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
_____Олена ОЛЬХОВСЬКА
(підпис)
«_____»___202_ р.

КВАЛІФІКАЦІЙНА РОБОТА

на тему

«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТРЕНАЖЕРУ З ТЕМИ «НОРМАЛЬНІ АЛГОРИТМИ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ АЛГОРИТМІВ»»

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

Виконавець роботи Бондар Дарина Олександрівна
_____«____» _____202_ р.
(підпис)

Науковий керівник к. ф.-м. н., доцент, Черненко Оксана Олексіївна
_____«____»_____202_ р.
(підпис)

Рецензент

ПОЛТАВА 2023

ЗАТВЕРДЖУЮ
Завідувач кафедри___Олена ОЛЬХОВСЬКА
« ___» вересня 2022р.

ЗАВДАННЯ І КАЛЕНДАРНИЙ ГРАФІК ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

на тему «Розробка програмного забезпечення тренажеру з теми «Нормальні алгоритми» дистанційного навчального курсу «Теорія алгоритмів»»
зі спеціальності 122 Комп'ютерні науки
освітня програма «Комп'ютерні науки» ступеня бакалавр
Прізвище, ім'я, по батькові Бондар Дарина Олександрівна
Затверджена наказом ректора № 144-Н від «1» вересня 2022р.
Термін подання студентом роботи «__» _____202_ р.
Вихідні дані до кваліфікаційно роботи: публікації з теми, навчальні тренажери в дистанційних курсах із комп'ютерних наук.
Зміст пояснювальної записки

ВСТУП

1. ПОСТАНОВКА ЗАДАЧІ
2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1.Типи комп'ютерних тренажерів

2.2. Огляд прикладів комп'ютерних тренажерів

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1 Теоретичні відомості по темі «Теорія алгоритмів»

3.2 Приклади виконання завдань з теми

3.3 Алгоритмізація задачі за темою роботи

4. ПРАКТИЧНА ЧАСТИНА

4.1 Блок-схема

4.2 Опис процесу програмної реалізації

4.3 Необхідна користувачу інструкція програми

ВИСНОВКИ

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

ДОДАТОК А.

Перелік графічного матеріалу: 1 аркуш блок-схем, 20 ілюстрацій.

Консультанти розділів кваліфікаційної роботи

Розділ	ПІБ, посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Постанова задачі	Черненко О. О.		
Інформаційний огляд	Черненко О. О.		
Теоретична частина	Черненко О. О.		
Практична реалізація	Черненко О. О.		

Календарний графік виконання кваліфікаційної роботи

Зміст роботи	Термін виконання	Фактичне виконання
1. Вступ		
2. Вивчення методичних рекомендацій і стандартів та звіт керівнику		
3. Постановка завдання		
4. Інформаційний огляд джерел бібліотек та Інтернету		
5. Теоретична частина		
6. Практична частина		
7. Закінчення оформлення		
8. Доповідь студента на кафедрі		
9. Доопрацювання (за необхідності), рецен-зування		

Дата видачі завдання «__» __202_ р.

Здобувач вищої освіти Бондар Дарина Олександрівна

Науковий керівник к. ф.-м. н., Черненко О. О.

Результати захисту кваліфікаційної роботи

Кваліфікаційна робота оцінена на _____
(балів, оцінка за національною шкалою, оцінка за ЄКТС)

Протокол засідання ЕК № _____від «_____» _____202_ р.

Секретар ЕК _____ _____
(підпис) *(ініціал та прізвище)*

Затверджую
Зав. кафедрою _____
к. ф.-м. н. Олена ОЛЬХОВСЬКА
«__» _____202_ р.

Погоджено
Науковий керівник____
к. ф.-м. н. Оксана ЧЕРНЕНКО
«__» _____202_ р.

План

кваліфікаційної роботи на тему
«Розробка програмного забезпечення тренажеру з теми «Нормальні алгоритми» дистанційного навчального курсу «Теорія алгоритмів»»
зі спеціальності 122 Комп'ютерні науки
освітня програма 122 «Комп'ютерні науки» ступеня бакалавр
Прізвище, ім'я, по батькові Бондар Дарина Олександрівна

Здобувач вищої освіти _____Дарина БОНДАР
«__» _____202_ р.

РЕФЕРАТ

Записка: 54 с., 20 рис., 0 таблиць, 1 додаток, 10 джерел.

НОРМАЛЬНІ АЛГОРИТМИ, НАВЧАЛЬНИЙ ТРЕНАЖЕР, NETBEANS IDE

Об’єкт розробки – процес дистанційного навчання математичній дисципліні.

Мета роботи – розробка програмного забезпечення тренажеру з теми «Нормальні алгоритми» дистанційного курсу «Теорія алгоритмів».

Методи дослідження – методи теорії нормальних алгоритмів, середовище візуальної розробки програм NetBeans IDE та об’єктно-орієнтована мова програмування Java.

Було зроблено огляд вже створених навчальних тренажерів для дистанційного навчання, виділено їх позитивні та негативні сторони. Вивчено відповідний матеріал та розроблено алгоритм програми з теми «Нормальні алгоритми». Програмно реалізовано тренажер з дисципліни «Теорія алгоритмів».

Здійснено програмну реалізацію тренажеру з можливістю перегляду теоретичного матеріалу перед проходженням тестування.

Зміст

ВСТУП

З розвитком технологій у людства з’явилось безліч способів полегшити собі життя. Взяти, наприклад, ситуації минулих років, коли через коронавірус ми не могли працювати чи навчатися у звичному режимі. Тоді нам на допомогу прийшли технології, які дозволили виконувати все дистанційно – у навчанні використовувалися онлайн зустрічі, а для закріплення матеріалу різноманітні тести або тренажери.

Електронні тренажери взагалі є незамінними при підготовці деяких фахівців, бо дають можливість не лише ознайомитися з теорією, але й спробувати на практиці, що покращує рівень підготовки. До того ж не завжди студент може сприйняти інформацію та зрозуміти матеріал з першого разу на парі. Вирішити цю проблему можливо через створення та впровадження різноманітних тренажерів до дистанційних курсів. Програма-тренажер забезпечує:

- послідовне виведення на екран завдань заданої складності з вибраної теми;
- контроль за діями користувача з розв’язання запропонованого завдання;
- миттєву реакцію на неправильні дії;
- виправлення помилок користувача;
- демонстрацію правильного розв’язання завдання;
- виведення підсумкового повідомлення про результати роботи користувача.

Окрім вказаного вище електронні тренажери мають такі переваги:

- можливість покрокового контролю виконання завдань;
- можливість вивчати матеріал у поза-навчальний час;
- формування завдань різного рівня складності;
- відсутність контролю зі сторони викладача (контроль надається інформаційній системі), що значно підвищує ефективність роботи самого викладача.

Тож можна зазначити, що створення тренажерів є актуальним[1].

Об’єктом розробки є процес дистанційного навчання математичній дисципліні.

Предметом розробки є програмний продукт, що реалізує тренажер з теми «Нормальні алгоритми » на мові програмування Java.

Перелік використаних методів полягає в застосуванні заданого алфавіту, слів та схем перетворень.

Для реалізації курсового проекту використано середовище візуальної розробки програм NetBeans IDE та об’єктно-орієнтована мова програмування Java.

1. ПОСТАНОВКА ЗАДАЧІ

Щоб виконати кваліфікаційну роботу слід скласти алгоритм роботи електронного тренажеру для дистанційного курсу «Теорія алгоритмів». План роботи такий:

- розглянути теоретичний матеріал з теми «Нормальні алгоритми» та розібратися у ньому;
- переглянути приклади електронних тренажерів;
- підібрати приклади, які будуть використані у роботі;
- розробити алгоритм тренажеру.

Тренажер повинен не лише давати користувачу різносторонні завдання, але й надавати можливість переглянути перед виконанням матеріал по темі. Також його інтерфейс повинен бути простим та зрозумілим, тобто мати :

- стартову сторінку;
- можливість прочитати теоретичний матеріал;
- виведення завдання чи запитання;
- варіанти або поле для самостійної відповіді;
- показ повідомлення у разі помилки, де виведений приклад схожого завдання або підказка;
- можливість завершити роботу тренажеру у будь-який момент.

2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Типи комп’ютерних тренажерів

Перший тип комп'ютерних тренажерів – електронний екзаменатор. Основна його функція – це заміна живого екзаменатора в строго регламентованих областях. Залежно від складності екзаменатора, він може забезпечувати наступні можливості:

- показ малюнків в кадрі питання;
- показ мультфільмів (анімація) в кадрі питання;
- аналіз відповіді у вигляді чисел і формул;
- попереднє навчання (показ правильних відповідей);
- редагування старих і створення нових питань.

Вартість розробки подібних екзаменаторів найнижча.

Другий тип – статичні (або логіко-динамічні) тренажери. В таких програмах відсутня фізико-математична модель процесів, що відбуваються в устаткуванні, але показується і перевіряється певний порядок дій. Порядок дій зазвичай жорстко задається; у складніших випадках передбачаються розгалуження в ланцюжку дій, що забезпечується логічними функціями. Головні недоліки:

- неможливість відхилення від жорстко заданого ланцюжка дій;
- складність програмування динамічних ефектів.

Ці недоліки неістотні, але визначають неможливість моделювання складних фізичних процесів. Як правило, розробники таких тренажерів спочатку роблять для себе інструментальні засоби (конструктори), що дозволяють легко намалювати потрібні електричні схеми і задати ланцюжки правильних дій. Такі конструктори також пропонуються на продаж, але по вищих цінах. Проте, замовникові вигідно набувати конструктор, ніж готовий тренажер. Вартість розробки залежно від комплектності (готовий тренажер або конструктор) низька і середня. Прості тренажери цілком може написати студент.

Третій тип – динамічні тренажери. Вони мають в своїй основі математичну модель реальних фізичних процесів і тому найбільш корисні для якісного навчання персоналу. Зрозуміло, що для розрахунку достатньо повної моделі потрібні вельми

значні машинні ресурси, тому модель слід спростити, але так, щоб її поведінка в обумовлених технічним завданням рамках відповідало реальній системі з певною точністю. Це складне творче завдання для інженера і науковця. Можна з упевненістю сказати, що вартість розробки такого продукту в багато разів перевищує ціну, яку згоден сплатити замовник. Вартість розробки висока.

Четвертий тип – пультові тренажери. У них, окрім комп'ютера, присутня апаратна частина. На пульті можуть бути представлені тільки основні прилади і органи управління (спрощений тренажер); управління якою-небудь частиною, окремою установкою (локальний тренажер); нарешті, пульт може бути копією реального пульта управління (повномасштабний тренажер). Комп'ютер в даному випадку замінює реальний керований об'єкт. Тут, як правило, потрібна хороша динамічна модель, тому до вищенаведеної оцінки динамічних тренажерів доводиться також додати вартість розробки апаратної частини. Вартість розробки: висока і дуже висока.

П'ятий тип – сучасна комп'ютерна технологія (мультимедіа). Вона дозволяє створювати діалогові повчальні програми і тренажери, що включають комп'ютерну мультиплікацію, аудіо і відеотехніку. Як мінімум, це підсилить відчуття реальності при роботі з тренажером і відкриє нові можливості в процесі навчання. Вартість розробки: залежно від внутрішньої наукової начинки від середньої до дуже високої. Вартість необхідної комп'ютерної техніки: відносно висока[2].

2.2 Огляд прикладів комп'ютерних тренажерів

У ході підготовки до написання кваліфікаційної роботи я ознайомилась з кількома тренажерами. Першим було розглянуто тренажер з теми «Машини Тюрінга». Під час запуску з’являється вікно, де можна побачити розробника тренажеру, наукового керівника. Також на стартовій сторінці є дві кнопки: одна дозволяє завантажити файл з теоретичним матеріалом, а інша розпочати тестування.

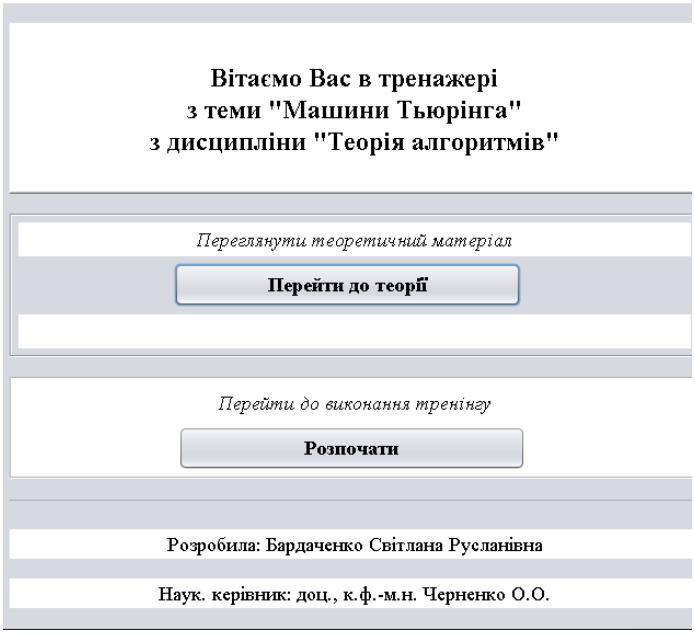

Рисунок 2.1 – Стартова сторінка тренажеру

На початку дається завдання з заданою машиною Тюрінга та словом для перетворення. На екрані постійно показується саме завдання, функціональна схема, схема поточних конфігурацій та питання. У ході тестування крок за кроком перетворюється слово, причому потрібно не лише вказувати результати кожного кроку, але й правильно визначити правило по функціональній схемі.

