ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка
до бакалаврської роботи
на ступінь вищої освіти бакалавр

на тему: «**РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ДЛЯ НАВЧАННЯ З КУРСУ “АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ” МОВОЮ C#**»

Виконала: студентка 4 курсу, групи ПД–42
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

_____ Логункова А.І.
(прізвище та ініціали)

Керівник _____ Негоденко О.В.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Нормоконтроль _____
(прізвище та ініціали)

Київ –2021

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти -«Бакалавр»
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
Негоденко О.В.
"____" _____2021 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТА

ЛОГУНКОВА АНАСТАСІЯ ІГОРІВНА
(прізвище, ім'я, по батькові)

1. Тема роботи: «Розробка програмного додатку для навчання з курсу "Алгоритми та структури даних мовою c#"»

Керівник роботи: Негоденко О.В., завідувач кафедри ІПЗ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «12» березня 2021 року №65.
2. Строк подання студентом роботи «01» червня 2021 року
3. Вхідні дані до роботи

У дослідженні застосовується системний, структурно-функціональний методи аналізу, метод математичного моделювання, принципи побудови програмних і апаратних комп'ютерних засобів.
В якості інструментального середовища проектування використовується Rational Software Architect.
Для опису моделі використовується мова UML.
При розробці даного програмного продукту використовувалися наступні програмні засоби: C#; SQL; Блокнот._____

4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).
4.1 Описати і обґрунтувати вибраний алгоритм щодо розв'язання задачі створення програмного додатку для навчання з курсу «Алгоритми та структури даних».
4.2 Провести опис-блок схеми вибраного алгоритму.
4.3 Здійснити вибір технічного та програмного забезпечення.
4.4 Розробити проектні рішення по системі та її частинам.
4.5 Виконати проектування програмного додатку для навчання з курсу «Алгоритми та структури даних».
4.6 Здійснити опис розробленого програмного додатку.
4.7 Розробити керівництво користувача.
4.8 Провести тестування розробленого програмного додатку.

5. Перелік демонстраційного матеріалу (назва основних слайдів)
1. Титульний слайд
2. Мета, об'єкт та предмет дослідження
3. Актуальність дослідження
4. Завдання дослідження
5. Дослідження аналогів та прототипів
6. Постановка завдань дослідження
7. Алгоритм додатка(обробка подій екранних форм програми)
8. Структурна модель системи тестування
9. Діаграма класів
10. Розроблений додаток
11. Висновки

6. Дата видачі завдання «19» квітня 2021

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури	17.09.2020	
2	Дослідження актуальності теми	25.09.2020	
3	Розгляд аналогів	15.10.2020	
4	Проектування логіки проекту	18.11.2020	
5	Конструювання основних сцен проекту	07.12.2020	
6	Реалізація проекту	10.02.2020	
7	Вступ, висновки, реферат	20.02.2021	
8	Розробка обов‘язкових демонстраційних матеріалів	21.04.2021	
9	Попередній захист роботи		
10	Здача роботи в деканат	01.06.2021	

Студент _____ Логункова А. І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Негоденко О. В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина бакалаврської роботи 71 с., 20 рис., 1 табл., 30 джерел.

Ключеві слова: VISUAL STUDIO, SQL, ТЕСТУВАННЯ, НАВЧАННЯ, АЛГОРИТМИ, БАЗА ДАНИХ, СТРУКТУРА, C#

Об'єкт дослідження – підвищення ефективності вивчення та засвоєння навчального матеріалу з курсу «Алгоритми та структури даних».

Предмет дослідження – додаток для системи навчання та перевірки знань з курсу «Алгоритми та структури даних».

Мета роботи – розробити програмний додаток для навчання з курсу «Алгоритми та структури даних».

Методи дослідження – методи аналізу, метод математичного моделювання, методи теорії інформації, методи розробки програмного забезпечення, методи тестування, валідації та верифікації програмного забезпечення.

Наукова новизна даної роботи полягає в наступному:

1. Розроблено алгоритм для покращення системи навчання та перевірки знань з курсу «Алгоритми та структури даних»
2. Встановлено, що використання мови C# та середовища розробки VISUAL STUDIO - це вдале рішення для досягнення поставленої мети.
3. Показано, що мова C# зручна у використанні та надає можливість ефективно розробити програмне забезпечення для навчання та перевірки знань з курсу «Алгоритми та структури даних».
4. На основі результатів виконаних досліджень розроблено додаток для системи навчання та перевірки знань з курсу «Алгоритми та структури даних»

Використання розробленої моделі дасть можливість підвищити роботу освітніх установ, а тим самим знизить кількість часу витраченого викладачем для тестування (опитування) студентів та підвищить якість роботи викладачів. Автором розроблено програмний додаток для навчання з курсу «Алгоритми та структури даних», щодо завдання у межах дипломної роботи. В якості інструментального середовища проектування використовується Rational Software Architect. Для опису моделі використовується мова UML. При розробці даного програмного продукту використовувалися наступні програмні засоби: C#; SQL; Блокнот.

Матеріали дипломного проекту можуть сприяти зменшенню трудомісткості роботи викладачів, автоматизації їх роботи при контролі та систематизації знань умінь і навичок з навчальної дисципліни.

Побудова єдиного інформаційного середовища є головним завданням, яке в рамках розвитку процесів інформатизації вирішує кожний.

Галузь використання – завдяки вільному доступу, застосунок може використовувати будь-яка людина, котра бажає вивчати основи алгоритми та структури даних. Результати роботи можуть бути використані при розробці, встановленні та налаштуванні програмного комплексу для вивчення матеріалів з курсу «Алгоритми та структури даних».

ЗМІСТ

СПИСОК ВИКОРИСТАНИХ СКОРОЧЕНЬ

БД	база даних
ЕОМ	електронно-обчислювальна машина
ОС	операційна система
ПЗ	програмне забезпечення
ПК	персональний комп’ютер
ППП	пакети прикладних програм
СКТ	системи комп'ютерного тестування

ВСТУП

Актуальність дослідження. Одним із завдань повсякденної викладацької праці є необхідність здійснювати контроль знань учнів. Форми контролю, що застосовуються викладачами, дуже різноманітні, але найбільш часто використовуються письмове або усне опитування. На жаль, ці форми не позбавлені недоліків. При проведенні усного опитування – це відносно велика витрата часу уроку при невеликій кількості оцінок, що виставляються, при проведенні письмових робіт кількість оцінок зростає, але багато часу йде на перевірку.

Тестування як ефективний спосіб перевірки знань знаходить все більше застосування. Одним з основних і безперечних його достоїнств є мінімум часових витрат на отримання надійних підсумків контролю. При тестуванні використовують як паперові, так і електронні варіанти. Останні особливо привабливі, тому що дозволяють отримати результати практично відразу по завершенні тесту.

