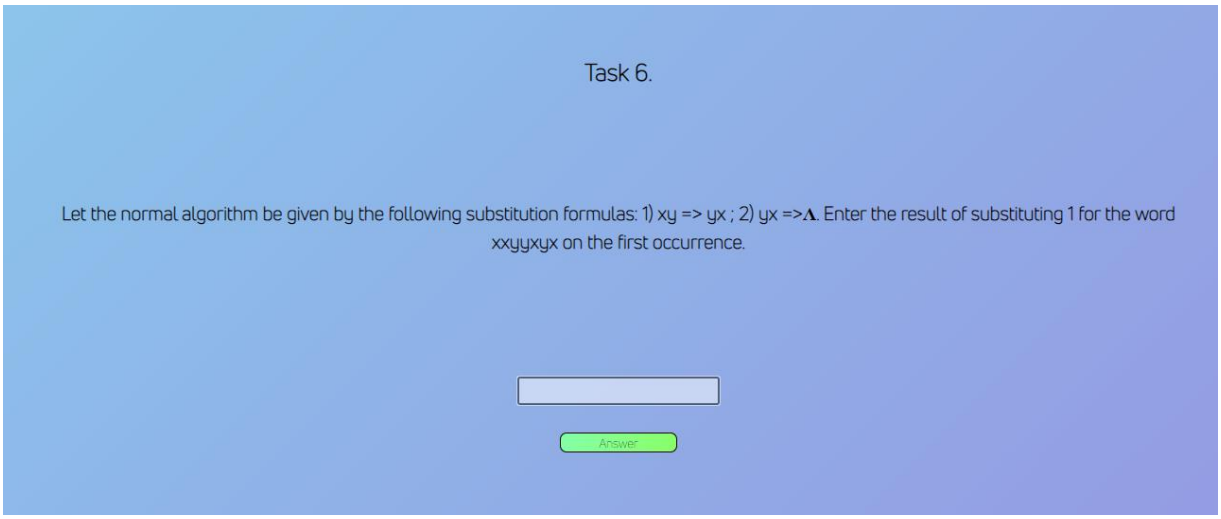

Рисунок 4.22 – Питання з введенням відповіді на англійською

На рисунку 4.23 демонструється фінальне вікно програми.

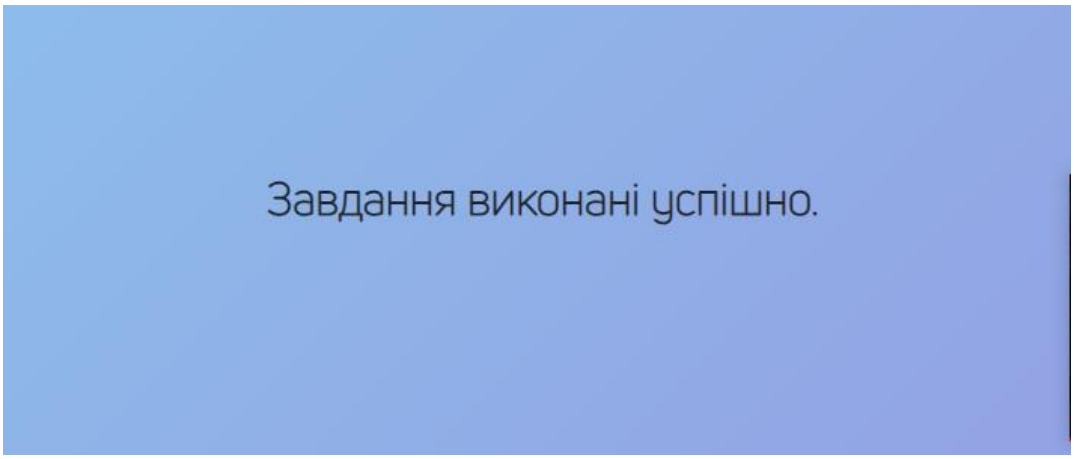

Рисунок 4.23 – Фінальне вікно програми

ВИСНОВКИ

Реалізовано програмне забезпечення для тренажера з теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів»

Під час виконання дипломної роботи були виконані наступні завдання:

- сформульовано постановку задачі,
- проаналізована література, що стосується дистанційного навчання(дистанційних курсів),
- проаналізована література, що стосується використання «Нормальних алгоритмів»,
- проведено інформаційний аналіз,
- проведений інформаційний огляд, ознайомлення зі схожими тренажерами,
- досліджені теоретичні відомості щодо теми,
- складений алгоритм роботи тренажера,
- створено блок-схеми роботи складеного алгоритму,
- розроблений тренажер у обраному середовищі,
- програма протестована на наявність помилок,

Основні вимоги до тренажера виконані:

- наявність блоку з теорією,
- можливість змінити мову тренажера, розробка простого та зрозумілого інтересу,
- сумісність середовища розробки з системою дистанційного навчання Moodle ПУЕТ,
- можливість зручної взаємодії з тренажером,
- створення алгоритму для перевірки відповідей, у разі помилки повідомити користувача про неї.