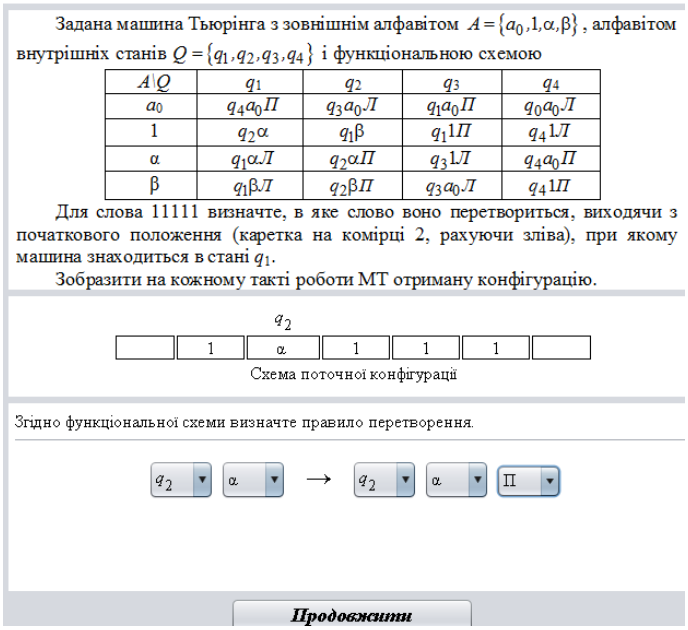

Рисунок 2.2 – Вибір правила

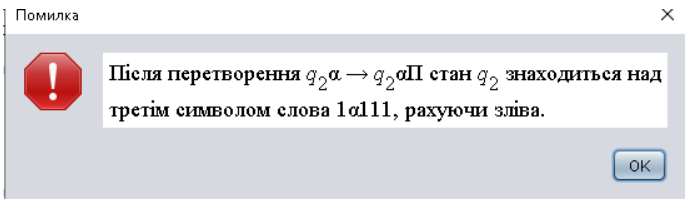

Рисунок 2.3 –Повідомлення про помилку

Після завершення цього завдання є ще кілька з вибором однієї відповіді, де потрібно обрати результат перетворення або чи зупиниться МТ взагалі. Після розв'язань прикладів можна розпочати тренінг знову або завершити виконання.[3]

Наступний розглянутий тренажер стосується теми «Дерево розбору». На стартовій сторінці вказано розробника та наукового керівника. Кнопка «Інформаційна сторінка» відкриває додаткове вікно з теорією до теми, а «Перейти до тестування» дає на вибір два варіанти з чого розпочати. Можна обрати теорію чи практичні завдання.

У частині про теорію дані питання з трьома варіантами відповідей, у разі помилки виводиться підказка. По закінченню тесту тренажер покаже скільки правильних відповідей дав користувач.

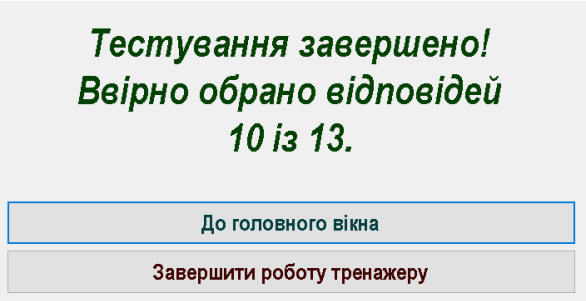

Рисунок 2.4 – Результати першої частини другого тренажеру

Щодо практичної частини тренажеру, то одразу дається вибір складності завдань. На легкому рівні дані питання про базову теорію, на середньому та складному завдання з великою кількістю кроків.[4]

Третім було розглянуто тренажер на таку ж тему «Нормальні алгоритми». У тренінгу дані на розв'язок три завдання. На екрані завжди є умова та послідовність виконання, що полегшує роботу. У разі неправильної відповіді користувач отримає підказку. Мінусом є те, що завдання однотипні та лише практичні.[5]

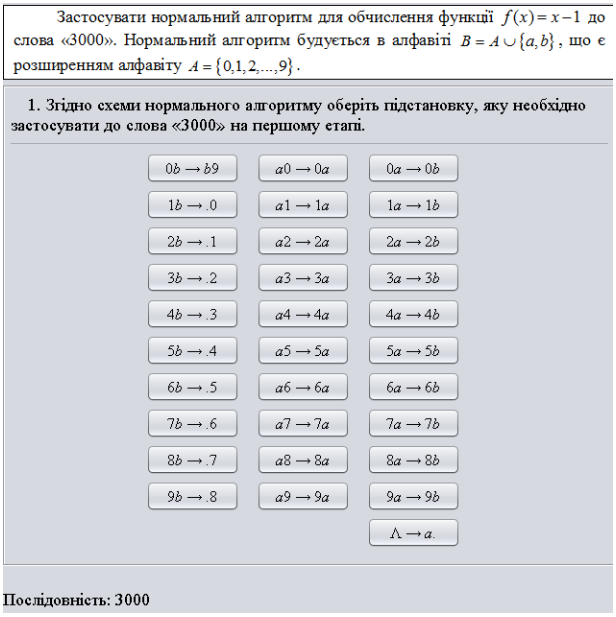

Рисунок 2.5 – Завдання третього тренажеру

3. ТЕОРЕТИЧНА ЧАСТИНА

3.1 Теоретичні відомості по темі «Нормальні алгоритми»

Теорія нормальних алгоритмів була створена радянським математиком А. А. Марковим в кінці 40-55 рр. XX ст. Нормальні алгоритми представляють собою деякі правила по перетворенню слів у деякому алфавіті, так що вхідні дані і результат алгоритмів є словами в деякому алфавіті.

Алфавіт – довільна непорожня множина символів. Його елементи – це символи (букви). Слово в даному алфавіті – довільна послідовність букв. Для зручності міркувань допускаються порожні слова (в їх складі немає жодної букви). Порожні слова позначаються Λ (“лямбда”). Якщо A і B – два алфавіта, причому $A \leq B$, то алфавіт B н-ся розширенням алфавіта A.

Слова позначаються великими латинськими літерами P,Q ,R (або буквами з індексами). Одне слово може бути складовою частиною іншого слова. Тоді перше слово називається підсловом другого, або входженням в друге.

Означення 1. Марківською підстановкою (МП) називається операція над словами, що задається за допомогою впорядкованої пари слів (P,Q), суть якої в наступному: в заданому слові R знаходять перше входження слова P (якщо таке є) і, не змінюючи інших частин слова R, замінюють у ньому це входження словом Q. Отримане слово називають результатом використання МП (P, Q) до слова R. Якщо ж першого входження P в слово R немає (і відповідно, P в R), то вважається що МП (P, Q) незастосована до слова R.

Окремими випадками марківських підстановок є підстановки з порожніми словами (Λ, Q),(P, Λ),(Λ, Λ).

Означення 2. Формулою підстановки (P, Q) називається марківська підстановка, що позначається $P \to Q$. Слово P називається лівою частиною в формулі підстановки, а Q – правою частиною.

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

Означення 4. Упорядкований скінчений список підстановок

$P_1 \to (.)Q_1$

$P_2 \to (.)Q_2$

$P_r \to (.)Q_r$

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

Означення 5. Нормальним алгоритмом (алгоритмом Маркова) в алфавіті A називається таке правило побудови послідовності V_i слів в алфавіті A:

1. В якості початкового слова V_0 послідовності беремо вихідне слово V_i
2. Нехай для деякого $i \geq 0$ слово V_i побудованого і процес побудови розглядуваної послідовності ще не завершене якщо при цьому в схемі НА немає формул, ліві частини яких входили б в V_i , то $V_{i+1} = V_i$ і процес побудови вважають завершеним. Якщо такі формули є, то в якості V_{i+1}вибираємо результат марківської підстановки правої частини першої із таких формул замість першого входження й лівої частини в слово V_i.
3. Процес побудови послідовності є завершеним, якщо на даному кроці була використана формула завершальної підстановки; і продовжується - в іншому випадку.

Означення 6. Якщо процес побудови завершується, то розглянутий нормальний алгоритм (НА) застосований до слова V.

Означення 7. Останнє слово послідовності називається результатом використання НА до слова V.

При цьому НА перетворює слово V в слово W.

Послідовність V_i записується так

$V_0 \to V_i \to V_2 \to \cdots \to V_m$, де $V_0 = V_1$, $V_m = W$.

Для випадку, якщо алгоритм задано в деякому розширенні алфавіту A,то це є НА над алфавітом A.

Означення 8. Якщо процес перетворення слова не завершується, результат перетворення цього слова заданим НА невизначений.

Як і МТ, нормальні алгоритми не виконують власне обчислень і вони лише

виконують перетворення слів, замінюючи в них одні букви іншими по заданим правилам.

Функція f, задана на деякій множині алфавіту A, називається нормально обчислюваною, якщо знайдеться таке розширення B даного алфавіту ($A \subseteq B$) і такий нормальний алгоритм в B, що кожне слово V (в алфавіті A) із області визначення функції f цей алгоритм перетворює в слово f(V).

Нагадаємо, що розглядаються числові функції, тобто функції область визначення і область значень яких співпадає з множиною натуральних чисел.

Натуральні числа кодуються словами в алфавіті A={1}. Для кодування чисел словами будемо використовувати десяткову систему числення, тобто алфавіт, що складається з десяти букв (цифр): A={0, 1, 2, …,9}. Його будемо називати основним алфавітом.

Таким чином, нормальні алгоритми прикладів 1 та 2 показують, що функції f(x)=x+1 та $\varphi_3(x)$ нормально обчислювані. Причому відповідні нормальні алгоритми вдалося побудувати в тому ж самому алфавіті A, на словах якого були задані розглянуті функції, тобто розширювати алфавіт не довелося (B=A).

Основоположник теорії нормальних алгоритмів Марков висунув гіпотезу (принцип нормалізації Маркова): для знаходження значень функції, заданої в деякому алфавіті, тоді і тільки тоді існує алгоритм, коли функція нормально обчислювана.[6]

3.2 Приклади виконання завдань з теми

Приклад 1.

В алфавіті $B = A \cup \{a, b, c\}$, що є розширенням алфавіту А, розглянемо

нормальний алгоритм, що задається схемою:

$0b \to \cdot\, 2$	$a0 \to 0a$	$0a \to 0b$	$0c \to \cdot\, 1$
$1b \to \cdot\, 3$	$a1 \to 1a$	$1a \to 1b$	$1c \to \cdot\, 2$
$2b \to \cdot\, 4$	$a2 \to 2a$	$2a \to 2b$	$2c \to \cdot\, 3$
$3b \to \cdot\, 5$	$a3 \to 3a$	$3a \to 3b$	$3c \to \cdot\, 4$
$4b \to \cdot\, 6$	$a4 \to 4a$	$4a \to 4b$	$4c \to \cdot\, 5$
$5b \to \cdot\, 7$	$a5 \to 5a$	$5a \to 5b$	$5c \to \cdot\, 6$
$6b \to \cdot\, 8$	$a6 \to 6a$	$6a \to 6b$	$6c \to \cdot\, 7$
$7b \to \cdot\, 9$	$a7 \to 7a$	$7a \to 7b$	$7c \to \cdot\, 8$
$8b \to \cdot\, c0$	$a8 \to 8a$	$8a \to 8b$	$8c \to \cdot\, 9$
$9b \to \cdot\, c1$	$a9 \to 9a$	$9a \to 9b$	$9c \to \cdot\, c0$
			$c \to \cdot\, 1$
			$\Lambda \to a$

Рисунок 3.1 – Схема алгоритму першого прикладу

Застосувати його до наступних слів:

а) 1998;

б) 999;

Розв'язання.

а)На першому кроці потрібно застосувати найпростішу підстановку $\Lambda \to a$, у нас вийде $1998 \Rightarrow a1998$. Далі застосовуються підстановки з другого стовпця, до тих пір, поки *a* не займе крайнє праве положення: $a1998 \Rightarrow 1a998 \Rightarrow 19a98 \Rightarrow 199a8 \Rightarrow 1998a$. Тепер спрацьовує одна з підстановок третього стовпця, у цьому випадку дев'ята: $1998a \Rightarrow 1998b$. Далі застосовується підстановка з першого стовпця: $1998b \Rightarrow 199c0$. Наступним кроком слід використати перетворення з четвертого стовпця: $199c0 \Rightarrow 19c00 \Rightarrow 1c000 \Rightarrow 2000$. На цьому перетворення завершене.

б)Знову ж таки спочатку застосовується $\Lambda \to a$, отримуємо: $a999$. Слідуємо підстановкам другого стовпця доки a не займе крайнє положення:

$a999 \Rightarrow 9a99 \Rightarrow 99a9 \Rightarrow 999a$. За підстановкою третього стовпця отримаємо: $999a \Rightarrow 999b$. Звертаємось до першого стовпця: $999b \Rightarrow 99c1$. Так як у нас було число більше 7, тепер потрібно застосувати підстановки з четвертого стовпця.: $99c1 \Rightarrow 9c01 \Rightarrow c001 \Rightarrow 1001$. Перетворення завершене [7].

3.3 Алгоритмізація задачі за темою роботи

Після запуску програми спочатку користувач бачить головну сторінку, на якій зазначена інформація про тренажер, є кнопка, яка дає можливість переглянути теоретичний матеріал перед виконанням завдань, та кнопка «Розпочати тренінг». При натисненні на неї відображається вибір складності, який у свою чергу запускає відповідне тестування.