Тестування в педагогіці виконує три основні взаємопов'язані функції: діагностичну, навчальну і виховну:

Діагностична функція полягає у виявленні рівня знань, умінь, навичок учня. Це основна, і найочевидніша функція тестування. За об'єктивності, широти і швидкості діагностування, тестування перевершує всі інші форми педагогічного контролю;

Навчальна функція тестування полягає у мотивуванні учня до активізації роботи по засвоєнню навчального матеріалу. Для посилення навчальної функції тестування, можуть бути використані додаткові заходи стимулювання студентів, такі, як роздача викладачем примірного переліку питань для самостійної підготовки, наявність в самому тесті навідних запитань і підказок, спільний розбір результатів тесту.

Виховна функція проявляється в періодичності й неминучості тестового контролю. Це дисциплінує, організовує і спрямовує діяльність

учнів, допомагає виявити і усунути прогалини в знаннях, формує прагнення розвинути свої здібності.

Тестування – більш справедливий метод, воно ставить всіх учнів в рівні умови, як у процесі контролю, так і в процесі оцінки, практично, виключаючи суб'єктивізм викладача.

Слід зазначити, що саме тестування поступово стає основною формою складання іспитів та виходить на перший план серед усіх рівнів навчання. Тому актуальність теми даного дослідження не викликає сумнівів.

Мета та завдання дослідження. Метою даної дипломної роботи виступає розробка програмного додатку для навчання та перевірки знань з курсу «Алгоритми та структури даних». Для досягнення поставленої мети у роботі необхідно виконати низку завдань:

– здійснити дослідження об’єкта та виконати обґрунтування необхідності створення програмного додатку для навчання з курсу «Алгоритми та структури даних»;

– описати і обґрунтувати вибраний алгоритм щодо розв’язання задачі створення програмного додатку для навчання з курсу «Алгоритми та структури даних»;

– провести опис-блок схеми вибраного алгоритму створення програмного додатку для навчання з курсу «Алгоритми та структури даних»; виконати вибір і обґрунтування вибору алгоритмічної мови програмування;

– здійснити вибір технічного та програмного забезпечення;

– розробити проектні рішення по системі та її частинам;

– виконати проектування програмного додатку для навчання з курсу «Алгоритми та структури даних»;

– здійснити опис розробленого програмного додатку для навчання з курсу «Алгоритми та структури даних»;

– розробити керівництво користувача;

– провести тестування розробленого програмного додатку для навчання з курсу «Алгоритми та структури даних».

Об’єкт та предмет дослідження. Об'єкт дослідження – підвищення ефективності вивчення та засвоєння навчального матеріалу з курсу «Алгоритми та структури даних».

Предмет дослідження – додаток для системи навчання та перевірки знань з курсу «Алгоритми та структури даних».

Методи дослідження – методи аналізу, метод математичного моделювання, методи теорії інформації, методи розробки програмного забезпечення, методи тестування, валідації та верифікації програмного забезпечення.

Практичне значення одержаних результатів. Використання розробленої моделі дасть можливість підвищити роботу закладів освіти, а тим самим знизить кількість часу витраченого викладачем для тестування (опитування) студентів та підвищить якість роботи викладачів.

Особистий внесок. Автором розроблено програмний додаток для навчання з курсу «Алгоритми та структури даних», щодо завдання у межах дипломної роботи. В якості інструментального середовища проектування використовується Rational Software Architect. Для опису моделі використовується мова UML. При розробці даного програмного продукту використовувалися наступні програмні засоби: C#; SQL; Блокнот.

Результати роботи. Матеріали дипломного проекту можуть сприяти зменшенню трудомісткості роботи викладачів, автоматизації їх роботи при контролі та систематизації знань умінь і навичок з навчальної дисципліни.

Побудова єдиного інформаційного середовища є головним завданням, яке в рамках розвитку процесів інформатизації вирішує кожний заклад освіти.

Структура роботи. Структуру роботи складають: вступ, три розділи, висновки, список використаних джерел, додаток. Загальний обсяг роботи становить 71 сторінка.

РОЗДІЛ 1.
КОНЦЕПТУАЛЬНА МОДЕЛЬ ДАНИХ

1.1 Обґрунтування розробки

В даний час використовуються в основному бланково-комп'ютерна та комп'ютерна (автоматизована) технології навчання та перевірки знань, тестування. У першому випадку тестований отримує тест (наприклад завдання з варіантами відповідей) і опитувальний лист (бланк). У опитувальному аркуші за кожним завданням він заштриховуєш (ставить мітку) той квадрат, який відповідає порядковому номеру правильного варіанту відповіді. Дані опитувального листа за допомогою сканера і відповідної програми зчитуються (автоматично вводяться) в ЕОМ, яка виробляє їх обробку і відповідно до визначених критеріїв виставляє оцінку, друкує результати. У другому випадку тестований читає завдання тесту на моніторі ЕОМ, вводить відповіді з клавіатури, вказує клацаннями миші, результати виводяться на екран і друк.

Бланково-комп'ютерна технологія використовується, наприклад, для державного централізованого тестування випускників шкіл, гімназій, ліцеїв, вищих навчальних закладів. Система комп'ютерного тестування для вступників до вузів заснована на використанні Інтернет-технологій.

Комп'ютерна технологія тестування використовується в вузах та інших навчальних закладах для поточного контролю знань, прийому заліків та іспитів.

Системи комп'ютерного тестування (СКТ) забезпечують розробку дисциплінарних, міждисциплінарних, кваліфікаційних та інших програм-тестів в різних предметних областях з використанням різних форм завдань. Для перевірки знань з технічних та інших дисциплін застосовуються графічні, мультимедійні завдання [1].

СКТ дозволяють автоматизувати не тільки оцінку теоретичних знань, а й умінь вирішувати розрахункові практичні завдання. Наприклад, може бути реалізована підготовка завдань з генерованими змінними. У задачі вказуються межі змін значень змінних і формула (формули) розрахунку. При тестуванні учні за допомогою вбудованого калькулятора вирішуватимуть однотипну задачу з різними значеннями змінних (генерованих в заданих межах), що виключає підбір правильної відповіді, списування.

СКТ включають дві основні програми – конструктор тестів (для викладачів) і програму контролю знань (для студентів). За допомогою конструктора тестів виробляється формування і запис на жорсткий (гнучкий) магнітний диск сценарію тесту. Конструктор тесту дозволяє змінювати сценарій, у тому числі коректувати параметри, редагувати, видаляти і додавати завдання. Введення завдань та інших елементів сценарію проводиться за допомогою вбудованого або іншого текстового, графічного редактора. При тестуванні сценарій тесту зчитується програмою контролю знань, завдання (кадри) пред'являються того, кого навчають на моніторі ЕОМ. Він вводить відповіді, які порівнюються з еталонними, і виставляється оцінка. Результати тестування виводяться на монітор ЕОМ і друк.