Даний тренажер впроваджено для вивчення теми «Нормальні алгоритми» англомовного дистанційного навчального курсу «Теорія алгоритмів» в Полтавському університеті економіки та торгівлі.

Результати дипломної роботи опубліковані у Збірнику наукових статей магістрів Вищого навчального закладу Укоопспілки «Полтавський університет економіки і торгівлі»

СПИСОК ЛІТЕРАТУРИ

1. Ємець О.О. Методичні рекомендації до виконання дипломної роботи / О.О. Ємець. – Полтава: Кафедра ММСІ ПУЕТ, 2018. Режим доступу: http://www2.el.puet.edu.ua/st/pluginfile.php/142962/mod_resource/content/0/Методичні%20рекомендації%20до%20виконання%20дипломної%20роботи.pdf
2. Агарева О.Ю. Математическая логика и теория алгоритмов : учеб. пособие / О.Ю. Агарева, Ю.В. Селиванов. – М.: МАТИ, 2011. – с. 88-92.
3. Клакович Л.М. Теорія алгоритмів: Навч. Посібник / Л. М. Клакович, С. М. Левицька, О. В. Костів – Львів : ВЦ ЛНУ ім. І. Франка, 2008. – с. 37. Режим доступу: http://194.44.152.155/elib/local/sk751718.pdf
4. Bootstrap документація – 2021. Режим доступу: https://blog.getbootstrap.com
5. Современный учебник JavaScript – 2021. Режим доступу: https://learn.javascript.ru
6. JavaScript учебные материалы – 2021. Режим доступу: https://developer.mozilla.org/ru/docs/Web/JavaScript
7. Основы Sass – 2021. Режим доступу: https://sass-scss.ru/guide/
8. Fullpage документація - 2021. Режим доступу: https://sass-scss.ru/guide/
9. Жайворонок Я. І. Пояснювальна записка до дипломної роботи на тему «Розробка програмного забезпечення для тренажера дистанційного навчального курсу «Теорія інформації та кодування» з теми «Коди із виявленням помилок» / Я. І. Жайворонок // Полтава: Кафедра ММСІ ПУЕТ, 2020. Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10045 с. 4-14, с. 37-61 .
10. Гусак Ю. С. Пояснювальна записка до дипломної роботи на тему Розробка тренажеру з теми «Моделювання булевих функцій за допомогою елементарного персептрону» дисципліни «Нейронно-мережеві технології в інформатиці» / Ю. С. Гусак // Полтава: Кафедра ММСІ ПУЕТ, 2020. Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10043 с. 6-13, с. 22-48 .
11. Стельнік А. І. Пояснювальна записка до дипломної роботи на тему «Розробка програмного забезпечення тренажеру з теми «Регулярні мови» дистанційного навчального курсу «Теорія програмування»» / А. І. Стельнік // Полтава: Кафедра

ММСІ ПУЕТ, 2020. Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10306 с. 4-14, с. 40-59.

12. Волосевич М. В. Пояснювальна записка до дипломної роботи на тему «Розробка програмного забезпечення тренажеру з теми «Регулярні мови» дистанційного навчального курсу «Теорія програмування»» / А. І. Стельнік // Полтава: Кафедра ММСІ ПУЕТ, 2020. Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10306 с. 4-14, с. 40-59.
13. Белінська В.В. Пояснювальна записка до дипломної роботи на тему «Створення програмного забезпечення тренажера з теми «Розподіли дискретних випадкових величин та їх числові характеристики» дистанційного навчального курсу «Теорія ймовірностей та математична статистика» / В. В. Белінська // Полтава: Кафедра ММСІ ПУЕТ, 2021. Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10305 с. 4-15, с. 40-60.
14. Кривошей С.О. Оптимізація перевезень сільгосппродукції: програмна реалізація тренажера (моделювання та розв'язування) дистанційного курсу «Проектне навчання з дисципліни «Методи оптимізації та дослідження операцій» / О. С. Кривошей, О. О. Ємець // Комп'ютерні науки і прикладна математика (КНіПМ-2019): матеріали науково-практичного семінару. – Полтава: Кафедра ММСІ ПУЕТ, 2021. С. 31-43. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/7014
15. Хорстман К. С. Современный JavaScript для нетерпеливых / К. С. Хорстман. – ДМК Пресс, 2021