Якщо обрано легкий рівень, на панелі виводяться питання та варіанти відповідей.

1 крок. Вкажіть, як позначають слова в теорії Маркова позначать:

- Маленькими латинськими літерами p, q, r (або буквами з індексами).
- Довільними символами.
- Великими латинськими літерами P, Q, R (або буквами з індексами).

При неправильній відповіді користувачу вказується правильна, якщо ж відповідь вірна – перехід на наступний крок.

2 крок. Виберіть класи функцій, які співпадають(відповідь не одна):

- Клас усіх частково рекурсивних функцій.
- Клас всіх обчислюваних функцій.
- Клас усіх рекурсивних функцій.
- Клас усіх функцій обчислюваних за Тюрінгом.
- Клас усіх нормально обчислюваних функцій.
- Клас елементарних функцій.

При неправильних варіантах вказується помилка. При правильних – перехід на наступний крок.

3 крок. Дані такі слова P_1: корабель, P_2: кора, P_3:бель. Оберіть правильне твердження:

- P_2 є підсловом P_3.
- P_2 та P_3 є підсловами P_1.
- P_1 є входженням в P_2 та P_3.

При неправильній відповіді користувачу вказується правильна, якщо ж

відповідь вірна – перехід на наступний крок.

4 крок. Установіть відповідність між твердженнями та їх значеннями.

Нормальний алгоритм	Фіксована множина символів
Λ	Довільна непорожня множна символів.
Слово.	Довільна множина символів.
Алфавіт	Довільна послідовність букв
	Змістовна послідовність символів.
	Упорядкований скінченний список підстановок.
	Скінченний список підстановок.
	Порожнє слово

При неправильній відповіді користувачу вказується правильна, якщо ж відповідь вірна – перехід на наступний крок.

5 крок. Як звучить принцип нормалізації Маркова?

- Для знаходження значень функції, заданої в деякому алфавіті, тоді і тільки тоді існує алгоритм, коли функція обчислювана.
- Для знаходження значень функції, заданої в деякому алфавіті, існує алгоритм, коли функція нормально обчислювана.
- Для знаходження значень функції, заданої в деякому алфавіті, тоді і тільки тоді існує алгоритм, коли функція нормально обчислювана.
- Для знаходження значень функції, заданої в деякому алфавіті, існує алгоритм, коли функція обчислювана.

При неправильній відповіді користувачу вказується правильна, якщо ж відповідь вірна – з’являється повідомлення: «Ви пройшли легкий рівень тренажеру» та відбувається перехід на стартову сторінку.

Якщо користувач обрав середній рівень, на панелі виводиться перше завдання: «Нормальний алгоритм в алфавіті $A\{a, b, 1\}$ задається схемою: $a \to 1$, $b \to 1$. Застосуйте його до слів $ababaa,\ 11aab, baaab1a$.»

Розв'язання.

1 крок. «Запишіть перший крок перетворення слова $ababaa$ – (_,_,_,_,_,_)».

Тут потрібно вписати: (1,b,1,b,1,1). Якщо користувач зробив помилку – висвітиться підказка: «Неправильно. Вірна відповідь – (1,b,1,b,1,1). Тепер запишіть другий крок перетворення – (_,_,_,_,_,_)», якщо все вірно – перехід до наступного кроку.

2 крок. «Тепер запишіть другий крок перетворення – (_,_,_,_,_,_)». Слідуючи схемі, потрібно замінити усі b на 1. Відповідь: (1,1,1,1,1,1). Якщо користувач зробив помилку – висвітиться підказка: «Слід було застосувати другу підстановку. Вірна відповідь – (1,1,1,1,1,1)», якщо все вірно – перехід до наступного кроку.

3 крок. «Запишіть перший крок перетворення слова $11aab$ – (_, _, _, _,_). Тут потрібно вписати: (1,1,1,1,b). Якщо користувач зробив помилку – висвітиться підказка: «Неправильно. Вірна відповідь - (1,1,1,1,b). Тепер запишіть другий крок перетворення - (_, _, _, _, _)», якщо все вірно – перехід до наступного кроку.

4 крок. «А тепер запишіть другий крок перетворення – (_, _, _, _,_).» Слідуючи схемі, потрібно замінити усі b на 1. Відповідь: (1,1,1,1,1). Якщо користувач зробив помилку висвітиться підказка: «Слід було застосувати другу підстановку. Вірна відповідь - (1,1,1,1,1)», якщо все вірно – перехід до наступного кроку.

5 крок. «Запишіть перший крок перетворення $\underline{baaab1a}$ – (_, _, _, _,_,_,_).»
Тут потрібно вписати: (b,1,1,1,b,1,1). Якщо користувач зробив помилку – висвітиться підказка: «Неправильно. Вірна відповідь - (b,1,1,1,b,1,1). Тепер запишіть другий крок перетворення – (_, _, _, _, _ ,_,_)», якщо все вірно – перехід до наступного кроку.

6 крок. «А тепер запишіть другий крок перетворення – (_, _, _, _, _ ,_,_).» Слідуючи схемі, потрібно замінити усі b на 1. Відповідь: (1,1,1,1,1,1,1). Якщо користувач зробив помилку висвітиться підказка: «Застосуйте другу підстановку», якщо все вірно – перехід до наступного кроку.

2 завдання.

Користувачу виводиться завдання: «Нормальний алгоритм в алфавіті $A\{a, b,\}$ задається схемою: $ab \rightarrow a$, $b \rightarrow \Lambda$, $a \rightarrow b$. Застосуйте його до слова - $abbbaaab$».

7 крок. Виводиться запитання: «Яким буде перший крок перетворення слова $abbbaaab$?». З даних варіантів потрібно обрати вірний.

- abbaaab;
- abaaab;
- bbbbaaab;
- abbbaab.

Якщо відповідь неправильна виводиться підказка: «Спочатку застосовується перша підстановка», якщо вірна – перехід на наступний крок.

8 крок. Виводиться запитання: «Яким має бути наступний крок для abbaaab?».

- aaab;
- bbaaab;
- abaaab;
- bbbaa.

У разі помилки з'являється підказка: «У слові ще є де застосувати першу підстановку», якщо все вірно – виводиться послідовність та відбувається перехід до наступного кроку.

9 крок. Користувачу виводиться запитання: «Вкажіть наступне перетворення для abaaab».

- aaaab;
- bbaaab;
- bbbaab;
- aaaa.

У разі помилки з'являється підказка: «У слові ще є де застосувати першу підстановку», якщо все вірно – виводиться послідовність та відбувається перехід до наступного кроку.

10 крок. Користувачу виводиться запитання: «Вкажіть наступне перетворення для aaaab».

- baaab;
- aaabb;
- baaaa;
- aaaa.

Якщо користувач помиляється – підказка: «У слові ще є де застосувати першу підстановку», якщо все вірно – виводиться послідовність та відбувається перехід до наступного кроку.

11 крок. Користувачу виводиться запитання: «Яким буде наступне перетворення для aaaa?». Потрібно вписати вірну відповідь – (_, _, _, _).

Якщо вводиться – (b, a, a, a), відбувається перехід на наступний крок, а якщо є помилка з'являється підказка: «Будь уважним, слід застосувати третю підстановку».

12 крок. Користувачу виводиться питання: «Вкажіть наступне перетворення

baaa».

- bbaa;
- aaa.

Якщо відповідь неправильна, користувачу виводиться підказка: «Перетворення з третьою підстановкою ще не завершені», у іншому випадку перехід на наступний крок.

13 крок. Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово bbaa через два кроки».

- aa;
- bbbb.

У разі помилки підказка: «Перетворення з третьою підстановкою не завершені».

14 крок. Користувачу дається завдання: «Вкажіть фінальний результат перетворення слова bbbb».

- bbbb;
- Λ.

У разі помилки на екрані виводиться підказка: «Зверніть увагу на схему перетворень», якщо ж відповідь вірна відбувається перехід на наступний крок.

3 завдання.

Користувачу виводиться завдання: «Нормальний алгоритм заданий схемою: bb → ba, ba → a, ba → a, b → Λ. Застосуйте його до слова – bbaab».

Розв'язання.

15 крок. На екрані виводиться запитання: «Яку підстановку першою слід застосувати до слова bbaab?».

- bb → ba;
- a → Λ;
- ba → a;
- b → Λ.

Якщо відповідь неправильна, то на екрані виводиться підказка: «Першою слід застосувати підстановку bb → ba», у іншому разі з'являється перший крок перетворення bbaab → baaab та відбувається перехід на наступний крок.

16 крок. На екрані виводиться запитання: «Яку підстановку далі слід застосувати до слова baaab?».

- bb → ba;
- ba → a;

- a → Λ;
- b → Λ.

Якщо відповідь неправильна, то на екрані виводиться підказка: «Тут слід застосувати підстановку ba → a», у іншому разі з'являється процес перетворення bbaab → baaab → aaab та відбувається перехід на наступний крок.

17 крок. На екрані виводиться запитання: «Яку підстановку далі слід застосувати до слова aaab?».

- bb → ba;
- a → Λ;
- ba → a;
- b → Λ.

Якщо відповідь неправильна, то на екрані виводиться підказка: «Тут застосовується підстановка a → Λ», у іншому разі з'являється процес перетворення bbaab → baaab → aaab → aab та відбувається перехід на наступний крок.

18 крок. На екрані виводиться запитання: «Яку підстановку далі слід застосувати до слова aab?».

- bb → ba;
- a → Λ;
- ba → a;
- b → Λ.

Якщо відповідь неправильна, то на екрані виводиться підказка: «Тут слід застосувати підстановку a → Λ», у іншому разі з'являється процес перетворення bbaab → baaab → aaab → aab → ab та відбувається перехід на наступний крок.

19 крок. На екрані виводиться завдання: «Оберіть підстановку ,яку слід застосувати на наступному кроці».

- a → Λ;
- b → Λ.

Якщо відповідь вірна, то відбувається перехід до наступного кроку та виводиться послідовність bbaab → baaab → aaab → aab → ab → b, якщо ж ні – підказка: «Потрібно використати підстановку a → Λ».

20 крок. На екрані виводиться завдання: «Оберіть як буде виглядати слово b на наступному кроці».

- Λ;
- b.

Якщо відповідь вірна, то виводиться послідовність bbaab → baaab → aaab → aab → ab → b → Λ та надпис: «Ви пройшли середній рівень тренажеру» та відбувається перехід на стартову сторінку.

Якщо користувач обрав складний рівень – на панелі виводяться перше завдання: «В алфавіті $B = A \cup \{a, b, c\}$, що є розширенням алфавіта А, розглянемо нормальний алгоритм, що задається схемою. Застосуйте його до наступних слів: 999, 1998.»

0b →· 2	a0 → 0a	0a → 0b	0c →· 1
1b →· 3	a1 → 1a	1a → 1b	1c →· 2
2b →· 4	a2 → 2a	2a → 2b	2c →· 3
3b →· 5	a3 → 3a	3a → 3b	3c →· 4
4b →· 6	a4 → 4a	4a → 4b	4c →· 5
5b →· 7	a5 → 5a	5a → 5b	5c →· 6
6b →· 8	a6 → 6a	6a → 6b	6c →· 7
7b →· 9	a7 → 7a	7a → 7b	7c →· 8
8b → c0	a8 → 8a	8a → 8b	8c →· 9
9b → c1	a9 → 9a	9a → 9b	9c → c0

c →· 1

∧ → a

1 крок.

Користувачеві відображається питання: «Із запропонованих варіантів оберіть той, що буде результатом після підстановки ∧ → a до слова 999» та наводяться варіанти:

- 999 → a999;
- 999 → 9a99 ;
- 999 → a9a99;
- 999 → 999a.

Якщо обрано перший варіант - відображується отримана послідовність, тобто 999 → a999, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 999 формули ∧ → a отримаємо a999».

2 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово a999 після застосування до нього підстановки a9 → 9a» та наводяться варіанти:

- a999 → a999;
- a999 → 9a99;

- a999 → 9b99;
- a999 → a9a99.

Якщо обирається другий варіант, користувачу відображується отримана послідовність, тобто 999 → a999 → 9a99 , та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова a999 підстановки a9 → 9a отримаємо 9a99».

3 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово 9a99 після застосування до нього підстановки a9 → 9a» та наводяться варіанти:

- 9a99 → 99a9;
- 9a99 → 99b9;
- 9a99 → 9a9a9;
- 9a99 → 999a.

Якщо обирається перший варіант, користувачу відображується отримана послідовність, тобто 999 → a999 → 9a99 → 99a9, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 9a99 підстановки a9 → 9a отримаємо 99a9».

4 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово 99a9 після застосування до нього підстановки a9 → 9a» та наводяться варіанти:

- 99a9 → 99b9;
- 99a9 → 999a;
- 99a9 → 99b9a;
- 99a9 → 99a9.

Якщо обирається третій варіант, користувачу відображується отримана послідовність, тобто 999 → a999 → 9a99 → 99a9 → 999a, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 99a9 підстановки a9 → 9a отримаємо 999a».

5 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово 999a після застосування до нього підстановки 9a → 9b » та наводяться варіанти:

- 999a → 99b9;
- 999a → 99a9;
- 999a → 999a;
- 999a → 999b.

Якщо обирається четвертий варіант, користувачу відображується отримана

послідовність, тобто $999 \rightarrow a999 \rightarrow 9a99 \rightarrow 99a9 \rightarrow 999a \rightarrow 999b$,та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова $999a$ підстановки $9a \rightarrow 9b$ отримаємо $999b$».