При контролі знань формується і зберігається в пам'яті протокол опитування кожного учня, а також групова відомість. Протокол опитування зазвичай включає найменування дисципліни (тесту), дату, ПІБ учня, номери завдань, відповіді (номери відповідей, слова), оцінки за кожним завданням (вірно, невірно). Протокол може бути роздрукований, наприклад для аналізу невірних відповідей.

1.2 Дослідження аналогів та прототипів

Нескладно помітити, що тести на різних етапах навчання можуть використовувати якісь загальні елементи (питання і відповідь, наприклад), а інші елементи (такі як коментар) додаються опціонально в залежності від

конкретного завдання тесту в навчальному процесі. Значить, слід організувати розробку тестів таким чином, щоб уникаючи дублювання, створити безліч різних додатків на базі деяких простих будівельних конструкцій – питання, відповідь, коментар, блок, ланцюжок, ключ [4].

Зовсім не слід готувати багато різних тестів з різними варіантами питань (для контролю залишкових знань, для самотестування, для підсумкового контролю тощо). Ефективніше розробити банк питань. Будь-яке питання може містити як власний список відповідей, коментарів або посилань, так і використовувати загальні будівельні конструкції, що полегшує створення варіативних однотипних питань, особливо в рамках однієї тематичної області.

Відповідь має бути формально визначена шаблоном. Це визначає і форму, з якою тестований буде взаємодіяти. Вибір шаблонів обмежений рамками HTML. Детальніше приклади форм обговорюються в практикумі з конкретними прикладами для різних типів питань. Тут же зупинимося на найбільш популярних формах: вибір відповіді зі списку (вибір єдиного варіанта і адитивний вибір декількох елементів списку)[5].

Виділимо в таких питаннях спільну конструкцію закритий список усіх можливих відповідей (у т.ч. неправильних). Такий список може бути загальним для декількох питань одночасно, що підвищить технологічність розробки, знизить витрати часу і розробника, і тьютора, а, головне, виявиться більш легким для студента (з точки зору техніки відповіді, а не з точки зору змісту, звичайно) [6].

Крім списку всіх можливих відповідей, кожному питанню можна зіставити безліч правильних відповідей для автоматизації контролю.

Закритий список відповідей може бути загальним для декількох питань тесту, а ось безліч правильних відповідей доведеться вказати індивідуально для кожного з питань.

Спрощено, будемо вважати, що правильна відповідь одна з деякого закритого списку. Можемо дозволити програмі тестування довільно

змінювати послідовність відповідей у списку, приховуючи правильну відповідь на різних місцях у списку. Більше того, список всіх відповідей може бути кілька надлишковий і тоді правильна відповідь буде з'являтися серед різних кандидатів, а не серед одних і тих же [7].

Такий прийом не дозволить механічно запам'ятовувати вірні відповіді (за номерами, наприклад) при самоконтролі. І, отже, дозволить одне і те ж питання використовувати в різних формах контролю та для різних студентів.

Важливе зауваження розробникові не слід ідентифікувати питання або відповіді, це робить програма тестування. Причому, на екрані порядок запитань та відповідей може бути різним і нумерація буде різною, а при контролі тьютором результатів тестування всі дані будуть узагальнені в порівнянній формі [8].

Однак, така виверт вельми прозора. Розробнику доведеться приділити увагу угрупованню не тільки відповідей, а й самих питань.

Множинний вибір (питання в закритій формі). Студенту задається питання і пропонується кілька варіантів відповідей, з яких він повинні відзначити ті, які вважає правильними. Є два різновиди питань цього типу:

– тільки з однією правильною відповіддю (студент ставить відмітку в одному з кружечків);

– з одним або декількома правильними відповідями (студент ставить відмітку в одному або декількох квадратиках).

Альтернативне питання (Вірно / Невірно). Відповідь на це питання студент вибирає з двох варіантів: Вірно чи Невірно.

Вкладені питання. Це гнучкий засіб, який дозволяє викладачеві довільним чином конструювати тестові питання, вставляючи поля для введення відповідей типів Множинний вибір, Коротка відповідь і Числове питання в довільні місця тексту питання.

Питання на відповідність. Викладач задає перелік питань і відповіді до них (наприклад, перелік країн і їх столиць). Студент повинен знайти

правильну відповідність між питанням і відповіддю на нього (країною та її столицею).

Коротка відповідь (питання у відкритій формі). Відповіддю на питання є слово або коротка фраза, яку студент сам набирає на клавіатурі. Допускається кілька правильних або частково правильних відповідей. Текст відповіді може бути чутливими або нечутливими до регістра (тобто великі і малі літери у відповіді можуть відрізнятися чи ні) [9].

Випадкове питання на відповідність. Для студента це виглядає як звичайне питання на відповідність, проте дані для нього підбираються не викладачем, а випадковим чином запозичуються з присутніх в даній категорії питань типу Коротка відповідь. Тобто це питання не містить власної навчальної інформації, воно лише дозволяє в іншому, більш зручному для студентів вигляді, подати матеріал, присутній в інших питаннях даної категорії.

Випадкове питання. Цей засіб, що дозволяє випадковим чином вибирати для включення в тест наявні в даній категорії питання. Випадкове питання не містить власної навчальної інформації, це лише посилання на інші питання цієї категорії [10].

Опис. Цей тип питання насправді не є питанням. Все що він робить – відображає деякий текст, який не потребує відповідей. Його можна використовувати, щоб відобразити опис наступної групи питань. Опис не оцінюється.

Есе. У відповідь на це питання студент повинен ввести письмову відповідь. Це єдиний тип питань, які вимагають ручного оцінювання викладачем.

Численні досліди викладачів показують, що студенти в класах відчувають як все більше збільшується стрес. У корені цієї проблеми, як зазначають деякі дослідники [11-13], лежить шкільна практика ранжирування студентів за результатами тестів.

Загальні ознаки «екзаменаційного стресу»: порушення режиму сну, втому, дратівливість, нерегулярні прийоми їжі, збільшену схильність до інфекцій, неможливість сконцентруватися.

Інтенсивні «відповіді на питання» некоректні з точки зору викладання, оскільки це зменшує мотивацію учнів. Дослідники відзначають, що студенти вчать тільки те, що може бути протестовано, і ігнорують всю іншу інформацію. В результаті для учнів здати тест більш важливо, ніж розуміння суті предметів. Отримання високого бала стає головною метою освіти.

Поширення тестування в освітній системі під вивіскою звітності може призвести до помилкових результатів. Методичні відділи будуть розробляти все нові тести, учні будуть показувати всі кращі результати, але ціною за це буде звуження кола знань, обмеження успішності навчання.

Шлях до вирішення цих проблем лежить у підвищенні інтерактивності навчального курсу, застосування ігрового формату та залучення засобів мультимедіа.

На жаль, більшість корпоративного електронного навчання складається з набору екранів, що абсолютно рівнозначно перегортанню сторінок у книзі. Такий різновид курсів чекає швидке і справедливе покарання – нудьга користувачів. Іронія полягає в тому, що саме онлайнове навчання найкращим чином пристосоване до інтерактивності і з малими витратами може забезпечити набагато більшу захопливість, в порівнянні з традиційним навчанням. Більшість експертів підтверджують, що інтерактивність - найкращий шлях до реалізації таких необхідних дорослому учневі, метакогнітівних процесів, як самооцінка і самокорекція.