6 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово $999b$ після застосування до нього підстановки $9b \rightarrow c1$ » та наводяться варіанти:

- $999b \rightarrow 99c1b$;
- $999b \rightarrow 9c19$;
- $999b \rightarrow 99c1$;
- $999b \rightarrow 99c19b$.

Якщо обирається другий варіант – користувачу відображується отримана послідовність, тобто $999 \rightarrow a999 \rightarrow 9a99 \rightarrow 99a9 \rightarrow 999a \rightarrow 999b \rightarrow 99c1$,та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова $999b$ підстановки $9b \rightarrow c1$ отримаємо $99c1$».

7 крок

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово $99c1$ після застосування до нього підстановки $9c \rightarrow c0$ » та наводяться варіанти:

- $99c1 \rightarrow 9c01$;
- $99c1 \rightarrow 90c1$;
- $99c1 \rightarrow 99c0$;
- $99c1 \rightarrow 99c01$.

Якщо обирається перший варіант, користувачу відображується отримана послідовність, тобто $999 \rightarrow a999 \rightarrow 9a99 \rightarrow 99a9 \rightarrow 999a \rightarrow 999b \rightarrow 99c1 \rightarrow 9c01$,та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова $99c1$ підстановки $9c \rightarrow c0$ отримаємо $9c01$».

8 крок.

Користувачу виводиться запитання: «Вкажіть, як буде виглядати слово $9c01$ після застосування до нього підстановки $9c \rightarrow c0$ » та наводяться варіанти:

- $9c01 \rightarrow 9cc001$;
- $9c01 \rightarrow 9c001$;
- $9c01 \rightarrow 99c0c0$;
- $9c01 \rightarrow c001$.

Якщо обирається четвертий варіант, користувачу відображується отримана послідовність, тобто $999 \rightarrow a999 \rightarrow 9a99 \rightarrow 99a9 \rightarrow 999a \rightarrow 999b \rightarrow 99c1 \rightarrow 9c01 \rightarrow c001$,та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При

застосуванні до слова 9c01 підстановки 9c → c0 отримаємо c001».

9 крок.

Користувачу виводиться запитання: «Вкажіть як буде виглядати слово c001 після застосування до нього підстановки c → .1» та наводяться варіанти:

- c001 → 1001;
- c001 → c. 1001;
- c001 → c001;
- c001 → c001.

Якщо обирається перший варіант, на панелі відображується отримана послідовність, тобто 999 → a999 → 9a99 → 99a9 → 999a → 999b → 99c1 → 9c01 → c001 → 1001, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова c001 підстановки c → .1 отримаємо 1001».

10 крок. Користувачу виводиться запитання: «Чи завершений процес застосування нормального алгоритму? » та варіанти відповіді:

- Ні, процес повинен бути продовжений;
- Так, робота алгоритму завершена.

Якщо користувач обрав перший варіант – відображується повідомлення про помилку: «Процес завершено!», якщо ж обрано другий – перехід до наступної частини завдання.

11 крок. Користувачеві відображається завдання: «Оберіть зліва підстановку першого кроку, а потім як буде виглядати слово після її застосування:».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «ʌ → a» ,то стає доступним вибір результату після застосування:

- 1998 → a1a998;
- 1998 → a1998 ;
- 1998 → 19a98;
- 1998 → 1998a.

Якщо обрано третій варіант – відображується отримана послідовність, тобто 1998 → a1998 та відбувається перехід на наступний крок, інакше – повідомлення про помилку: «При застосуванні до слова 1998 формули ʌ → a отримаємо a1998».

12 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова a1998».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана

кнопка «a1 → 1a» ,то стає доступним вибір результату після застосування:

- a1998 → a1a998;
- a1998 → b1*a*998;
- a1998 → 1a998;
- a1998 → 1c998.

Якщо обирається варіант «a1998 → 1a998», відображується отримана послідовність, тобто 1998 → a1998 → 1a998, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова a1998 підстановки a1 → 1a отримаємо 1a998».

13 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 1a998».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «a9 → 9a» ,то стає доступним вибір результату після застосування:

- 1a998 → 19a98;
- 1a998 → 1*a*9*a*98;
- 1a998 → 1a998;
- 1a998 → 19a9a8.

Якщо обирається перший варіант, відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 1a998 підстановки a9 → 9a отримаємо 19*a*98».

14 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 19*a*98».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «a9 → 9a» ,то стає доступним вибір результату після застосування:

- 19a98 → 199a8;
- 19a98 → 1*a*9*a*9*a*8;
- 19a98 → 1a9a98;
- 19a98 → 19a9a8.

Якщо обирається перший варіант, відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 19a98 підстановки a9 → 9a отримаємо 199a8».

15 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 199a8».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «a8 → 8a» ,то стає доступним вибір результату після застосування:

- 199a8 → 199a8a;
- 199a8 → 1a9a9a8a;
- 199a8 → 19a98a;
- 199a8 → 1998a.

Якщо обирається четвертий варіант – відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8 → 1998a, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 199a8 підстановки a8 → 8a отримаємо 1998a».

16 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 1998a».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «8a → 8b» ,то стає доступним вибір результату після застосування:

- 1998a → 1998ab;
- 1998a → 1998b;
- 1998a → 19a98b;
- 1998a → 199b.

Якщо обирається третій варіант – користувачу відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8 → 1998a → 1998b, та відбувається перехід на наступний крок, інакше – повідомлення про помилку: «При застосуванні до слова 1998a підстановки 8a → 8b отримаємо 1998b».

17 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 1998b».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «8b → c0» ,то стає доступним вибір результату після застосування:

- 1998b → 199c0a;
- 1998b → 1998c0;
- 1998b → 199c0;
- 1998b → 1998b.

Якщо обирається другий варіант – відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8 → 1998a → 1998b → 199c0, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 1998b підстановки 8b → c0 отримаємо 199c0».

18 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 199c0».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «9c → c0» ,то стає доступним вибір результату після застосування:

- 199c0 → 19c0;
- 199c0 → 19c9c0;
- 199c0 → 199c0;
- 199c0 → 19c00.

Якщо обирається четвертий варіант - користувачу відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8 → 1998a → 1998b → 199c0 → 19c00, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 199c0 підстановки 9c → c0 отримаємо 19c00».

19 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 19c00».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «9c → c0» ,то стає доступним вибір результату після застосування:

- 19c00 → 1c000;
- 19c00 → 1c00;
- 19c00 → 19c0c0;
- 19c00 → 19c00.

Якщо обирається перший варіант - користувачу відображується отримана послідовність, тобто 1998 → a1998 → 1a998 → 19a98 → 199a8 → 1998a → 1998b → 199c0 → 19c00 → 1c000
, та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 19c00 підстановки 9c → c0 отримаємо 1c000».

20 крок. На панелі відображується завдання: «Оберіть наступну підстановку для слова 1c000».

Зліва на панелі знаходяться кнопки з кожною підстановкою. Якщо обрана кнопка «1c →· 2» ,то стає доступним вибір результату після застосування:

- 1c000 → c2000;
- 1c000 → 2c000;
- 1c000 → 1.2000;
- 1c000 → .2000.

Якщо обирається четвертий варіант - користувачу відображується отримана послідовність та відбувається перехід на наступний крок, інакше - повідомлення про помилку: «При застосуванні до слова 1c000 підстановки 1c → .2 отримаємо

2000».

21 крок. Користувачу виводиться запитання: «Чи завершений процес застосування нормального алгоритму? » та варіанти відповіді:

- Ні, процес повинен бути продовжений;
- Так, робота алгоритму завершена.

Якщо користувач обрав перший варіант – відображується повідомлення про помилку: «Процес завершено!», якщо ж обрано другий – перехід до наступного завдання.

2 завдання

22 крок. Користувачу на екрані виводиться умова: «Сконструюйте нормальний алгоритм, що обчислює функцію f(w) = wu, яка до кожного слова w в алфавіті A додає справа фіксоване слово u. Для шуканого нормального алгоритму даний алфавіт $B = A \cup \{2\} = \{0,1\} \cup \{2\} = \{0,1,2\}$. Оберіть зі списку запропонованих перетворень потрібні і вирішіть короткий приклад аби продемонструвати, що алгоритм працює» та на панелі зліва наводиться потрібна інформація:

$21 \to 12$	$21 \to 20$
$\Lambda \to 1$	$2 \to .u$
$20 \to 2$	$20 \to 02$
$\Lambda \to u2$	$2 \to 2u$
$20 \to 01$	$\Lambda \to 2$

23 крок. Користувач обирає з запропонованих варіантів ті перетворення, які на його думку допоможуть вирішити завдання.

Якщо його вибір складається з: «$20 \to 02,\ 21 \to 12,\ 2 \to .u,\ \Lambda \to 2$», то нижче з'являється схема, а також розблоковується наступна панель для розв'язку прикладу. У іншому випадку йому дається шанс переобрати відповіді.

24 крок. На другій панелі відображається завдання: «Тепер ми маємо схему. Застосуйте її до слова 0111001. Яку підстановку слід використати першою?» та наводяться чотири варіанти з алгоритму.

Якщо користувач обирає підстановку «$\Lambda \to 2$», то ця панель блокується,

розблоковується третя та наступне завдання, а також нижче можна бачити ланцюжок перетворень. У іншому випадку – підказка.

25 крок. На екрані є завдання: «Впишіть, як буде виглядати слово 20111001 після використання підстановки $20 \rightarrow 02$?».

Якщо користувач вводить «02111001», то відбувається перехід на наступний крок, тобто третя панель блокується, а друга знову стає доступною. Інакше – підказка.

26 крок. На другій панелі відображається завдання: «Яку з підстановок слід застосувати до слова 02111001?».

Якщо обрано «$21 \rightarrow 12$», то відбувається перехід на наступний крок, тобто друга панель блокується, третя стає доступною та нижче продовжується ланцюжок перетворення. У іншому випадку – підказка.

27 крок. На третій панелі відображується завдання: «Впишіть, як буде виглядати слово 01211001 після використання підстановки $21 \rightarrow 12$?».

Якщо введено «01121001», то відбувається перехід на наступний крок, тобто ця панель блокується, друга стає доступною, а ланцюжок продовжує будуватись. У іншому випадку – підказка.

28 крок. На другій панелі відображується питання: «Яку з підстановок слід застосувати до слова 01121001?».

Якщо обрано «$21 \rightarrow 12$», то відубвається перехід на наступний крок, тобто ця панель блокується, третя стає доступною. У іншому випадку – підказка.

29 крок. На третій панелі з`являється завдання: «Впишіть, як буде виглядати слово 01121001 після використання підстановки $21 \rightarrow 12$?».

Якщо введено «01112001», то відбувається перехід на наступний крок. Інакше – підказка.

30 крок. На цій же панелі відображається наступне завдання: «Впишіть, як буде виглядати слово 01112001 після використання підстановки $20 \rightarrow 02$?».

Якщо введено «01110201», то відбувається перехід на наступний крок, у іншому випадку – підказка.

31 крок. Відображується завдання: «Впишіть, як буде виглядати слово

01110201 після використання підстановки $20 \rightarrow 02$?».

Якщо введено «01110021», то відбувається перехід на наступний крок, інакше – підказка.

32 крок. Відображується завдання: «Впишіть, як буде виглядати слово 01110021 після використання підстановки $21 \rightarrow 12$?».

Якщо введено «01110012», то відбувається перехід на наступний крок, тобто ця панель блокується, друга стає доступною та будується ланцюжок. У іншому випадку – підказка.

33 крок. На другій панелі з’являється запитання: «Яку з підстановок слід застосувати до слова 01110012?»

Якщо обрано «$2 \rightarrow .u$», то всі панелі блокується, висвічується повідомлення про те, що перетворення завершене, добудовується ланцюг, а також стає доступною кнопка «Завершити рівень». Інакше – підказка.

34 крок. При натисненні на кнопку «Завершити рівень», з’являється повідомлення про це та відбувається повернення на стартову сторінку[8,6].

4. ПРАКТИЧНА ЧАСТИНА

4.1 Блок-схема

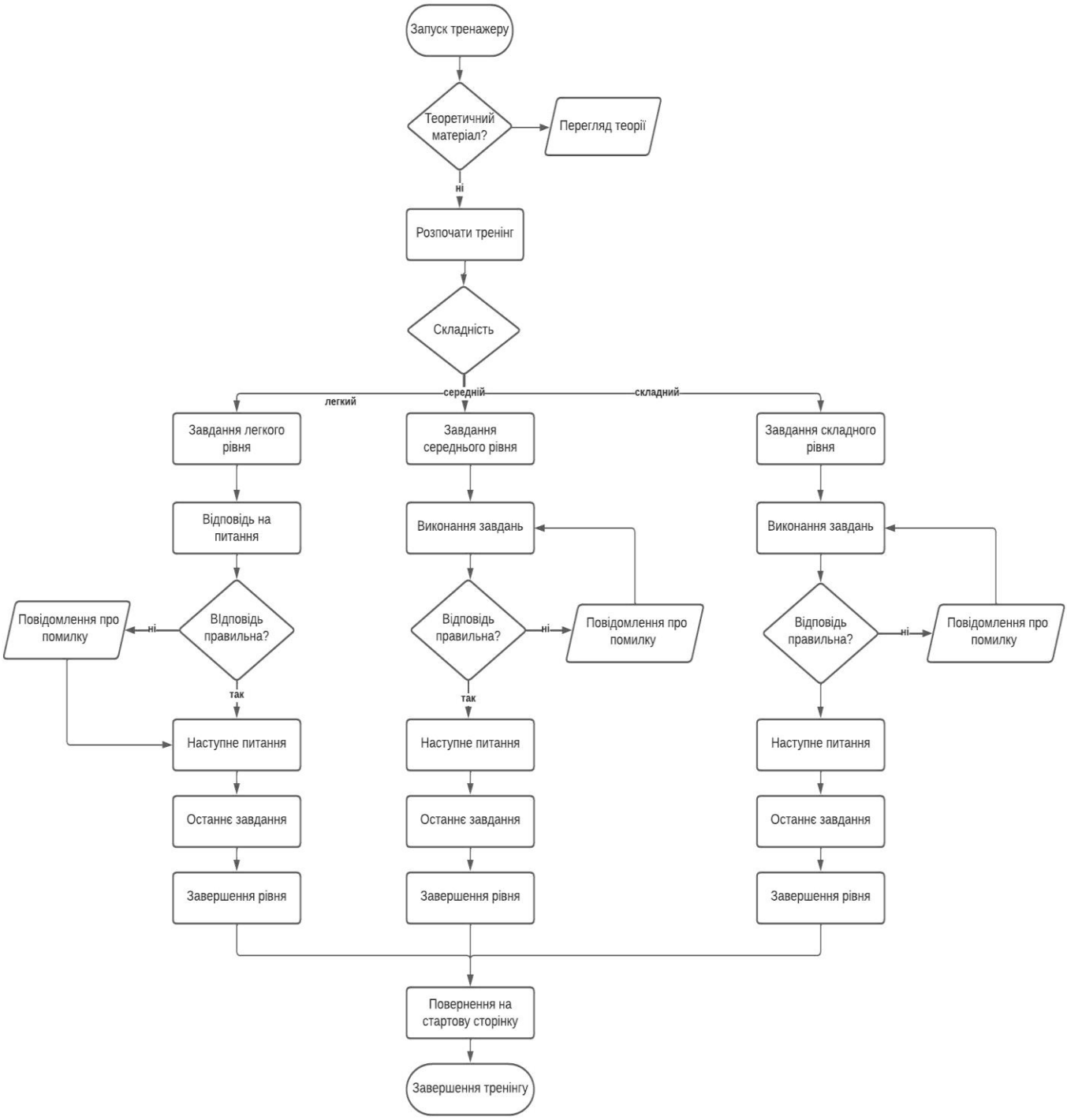

Рисунок 4.1 – Блок-схема

4.2 Опис процесу програмної реалізації

Навчальний тренажер створено за допомогою мови програмування Java та інтегрованого середовища розробки NetBeans IDE.

Для того, щоб розпочати розробку програми слід створити новий проект без головного класу ,а потім додати до нього JFrame Form[9].

Після відкриття проекту необхідно редагувати головну форму, тобто змінити її розмір та додати необхідне: кнопки, текстові елементи(рис 4.2).

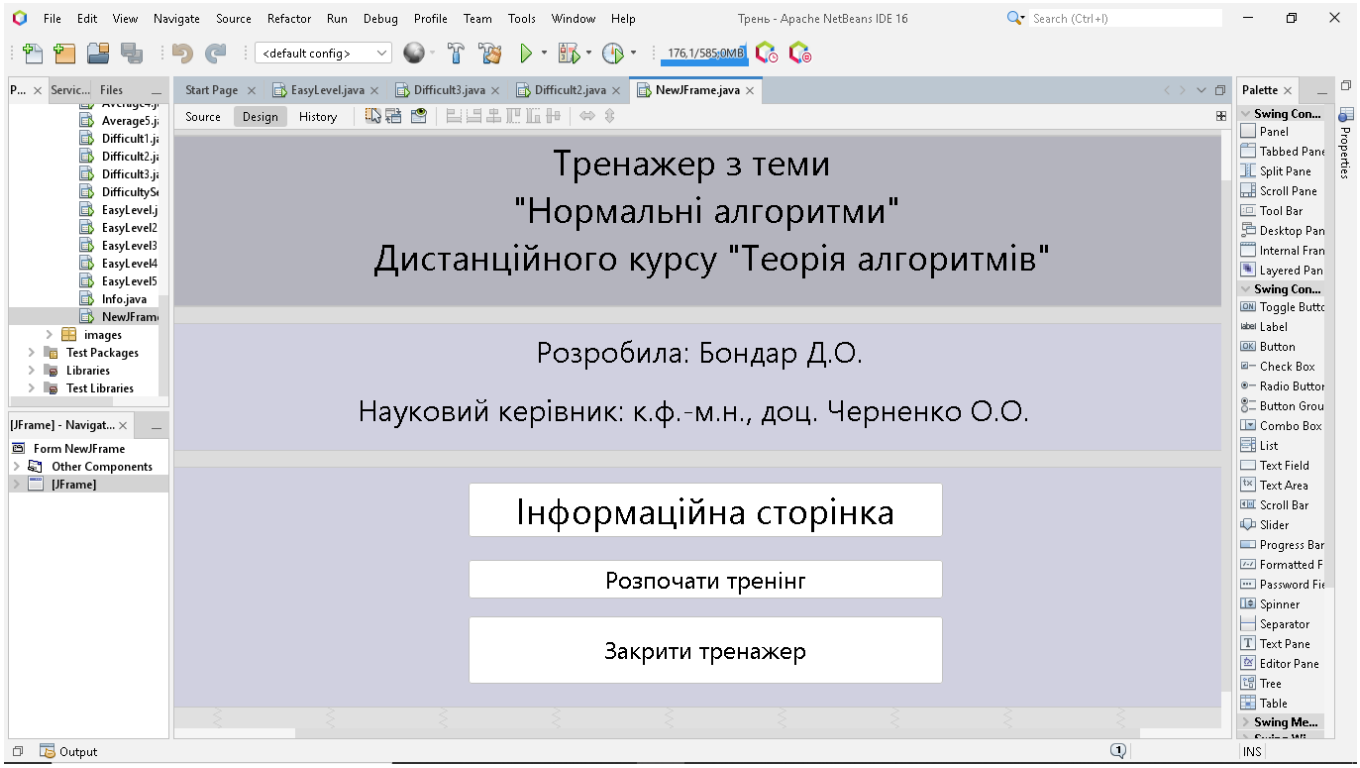

Рисунок 4.2 – Стартова сторінка

Після розробки дизайну слід зайнятись функціоналом даної сторінки. Якщо користувач бажає спочатку переглянути теоретичний матеріал, то відкривається вікно з інформацією по темі.

```
information.setVisible(true);
information.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
```

Коли користувач натискає на кнопку «Розпочати тренінг» відкривається вікно

з вибором складності тестування, а стартова сторінка стає недоступною(рис 4.3).

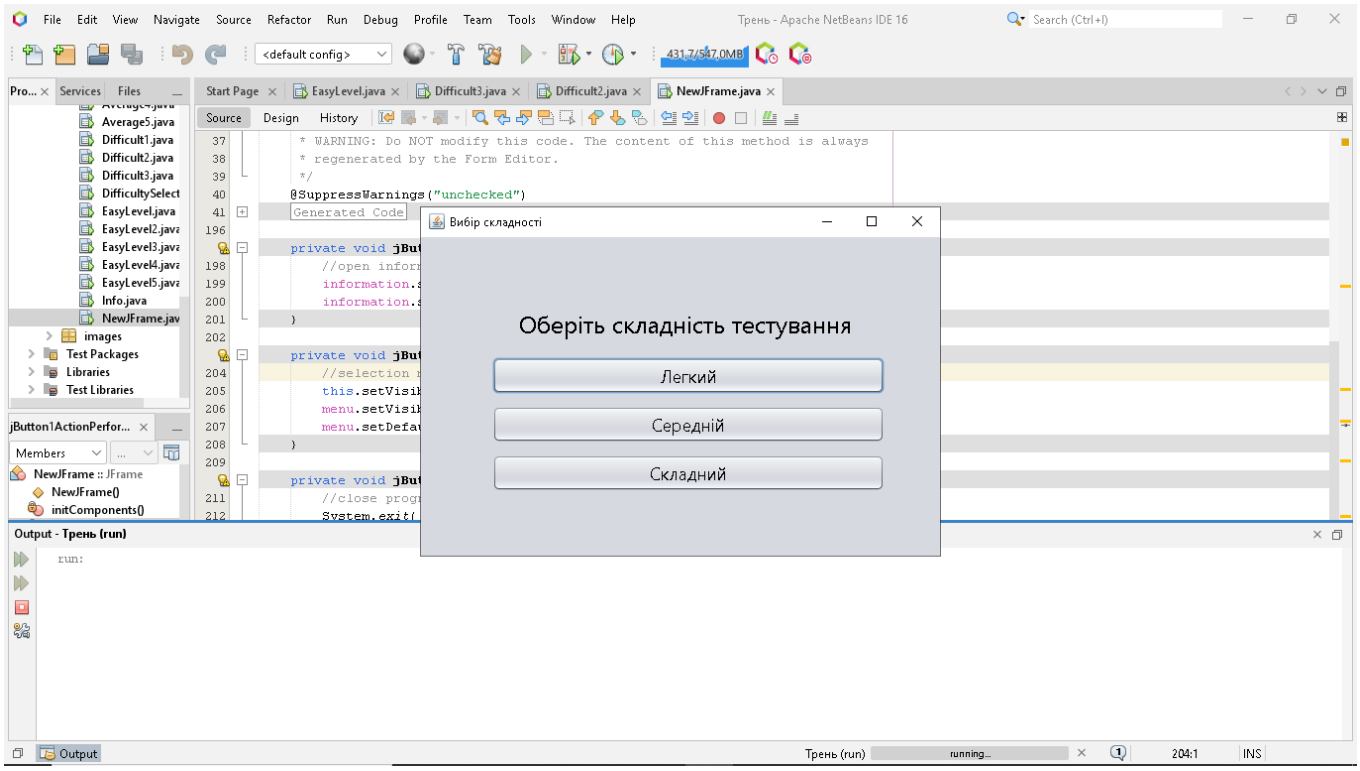

Рисунок 4.3 – Вибір складності

Наступник кроком є розробка дизайну та робота над функціоналом сторінки першого завдання легкого рівня(рис 4.4).

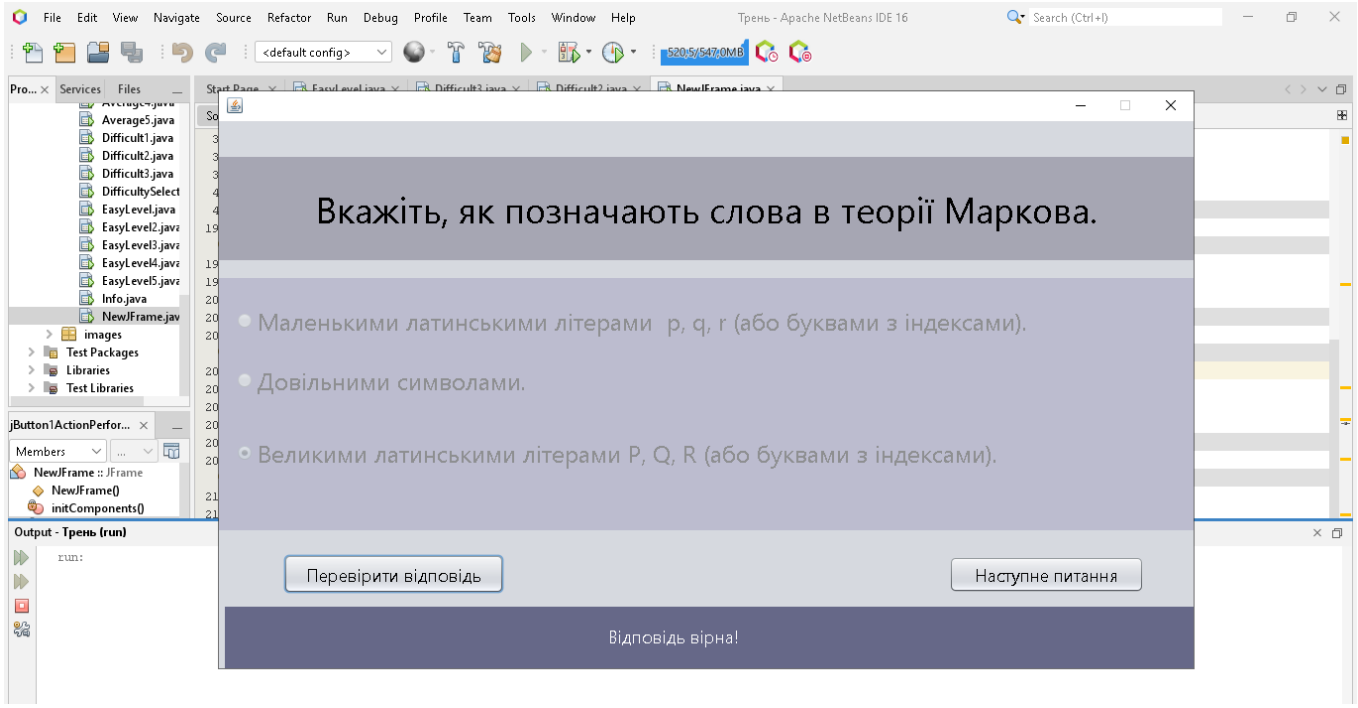

Рисунок 4.4 – Перше завдання легкого рівня

У першому, другому, третьому та п’ятому завданнях цього рівня слід обрати одну або кілька правильних відповідей. Якщо користувач натискає на правильні варінати відповідей, нижче відображується повідомлення про це, інакше – повідомлення про помилку з вказаною правильною відповіддю. У будь-якому разі розблоковується кнопка переходу до наступного завдання.