На сьогоднішній день існує безліч різноманітних програм, які виробляють перевірку знань учнів, по різних предметам навчання. Всі навчальні програми можу бути написані на різних мовах програмування: Delphi, C ++, C Sharp та ін..[14].

На сьогодні на ринку навчальних програм представлена чимала кількість програмних засобів для вивчення різних предметів у тому числі

іноземної мови. Більшість з них є у вільному доступі у Інтернет мережі. Розглянемо найпопулярніші з них:

Сайт Premier Training Center (рис. 1.1) пропонує пройти тест який приблизно оцінює рівень знань англійської граматики і словниковий запас. Коли користувач пройде тест, він отримає результат за британською та загальноєвропейською системами оцінювання. Для більш точного аналізу знань, необхідно оцінити розмовні навички, навички читання і сприйняття на слух.

Даний сайт є платним тому для навчання та перевірки знань студентів не підходить.

Рисунок 1.1 Головна сторінка сайту Premier Training Center

Наступною мережевою знахідкою став сайт NativeEnglish (рис 1.2). Даний сайт є безкоштовним та простим у користуванні. За умовами сайту пропонується тест який перевіряє вміння правильно вживати неособисті форми дієслова (інфінітива) в англійській мові. Із запропонованих варіантів відповідей до кожної пропозиції необхідно вибрати вірну форму.

При проходженні даного тесту програма вибирає 15 випадкових питань з 50 спеціально для нього підготовлених. Таким чином, користувач може проходити цей тест багаторазово, щоразу з новими питаннями.

По завершенню тесту буде виведена сторінка із загальним результатом, а також інформацією про правильні і неправильні відповіді.

Головною перевагою є безкоштовність. До недоліків варто віднести обмеженість ресурсів, та певна простота питань.

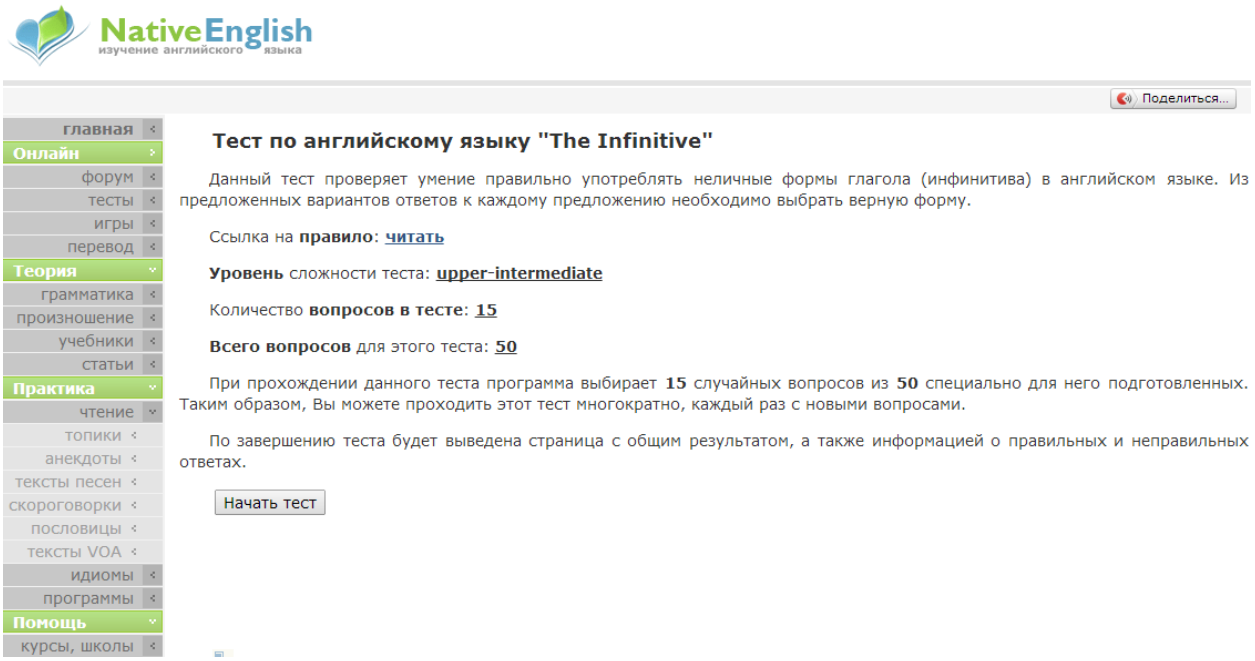


Рисунок 1.2 Головне вікно сайту NativeEnglish

Сайт Тесторіум «Тестування навчальних знань» (рис. 1.3) навпаки є багатофункціональним та інтерактивним. Відповідає всім нормам викладення та перевірки матеріалу.

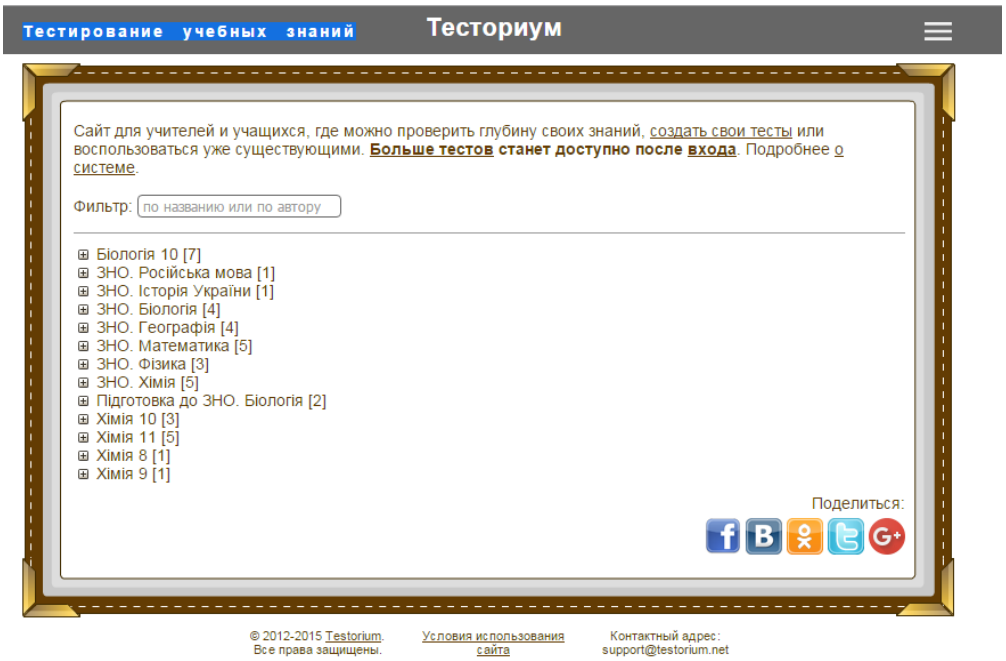


Рисунок 1.3 Сайт Тесторіум «Тестування навчальних знань»

Однак варто відзначити, що за обмеженням у доступі до Інтернет ресурсів (що на сьогодні є присутнім у Вищій школі) навчання та перевірка знань є просто неможливим.

Сайт «Тестування знань» (рис 1.4) має вузьке направлення, загалом це тести з Javascript.

Принципи сайту:

Тести припускають сучасні браузери.

Всі налаштування браузера – за замовчуванням.

Версія Javascript – найпоширеніша на поточний день, тобто ES5.

Скрізь "use strict"..

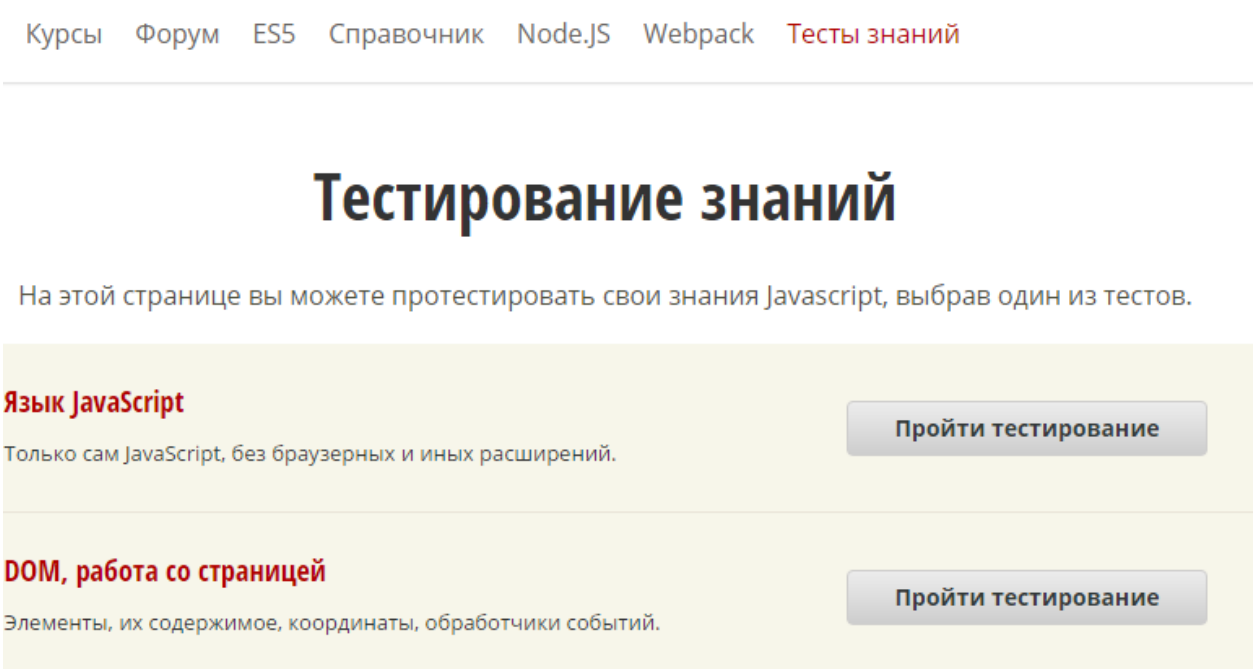


Рисунок 1.4 Сайт Тестування знань

Проте тут також постає питання методологічних аспектів, та наявності мережі.

Тому у межах даної дипломної роботи пропонується створити програмний додаток у вигляду тестового додатку. Створюваний додаток має працювати на стаціонарному ПК без застосування мережі Інтернет.

1.3 Підтвердження актуальності проектування

Застосування комп'ютерного тестування забезпечує об'єктивність оцінки знань на основі єдиних вимог (стандартів) - міжнародних, національних, галузевих, вузівських. Це особливо важливо для випускних і кваліфікаційних іспитів (одержання ліцензій, атестатів) аудиторів, банківських працівників, експертів та інших фахівців.

Сценарій тесту вводиться в ЕОМ за допомогою спеціальної програми, званої конструктором (дизайнером) тестів. Конструктор тестів, що входить в систему комп'ютерного тестування, дозволяє в діалоговому режимі ввести дані титулу сценарію, тестові завдання з еталонними відповідями,

інструкцію, вибрати з пропонованих варіантів необхідні параметри тесту, змінити пропоновані налаштування. Можна, наприклад, змінити пропоновану програмою шкалу оцінок. Вибираючи (встановлюючи) ті чи інші параметри, можна сформувати різні варіанти тесту, в т.ч. контрольно-навчальні.

У тесті рекомендується використовувати завдання однієї форми, допускається також поєднання різних форм завдань.

У залежності від призначення тесту і можливостей системи комп'ютерного тестування в сценарій можуть бути включені й інші частини, доповнені параметри тесту і завдань. Наприклад, крім розглянутих вище чотирьох частин сценарію тесту системи комп'ютерного тестування дозволяють включити список літератури, навчально-довідковий матеріал по тесту. Короткий конспект, статистичні дані, нормативи і т.п. призначені для самонавчання, самоконтролю (тренінгу) [2]. При роботі в контрольно-навчальному режимі студент, відповідаючи на завдання, може звернутися до даного навчально-довідкового матеріалу.

Якщо тест призначений для використання в контрольно-навчальному режимі, то за завданням може бути дозволено 2 або 3 спроби відповіді. За замовчуванням при комп'ютерному тестуванні дається тільки одна спроба відповіді на завдання. Слід зазначити, що зазвичай дозвіл на повторну відповідь на завдання дається при встановленні в «Параметрах тесту» порядку пред'явлення завдань «У порядку, вибрано учнем». При комп'ютерному тестуванні навчаний із загального списку завдань сам вибирає, на які відповідати в першу чергу. Як і при письмовому тестуванні, навчаний нерідко спочатку відповідає на легкі, а потім на важкі (складні) завдання.

При виставленні оцінки за чотирибальною або заліковою системою за відповідною шкалою за правильну відповідь на завдання умовно за замовчуванням дається 1 бал, за невірну відповідь – нуль (в т.ч. за частково правильний). Завдання можуть відрізнятися за труднощі (складності), але

передбачається, що всі учні перебувають у рівних умовах, відповідаючи на відносно велику кількість одних і тих же простих і важких завдань. Як показує багаторічна практика, при встановленні відносно високого процентного рівня для отримання заліку або задовільної оцінки (70-80%) такий підхід може використовуватися для поточної атестації, прийому семестрових заліків та іспитів.

Оцінка може визначатися і за сумою набраних балів при відповідному призначенні їх кількості за вірну відповідь за кожним завданням в залежності від рівня труднощі. За сумою набраних балів визначається місце учня в списку людей, що пройшли тестування (рейтинг), виставляється прийнята в вузах оцінка (незадовільно, задовільно, добре, відмінно).