У четвертому завданні користувачеві потрібно встановити відповідність між твердженням та його визначенням. Для реалізації було використано елемент Choice та jTextField(рис 4.5).

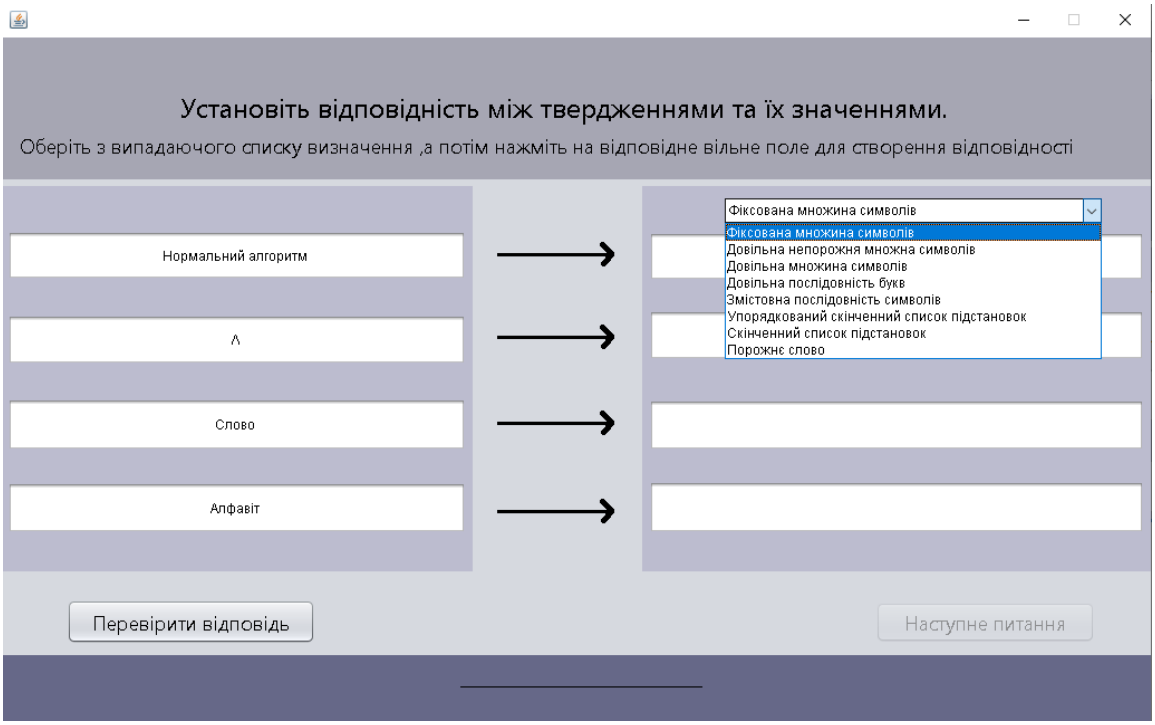

Рисунок 4.5 – Четверте завдання легкого рівня

Щоб виконати завдання потрібно обрати з випадаючого списку Choice визначення, а потім клікнути на відповідне пусте поле.

Якщо при натисненні на кнопку «Перевірити відповідь» якесь поле залишається пустим, висвічується повідомлення про це[10].

```
if(jTextField5.getText().isEmpty()||jTextField6.getText().isEmpty()||jTextField7.get
Text().isEmpty() || jTextField8.getText().isEmpty()){
    JOptionPane.showMessageDialog(null,"Оберіть відповіді");
    }else{
```

Якщо поля заповнені, то слідує перевірка. При правильних відповідях з’являється повідомлення про це, інакше – повідомлення про помилку. У будь-якому разі розблоковується кнопка переходу до наступного завдання, а поля відповідей блокуються[9].

```
if(jTextField5.getText().equals("Упорядкований скінченний список підстановок") && jTextField6.getText().equals("Порожнє слово") && jTextField7.getText().equals("Довільна послідовність букв") && jTextField8.getText().equals("Довільна непорожня множна символів")){
    answerstatus.setText("Відповідь вірна!");
    answerstatus.setForeground(Color.white);
    jButton7.setEnabled(true);
}else{
if(!jTextField5.getText().equals("Упорядкований скінченний список підстановок") || !jTextField6.getText().equals("Порожнє слово") || !jTextField7.getText().equals("Довільна послідовність букв") || !jTextField8.getText().equals("Довільна непорожня множна символів"));
     answerstatus.setText("Ви зробили помилку. Вам слід перечитати теоретичний матеріал на стартовій сторінці");
     answerstatus.setForeground(Color.black);
     jButton7.setEnabled(true);
    }}
    jTextField5.setEnabled(false);
    jTextField6.setEnabled(false);
    jTextField7.setEnabled(false);
    jTextField8.setEnabled(false);
   }
```

По завершенню легкого рівня користувач отримує повідомлення про оцінку, а потім повертається на стартову сторінку.

Наступним кроком є створення середнього рівня тренажеру. Завдання цього рівня передбачають введення з клавіатури ,тож будуть використовуватись jTextField. При відкритті першого завдання користувач має на екрані умову, одне поле для введення та одну кнопку. Якщо буде введена правильна відповідь з’явиться повідомлення про це, інакше – повідомлення з тим, як треба було відповісти. У будь-якому разі нижче з’являється ще одне поле для вводу. Перевірка здійснюється таким же чином, як і у попередньому полі, а також розблоковується кнопка переходу до наступного завдання. Відповіді можна вводити просто підряд літери чи цифри, а можна через коми(рис 4.6).

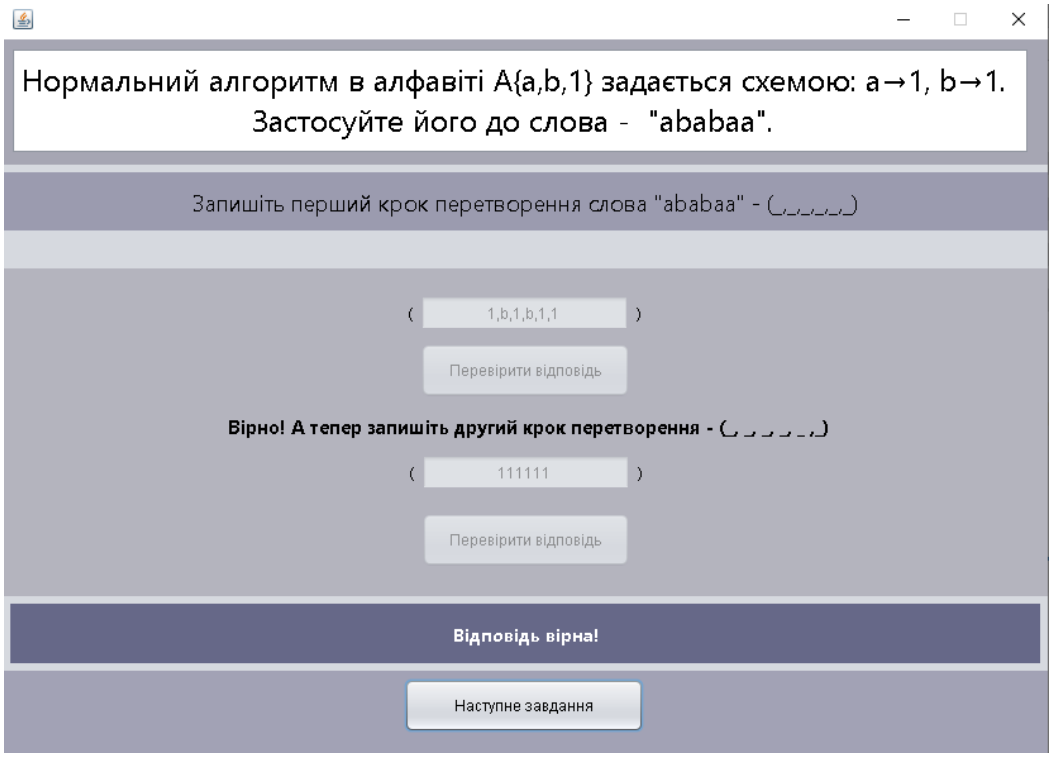

Рисунок 4.6 – Перше завдання середнього рівня

```
if(task1_1.getText().isEmpty()){
    JOptionPane.showMessageDialog(null,"Впишіть відповідь");
}else{
if(task1_1.getText().equals("1,b,1,b,1,1")  ||  task1_1.getText().equals("1b1b11")){
    answerstatus.setForeground(Color.white);
    task1_2.setText("Вірно! А тепер запишіть другий крок перетворення - (_, _, _, _, _ ,_)");
```

```
}else{
      answerstatus.setForeground(Color.black);
      task1_2.setText("Неправильно. Вірна відповідь - (1,b,1,b,1,1). Тепер запишіть другий крок перетворення - (_, _, _, _, _ ,_)");
          }

if(answer1_2.getText().isEmpty()){
     JOptionPane.showMessageDialog(null,"Впишіть відповідь");
}else{
if(answer1_2.getText().equals("1,1,1,1,1,1")||answer1_2.getText().equals("111111")){
     answerstatus.setText("Відповідь вірна!");
     answerstatus.setForeground(Color.white);
     jButton2.setEnabled(true);
}else{
    answerstatus.setText("Слід було застосувати другу підстановку. Вірна відповідь - (1,1,1,1,1,1)");
   answerstatus.setForeground(Color.black);
   jButton2.setEnabled(true);
  }
```

Друге та третє завдання аналогічні першому, змінюється лише слово, до якого необхідно застосувати алгоритм. Четверте завдання складається з кількох підетапів, спочатку потрібно кілька разів обрати одну відповідь з чотирьох, потім раз ввести з клавіатури, далі знову обрати відповідь, але тепер з двох варіантів(рис 4.7).

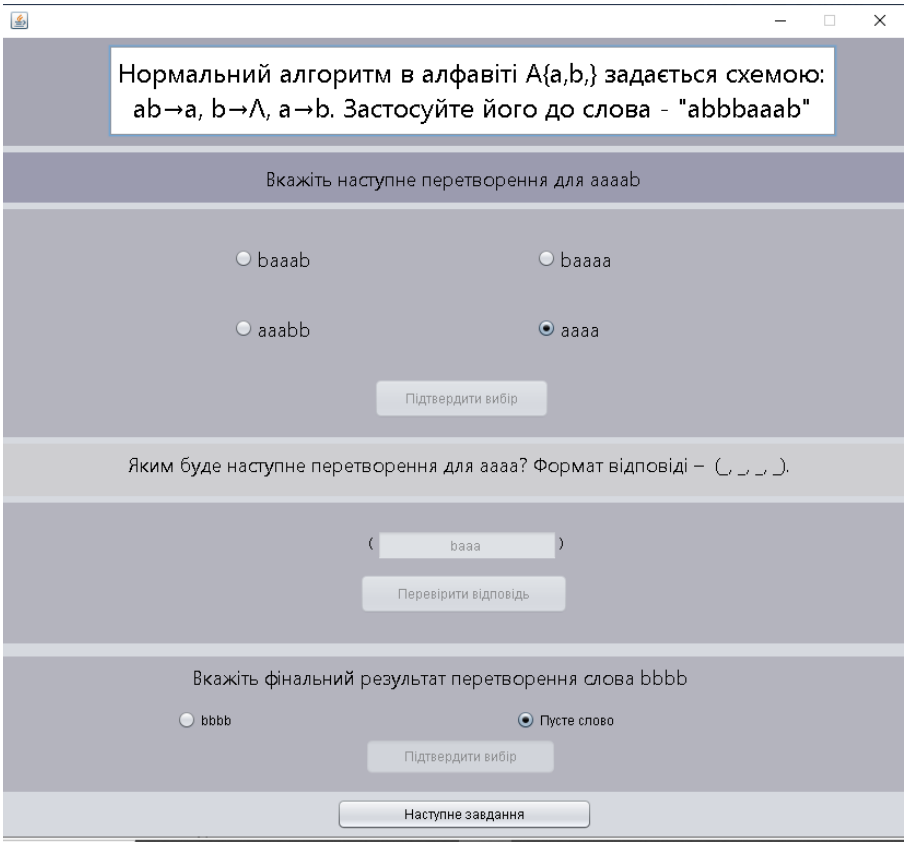

Рисунок 4.7 – Четверте завдання середнього рівня

У п’ятому завданні користувачу потрібно буде обирати потрібні підстановки з нормального алгоритму. Якщо його відповідь правильна – нижче будується ланцюжок перетворення, інакше – повідомлення про помилку та шанс обрати правильну відповідь(рис 4.8).