1.4 Постановка завдань дослідження

Метою роботи є розробка програмного додатку для навчання з курсу «Алгоритми та структури даних» мовою C#.

Використовувані на усних і письмових заліках та іспитах традиційні питання і завдання без переробки, як правило, не можуть бути використані в тестах. Для них спеціально розробляються завдання в тестовій формі. По трудомісткості, складності та важливості дану роботу можна порівняти з поетапною підготовкою підручника з нової дисципліни (написання лекцій, навчального посібника, підручника) [3].

За змістом дій учня при контролі знань можна виділити завдання на:

– вибір однієї відповіді;

– вибір декількох відповідей;

– встановлення (знаходження) відповідності між елементами двох множин;

– встановлення правильної послідовності в ряду пропонованих елементів;

– ранжування пропонованих елементів;

– заповнення пропусків, завершення пропозицій;
– підстановку;
– складання відповіді;
– обчислення відповіді;
– обчислення і вибір відповіді.

Тестові завдання можуть бути питальними, ствердними, текстовими, табличними, графічними (використовуються графіки, малюнки, діаграми).

Система тестування має бути розроблена в системі програмування, яка має включати в себе редактор тестів, що дозволяє з'ясувати наявність повторень питань в темах, відсутність правильних і неправильних варіантів відповіді, некоректний вибір кількості питань для тестування по темі. База тестових завдань повинна складатися з питань двох типів: відкритих тестових завдань і багатоальтернативних тестових завдань. На екран завдання повинні виводитися по одному, щоб не відволікати увагу від пропонованого завдання. У випадку, якщо випробуваний не вибере відповідь, на екрані повинно з'явитися попереджувальне повідомлення. По закінченню тестування на екран має виводитися кількість правильних відповідей (надалі по закінченню тестування планується виставляння підсумкової оцінки).

Програмі для нормальної роботи потрібен комп'ютер, що задовольняє наступним вимогам: операційна система: Windows 7, Windows 10; процесор: Pentium 663 Mhz і вище; оперативна пам'ять:800 Мб і вище; місце на жорсткому диску: 300 Mb; монітор: з будь-яким дозволом; пристрої введення: клавіатура, миша.

Інтерфейс повинен бути інтуїтивно зрозумілий, максимально простий і зручний. Середовище функціонування програмного продукту – операційні системи сімейства MS Windows та інших.

Технічне завдання:

Розробити програмний додаток для навчання з курсу «Алгоритми та структури даних» мовою C#.

Використовуючи систему програмування, створити систему тестування для забезпечення якісного, швидкого контролю вмінь знань учнів.

Включає в себе:

Контролюючі та тестуючі матеріали:

– вибір однієї відповіді;

– вибір декількох відповідей;

– встановлення (знаходження) відповідності між елементами двох множин.

Проектований програмний продукт повинен реалізовувати такі вимоги до функціональних характеристик:

– вимоги до надійності;

– налаштування;

– умови експлуатації;

– вимоги до складу і параметрів технічних засобів;

– вимоги до інформаційної та програмної сумісності;

– вимоги до документації.

1.5 Засоби вирішення поставленого завдання

У попередніх підрозділах було сказано, що процес вирішення поставленого завдання розбивається на 3 етапи, перший з яких полягає в побудові інтерфейсу з користувачем на основі екранних форм і візуальних об'єктів проектування.

Для початку тесту передбачено вибір однієї з трьох функцій: почати тест, редагування тесту, довідка.

Для виведення питань, введення відповідей користувача передбачимо окрему форму FormTest

При побудові форми передбачалося, що база питань буде побудована таким чином, щоб відповідь припускає вибір тільки одного з перерахованих

чотирьох варіантів відповідей. Тобто, база не повинна містити питань з множинним вибором, або варіантів без правильної відповіді. Також до кожного питання обґрунтовується відповідь.

Аналізуючи вимоги до функцій, що розробляються під розроблені екранні форми і словесний алгоритм, можна виділити наступні програмні модулі:

1) Генерація питань;

2) Аналіз поточної відповіді і перехід на наступне питання, якщо питання не останнє здійснити будь-який висновок результатів тестування при досягненні останнього питання;

3) Модулі переходу між екранними формами тестування.

Кожен з вищенаведених модулів реалізуємо за допомогою процедури. Модуль 1 доцільно запускати один раз при створенні форми з тестами. Модуль 2 слід запускати кожен раз при початку нового тесту, тобто при активізації форми з тестами. Модуль 3 повинен запускатися кожен раз при натисканні на кнопку «Результат». Модулі 4 повинні бути обробниками натискань на кнопку «Почати тестування» редагування тесту і кнопку «Довідка» екранної форми тестування.

Складемо алгоритм додатка. На алгоритмі (рис. 1.6) наведено основні події екранних форм. У цьому алгоритмі:

Блок №4 реалізований процедурою Form, BitBtnStartClick,

блок №5 - процедурою FormTest. FormCreate,

блок №6 - процедурою FormTest. FormActivate,

блок №7 - процедурою FormTest BNewTestClick,

блок №8 реалізований процедурою FormTest. BitBtn1Click.

Аналізуючи всі вищевикладені припущення і розроблені форми, можемо скласти словесні алгоритми програми.

1. Запустити екранну форму авторизації Form

2. При натисканні на кнопку «Почати тестування» слід виконати дії:

2.1 Згенерувати базу питань, і перемішати їх випадковим чином;

2.2 Сформувати ім'я файлу з імені студента і його групи і відкрити файл для запису;

2.3 Обнулити кількість правильних відповідей;

2.4 Встановити номер поточного питання на перше питання;

2.5 Приховати форму Form і показати форму з тестами FormTest;

2.6 Показати перше питання і варіанти його відповідей на формі FormTest;

2.7 Очікувати натискання на кнопку «Далі».

3. При натисканні на кнопку «Далі» слід виконати дії:

3.1 Проаналізувати обраний варіант відповіді і порівняти його з правильним, і якщо відповідь правильна, то наростити кількість правильних відповідей на 1;

3.2 Записати у файл протоколу поточне питання і варіант, який вибрав користувач в якості відповіді;

3.3 Якщо номер поточного питання менше загальної кількості питань в базі, то наростити номер на 1, вивести чергове запитання на форму і перейти до початку пункту 3.

3.4 Якщо номер поточного питання дорівнює кількості запитань у базі, то:

3.4.1. підрахувати оцінку пропорційно кількості правильних відповідей;

3.4.2. показати повідомлення з оцінкою;

3.4.3. зробити кнопку «Повернутися» видимою

4. При натисканні кнопки «Повернутися» закрити форму з тестами FormTest і показати форму вибору функцій Form.

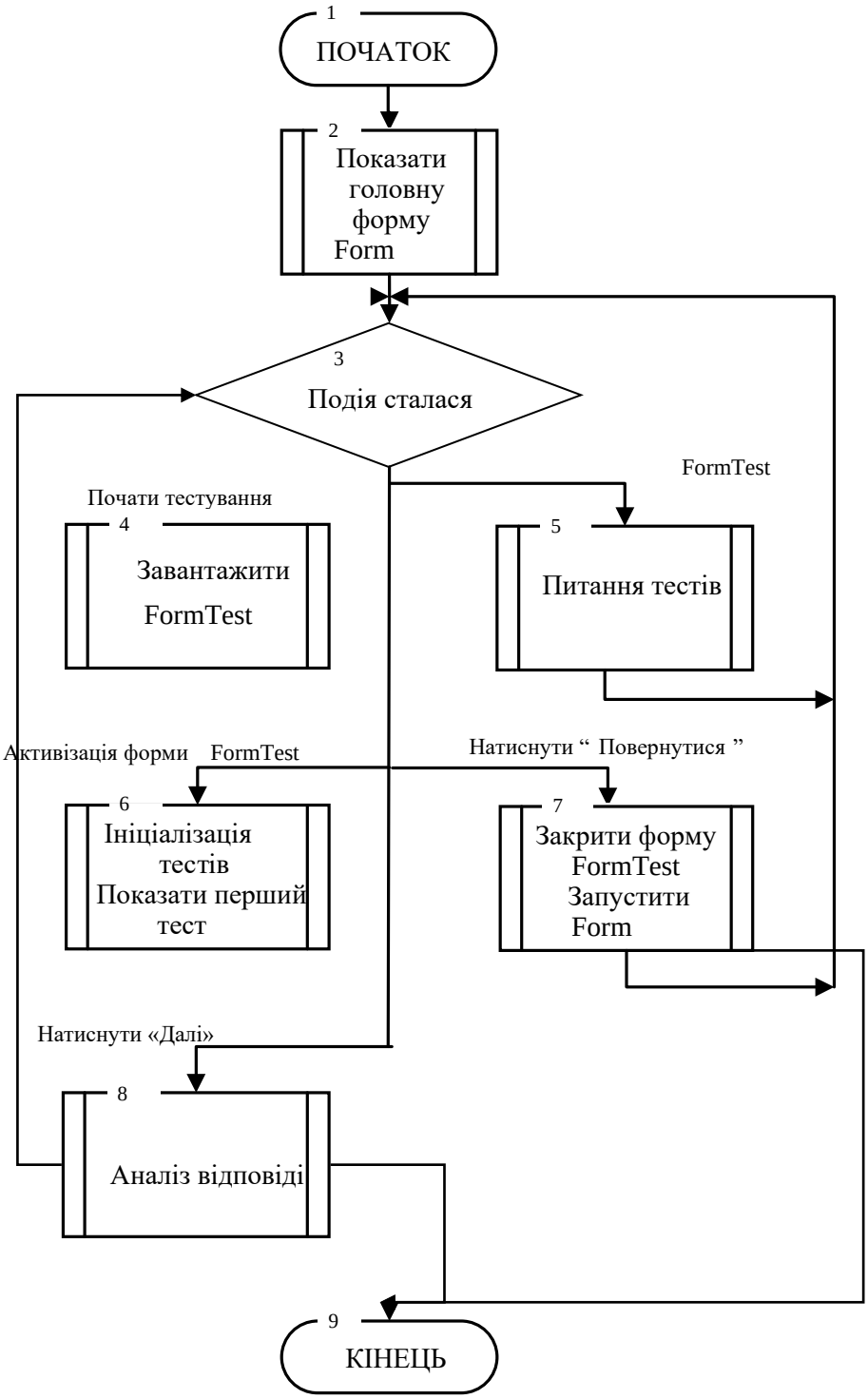


Рисунок 1.6 Обробка подій екранних форм програми

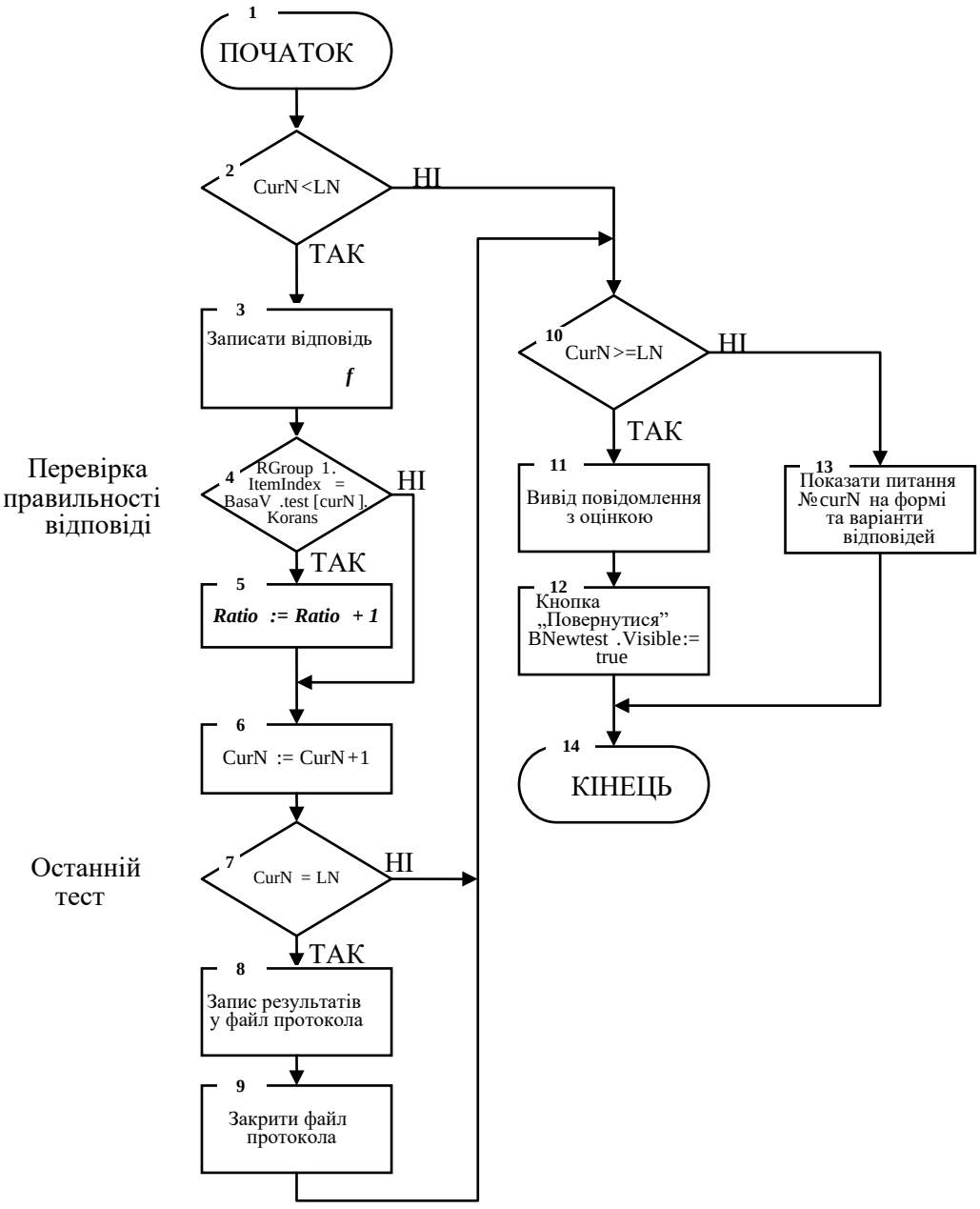


Рисунок 1.7 Алгоритм процедури Click

РОЗДІЛ 2.
ПРОЕКТНІ І ТЕХНІЧНІ РІШЕННЯ СИСТЕМИ

2.1 Інформаційне забезпечення проектованої системи

Моделювання являє собою метод опосередкованого пізнання за допомогою використання об'єктів-замінників. Моделювання спирається на філософську теорію відображення і загальнонауковий метод абстракції. Воно включає в себе побудову, вивчення і застосування моделей, опосередковуючи відносини між пізнаваним об'єктом і суб'єктом, що пізнає.

Моделлю називається матеріальний чи подумки представлений об'єкт, який заміщає об'єкт-оригінал і з певним ступенем наближення відображає його найважливіші характеристики і поведінку.

Моделювання означає також використання моделей для визначення або уточнення характеристик і раціоналізації способів побудови об'єктів, які знову конструюються. Структурна модель системи тестування відображає його найважливіші характеристики і поведінку (рис. 2.1).

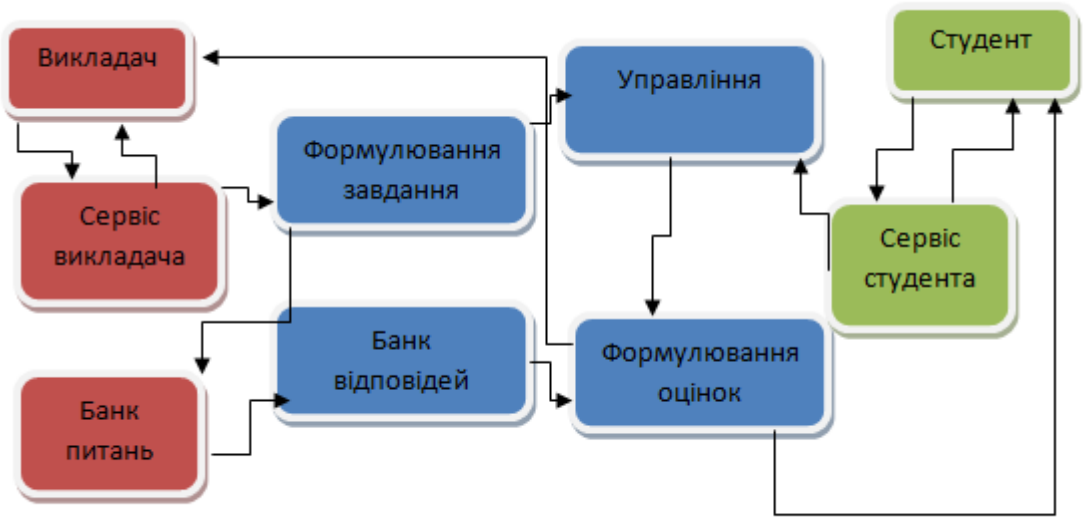


Рисунок 2.1 Структурна модель системи тестування

Сервіс викладача, включає в себе редактор тестів, що дозволяє створювати нові завдання, з'ясовувати наявність повторень питань в темах, відсутність правильних і неправильних варіантів відповіді.

Формування завдання. Відповідно до вказівок викладача цей блок створює сценарій перевірочної роботи для кожного студента, викладач вибирає завдання по кожній темі.

Банк питань. База тестових завдань, яка на даний момент складається з питань двох типів: Відкриті тестові завдання, багатоальтернативні тестові завдання.

Банк відповідей містить правильні відповіді до кожного завдання.

Блок управління забезпечує звірку даного студентом відповіді з вмістом банку відповідей.