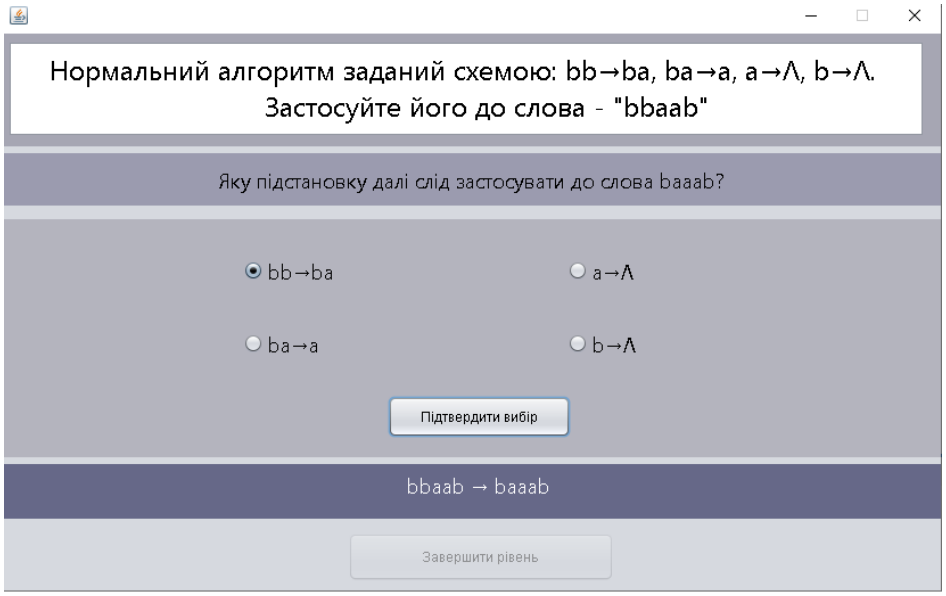

Рисунок 4.8 - П’яте завдання

Для перевірки відповідей застосовувалась конструкція switch case[9].

```
switch(i){
case 1:
if(jRadioButton1.isSelected()){
    JOptionPane.showMessageDialog(null,"Відповідь вірна!");
    i++;
   jRadioButton1.setText(answer2[0]);
   jRadioButton2.setText(answer2[1]);
   jRadioButton3.setText(answer2[2]);
   jRadioButton4.setText(answer2[3]);
   jLabel3.setText(task[0]);
   task4_2.setText(word[0]);
}else
if(jRadioButton2.isSelected()||jRadioButton3.isSelected()||jRadioButton4.isSelected()){
  JOptionPane.showMessageDialog(null,"Першою слід застосувати підстановку
bb→ba");}
          break;
         ...
case 6:
    if(jRadioButton3.isSelected()){
    JOptionPane.showMessageDialog(null,"Відповідь вірна!");
    task4_2.setText(word[5]);
    jButton3.setEnabled(true);
    jButton1.setEnabled(false);
}else if(jRadioButton3.isSelected()){
    JOptionPane.showMessageDialog(null,"Потрібно використати підстановку
b→Λ");  }
          break;     }
```

Натиснувши на кнопку «Завершити рівень» користувач отримає оцінку та буде повернений на стартову сторінку.

```
JOptionPane.showMessageDialog(null,"Ви завершили середній рівень!");
 NewJFrame start = new NewJFrame();
 start.setVisible(true);
dispose();
```

Наступним етапом є створення складного рівня тренажеру. Перше завдання включає у себе вибір кожного етапу перетворення, коли правильна підстановка уже обрана та вказана у задачі(рис 4.9).

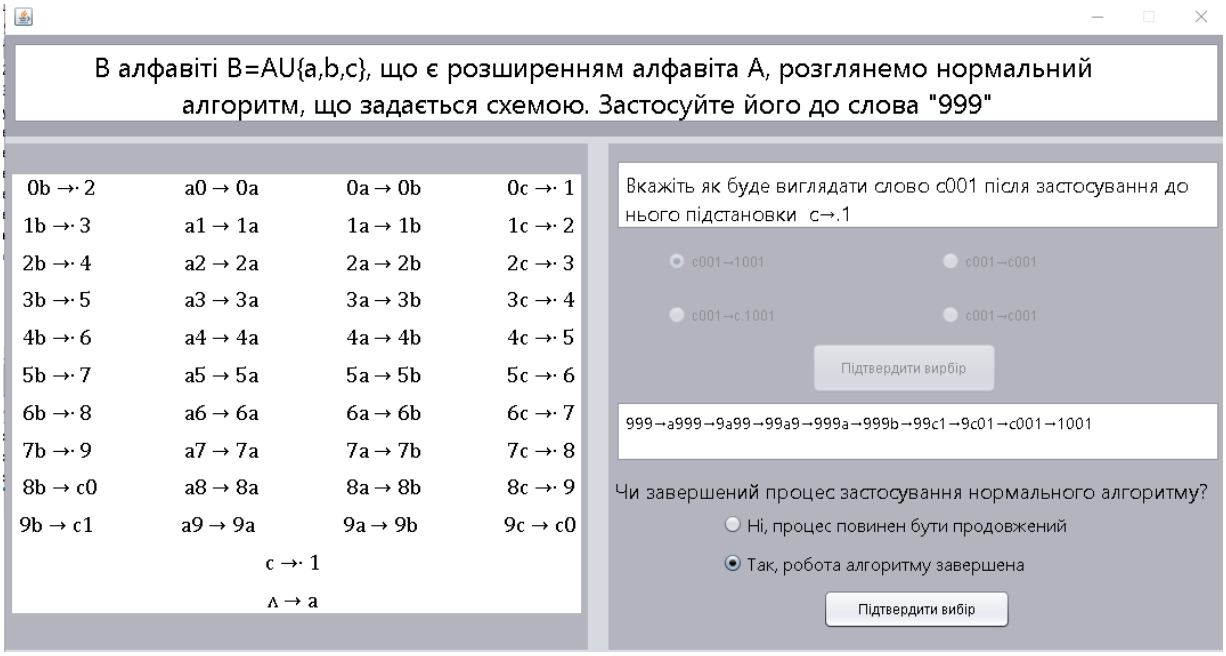

Рисунок 4.9 – Перше завдання складного рівня

Перевірка у цьому завданні також здійснюється за допомогою конструкції switch case та if-else[9].

```
switch(i){
case 1:
   if(jRadioButton1.isSelected()){
      JOptionPane.showMessageDialog(null,"Відповідь вірна!");
```

```
        i++;
        jRadioButton1.setText(answer2[0]);
        jRadioButton2.setText(answer2[1]);
        jRadioButton3.setText(answer2[2]);
        jRadioButton4.setText(answer2[3]);
       jTextArea4.setText("999→a999");
       jTextArea3.setText(task[0]);
}else
if(jRadioButton2.isSelected()||jRadioButton3.isSelected()||jRadioButton4.isSelected()){
     JOptionPane.showMessageDialog(null,"При застосуванні до слова 999 формули л→а отримаємо а999");}
          break;
          ...
case 9:
     if(jRadioButton1.isSelected()){
        JOptionPane.showMessageDialog(null,"Відповідь вірна!");
        i++;
       jTextArea4.setText("999→a999→9a99→99a9→999a→999b→99c1→9c01→c001
→1001");
        jLabel2.setVisible(true);
        }else
if(jRadioButton3.isSelected()||jRadioButton2.isSelected()||jRadioButton4.isSelected()){
       JOptionPane.showMessageDialog(null,"При застосуванні до слова с001
підстановки с→.1 отримаємо 1001");
        }break;}
```

У другому завданні складного рівня користувачу потрібно спочатку обрати підстановку, а потім як буде виглядати слово після її застосування. Форма має такий вигляд(рис 4.10):

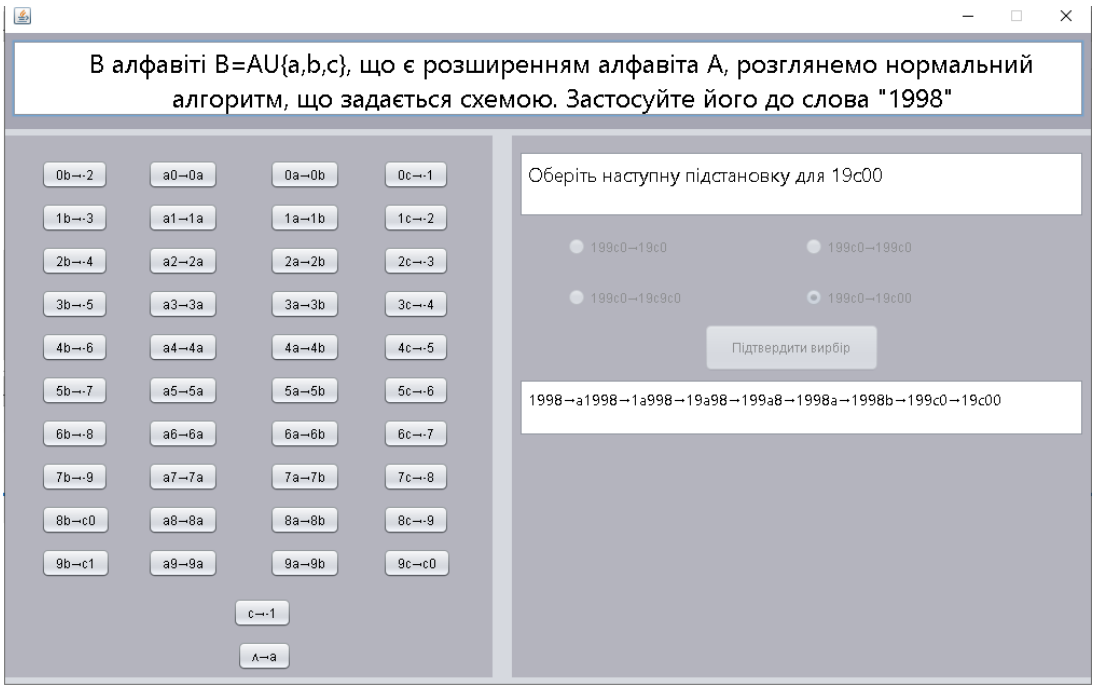

Рисунок 4.10 – Друге завдання складного рівня

Перевірка відповідей здійснюється за допомогою конструкції switch case та if-else. Вибір правильної кнопки(підстановки) перевіряється через if-else. Для усієї форми об’явлені змінні і та k. Перша для результатів підстановок, а друга для їх вибору. Змінна збільшується кожного разу, коли користувач обирає правильну відповідь. Також правильна відповідь замінює варіанти у jRadioButton, які є відповідями на результати після кожної підстановки[10].

```
if(k==2){
    k++;
    jTextArea3.setText(task[0]);
    jRadioButton1.setText(answer2[0]);
    ...
    jRadioButton4.setText(answer2[3]);
    jRadioButton1.setEnabled(true);
    ...
   jRadioButton4.setEnabled(true);
   jButton43.setEnabled(true);
```

```
}else{
    JOptionPane.showMessageDialog(null,"Подумайте ще!");
}
```

У останньому завданні складного рівня спочатку потрібно створити панель вибору необхідних підстановок, щоб отримати нормальний алгоритм для подальшого розв'язку. Початковий вигляд форми має такий вигляд (рис 4.11):

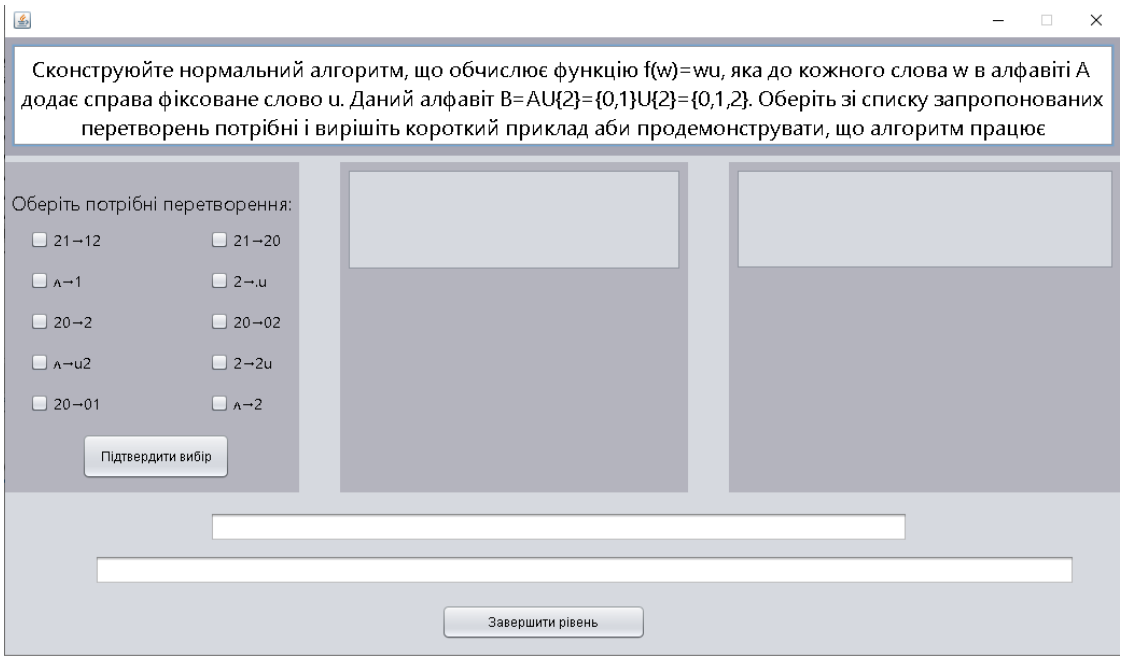

Рисунок 4.11 – Початковий вигляд третього завдання

Користувач обирає з запропонованих перетворень ,створених на основі алфавіту, потрібні. Якщо його відповідь неправильна, то він отримає повідомлення про це та шанс переобрати. Інакше – нижче у вільному полі з`явиться алгоритм та на середньому полі наступна частина завдання.

```
if(jCheckBox18.isSelected()&&jCheckBox11.isSelected()&&jCheckBox20.isSelected()&
& jCheckBox17.isSelected()){
   JOptionPane.showMessageDialog(null,"Вірно, переходимо до наступної частини
завдання");
   jCheckBox11.setEnabled(false);
   ...
```

```
 jCheckBox20.setEnabled(false);
 jButton2.setEnabled(false);
jTextArea1.setVisible(true);
jRadioButton1.setVisible(true);
 ...
jRadioButton4.setVisible(true);
jButton1.setVisible(true);
jTextField3.setText("Алгоритм: 1) 20→02; 2) 21→12; 3)  2→.u; 4)  л→2");
}else{
JOptionPane.showMessageDialog(null,"Подумайте ще!");
 }
```

Далі користувачу потрібно по черзі обирати підстановки та вводити самостійно, як буде виглядати слово після застосування перетворень, також нижче на кожному кроці буде відображатись ланцюжок перетворень(рис 4.12).

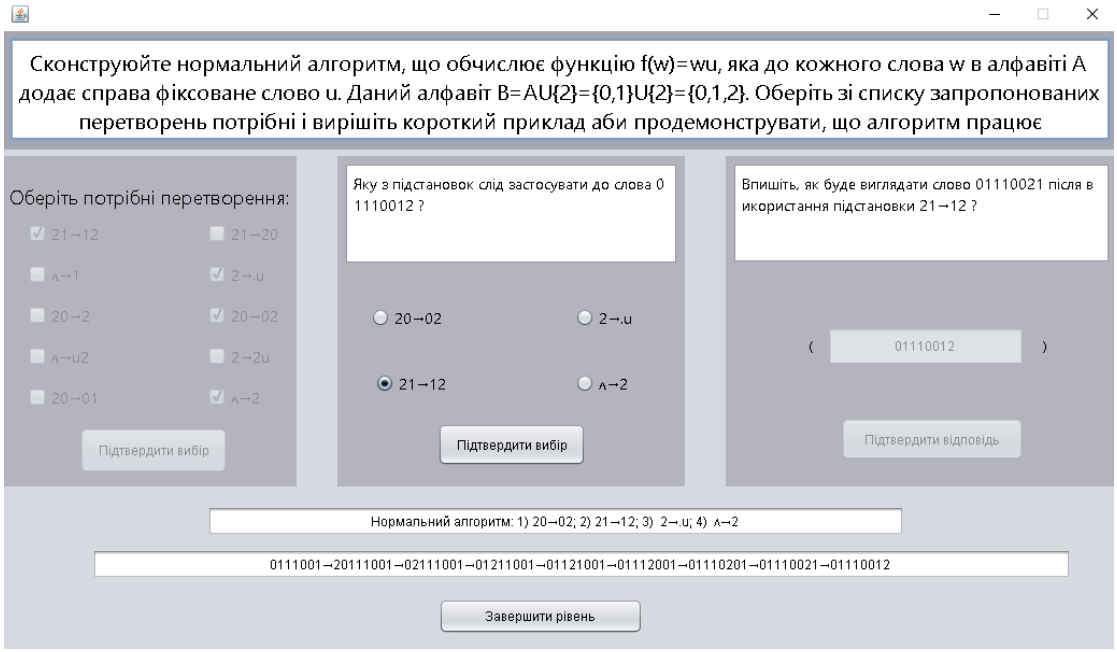

Рисунок 4.12 – Третє завдання складного рівня

Після завершення перетворення стане доступною кнопка «Завершити рівень», яка поверне користувача на стартову сторінку.

4.3 Необхідна користувачу інструкція програми

На головній сторінці відображається тема тренажера, виконавець, науковий керівник та кнопки(рис 4.13).

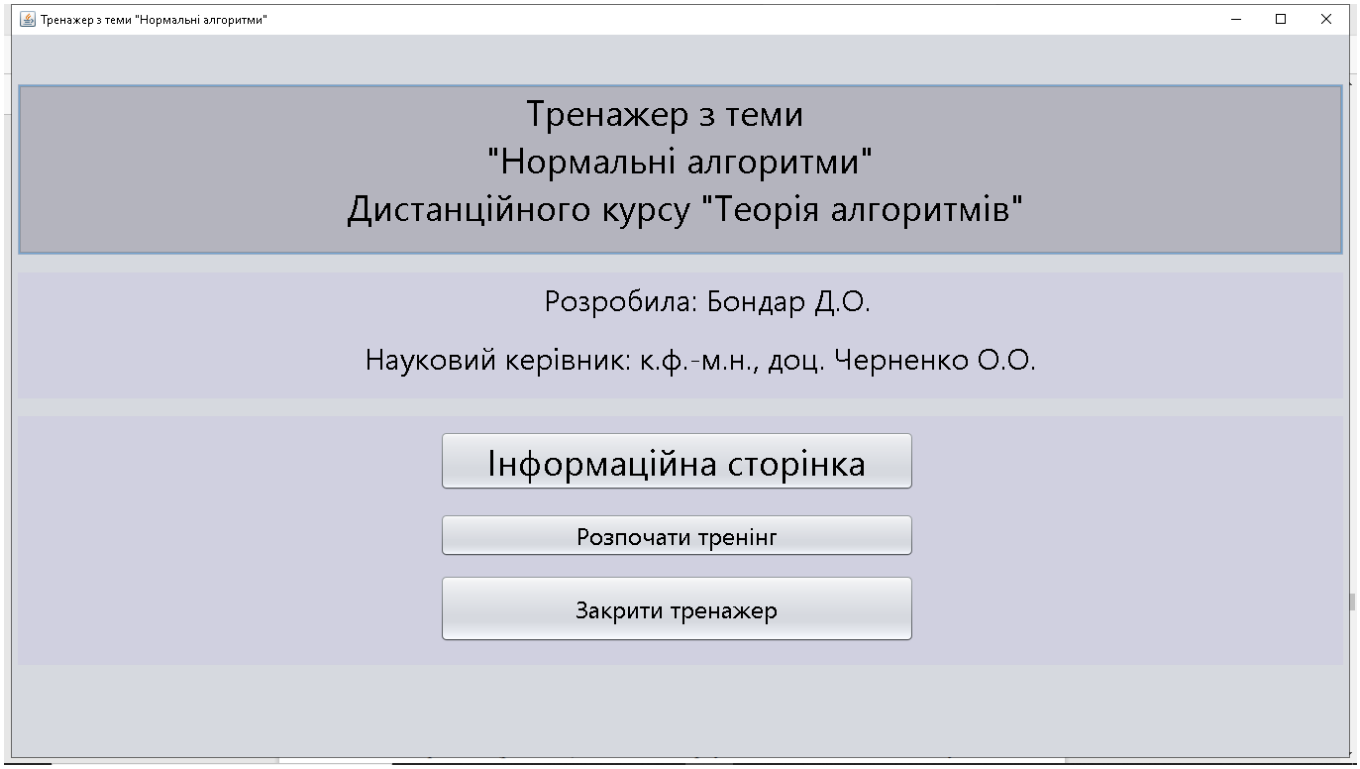

Рисунок 4.13 – «Стартова сторінка»

Далі користувачу потрібно обрати «Інформаційна сторінка» для перегляду теорії або «Розпочати тренінг». Другий варіант відкриває вікно з трьома кнопками, кожна з яких відкриває завдання певної вказаної складності.

У тестах на кожному кроці виводяться завдання та варіанти відповідей або потрібні поля. Якщо допущена помилка - з’явиться повідомлення про це та підказка (рис 4.14).

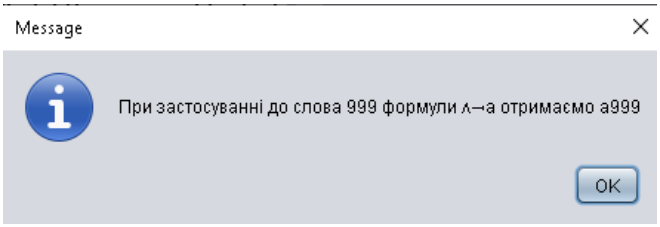

Рисунок 4.14 – «Помилка у завданні складного рівня»

Далі виведеться наступне завдання і варіанти відповідей або поля для вводу.

На останньому кроці кожного рівня висвічується повідомлення про завершення рівня, оцінка, а користувач повертається на стартову сторінку, де може знову переглянути теорію чи заново виконати тест.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи був повторно розглянутий теоретичний матеріал з теми «Нормальні алгоритми», який був наданий у лекційному матеріалі, було оглянуто схожі по темі роботи, а також створено тренажер до курсу «Теорія алгоритмів». .

Результати роботи:

1) Виконаний інформаційний огляд попередніх робіт;
2) Виділені їх позитивні та негативні сторони;
3) Обрано завдання з практичних робіт дистанційного курсу;
4) Розроблено алгоритм роботи тренажеру;
5) Створено блок-схему;
6) Розроблено тренажер до теми «Нормальні алгоритми» дистанційного навчального курсу «Теорія алгоритмів»;

Робота пройшла апробацію на XLVI Міжнародній студентській конференції «Актуальні питання розвитку науки та забезпечення якості освіти у XXI столітті» (за підсумками науково-дослідних робіт студентів 2022 рік) 25 квітня 2023 року.

За результатами кваліфікаційної роботи підготовлено наукову статтю Development of simulator software on the topic "Normal algorithms" of the distance learning course "Theory of Algorithms" до Центральноукраїнського наукового вісника. Тренажер передано до дистанційного відділу ПУЕТ та імплементовано у відповідний дистанційний курс, про що свідчить акт впровадження.

СПИСОК ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Черненко, О.О., Чілікіна, Т.В., Ольховська, О.В. Розробка та використання навчальних тренажерів при підготовці фахівців напряму «Комп`ютерні науки». International scientific and practical conference ``Mathematics, physics, mechanics, astronomy, computer science and cybernetics: issues of productive interaction``: conference proceedings, Yuly 9-10. 2021. Wloclawek, Republic of Poland: ``Baltija Publishing``, 2021. С. 55-59.
2. Introduction To Computer Simulations For Integrated Stem College Education. WSPC, 2019. 234 p.
3. Бардаченко С.Р. Електронний тренажер для дистанційного курсу «Теорія алгоритів» на тему: «Машини Тьюрінга» [Електронний ресурс]: – Режим доступу: http://www2.el.puet.edu.ua/st/mod/resource/view.php?id=98521
4. Алексов С.В. Електронний тренажер для дистанційного курсу «Теорія програмування» на тему: «Дерево розбору» [Електронний ресурс]: – Режим доступу: http://www2.el.puet.edu.ua/st/mod/resource/view.php?id=158345
5. Гребенюк Д.С. Електронний тренажер для дистанційного курсу «Теорія алгоритмів» на тему: «Нормальні алгоритми» [Електронний ресурс]: – Режим доступу: http://www2.el.puet.edu.ua/st/mod/resource/view.php?id=98541
6. Бородкіна І.Л. Теорія алгоритмів: посібник для студентів вищих навчальних закладів / Бородкіна І.Л. – ТОВ «Видавництво "Центр навчальної літератури"», 2019. – 184 с.
7. Дистанційний курс ПУЕТ [Електронний ресурс]: Теорія алгоритмів, практичне завдання 3 – Режим доступу: http://www2.el.puet.edu.ua/st/mod/assign/view.php?id=98538
8. Коваль В.С. Алгоритми і структури даних: навчальний посібник/ В.С. Коваль, П.Р. Струбицький. – Тернопіль: ФОП Шпак В. Б. – 2017. – 74 с.
9. Schildt H. Java: A Beginner's Guide, Eighth Edition. McGraw-Hill Education, 2018. 720 p.
10. Bloch J. Effective Java. Addison-Wesley Professional, 2018. 412 p.

11. Волкова, Н.П. Інтерактивні технології навчання у вищій школі: навчально-методичний посібник. Дніпро: Університет імені Альфреда Нобеля, 2018. 360 с.
12. Доценко, Н. Застосування навчальних комп'ютерних інтерактивних тренажерів здобувачами вищої освіти інженерних спеціальностей в умовах інформаційно-освітнього середовища. Педагогічні науки: теорія, історія, інноваційні технології, 2018. № 2(76). С. 118–128. URL: https://doi.org/10.24139/2312-5993/2018.02/118-128 (дата звернення: 18.05.2022).
13. Virk R. The Simulation Hypothesis: An MIT Computer Scientist Shows Why AI, Quantum Physics and Eastern Mystics All Agree We Are In a Video Game. Bayview Books, 2019. 330 p.
14. Ольховська, О.В., Собіборець, О.Ю. Програмна реалізація елементів тренажеру з теми «Системи числення, арифметичні операції в різних системах числення» дисципліни «Архітектура обчислювальних систем». Новітні інформаційно-комунікаційні технології в освіті: матеріали VII Всеукраїнської науково-практичної Інтернет-конференції молодих учених та студентів (Полтава, 24-25 листопада 2021 р.). Полтава: ПП "Астрая", 2021. С. 141-142.
15. Bhakat Rohit Development of software simulator for the cut-off method of the distance learning course Elements of combinatorial optimazation. Комп'ютерні науки та інформаційні технології (КНІТ-2022): матеріали науково-практичного семінару. Випуск 1. Полтава: кафедра КНІТ ПУЕТ, 2022. С.36-38.
16. Ghrmida, S. M., Harkusha, S. V., Koshova, O. P., Orikhivska, O. G. Some Peculiarities of Development of Desktop Application «Using Array in Java». Збірник наукових статей магістрів. Полтава: ПУЕТ, 2022. С. 105-109.
17. Яновський, А. Інформаційно-освітнє середовище в умовах дистанційного навчання. Humanities science current issues. 2020, Т. 4, № 30. С. 310-315. URL: https://doi.org/10.24919/2308-4863.4/30.212627 (дата звернення: 19.10.2022).