Блок формування оцінок. Розраховує підсумкову оцінку і виводить її на екран.

2.2 Математичне забезпечення програмного додатку

Досить поширеним методом контролю рівня знань студентського контингенту, в даний час є тестування, яке також представляє певну діалогову форму навчання і зазвичай реалізується в рамках відповідної інформаційної технології.

Процедура тестування описується кіберсистемою з двох кінцевих автоматів A і A′, з'єднаних за схемою рис.2.2, і алгоритм їх поведінки, що представляє окремий випадок загального алгоритму діалогу, наведеного в табл. 2.1, в тому сенсі, що алгоритм тестування реалізується в одному циклі. Формально тест являє собою слово $z = z_1 z_2 \ldots z_n$ в алфавіті Z, так, що кожна буква $z_i \in Z$, $i = \overline{L;n}$, де і - це i -е завдання тесту z [15].

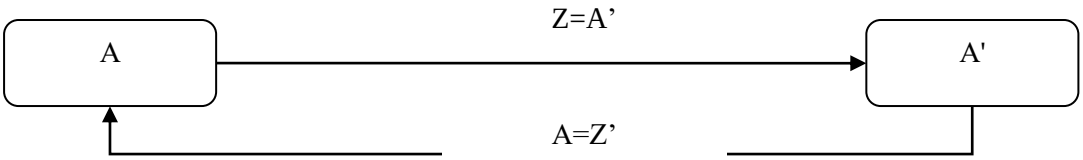


Рисунок 2.3 Схема кіберсистеми з двох кінцевих автоматів А і А'[16]

Даний тест z надходить на вхід $\acute{A} = Z$ автомата А′ (тестований об'єкт) і обробляється як слово відповідно до алгоритму, який в даному випадку задає процедуру «рішення» тестових завдань. На основі даних рішень за кожним завданням $z_i \in \acute{A}$ дається відповідь $a\ i = \acute{g}(s_i'; z_i)$, де $s_i' \in S$ - варіант рішення, запропонований тестованим за завданням z_i. В результаті, за допомогою алгоритму, на виході А′ формується слово $a = a_1\ a_2 \ldots a_n;\ a_i \in Z' = A$, Яке для контролю надходить на вхід A автомата А.

Таблиця 2.1 – Опис алгоритму процесу навчання [17]

Автомат А(викладач)	Автомат А′(студент)	Рівень навчання				
A	Резолюція	S	Z= А’	Резолюція	Z'	
-	-	-	z_0	$s''_{01} \in S_0$‘	a_1	$S_I' = S_0' \cup \Delta S_0'$
a_1	$s_{11} \in S$	$s_{12} \in S$	z_1	$s''_{02} \in S_0$‘	a_2	
a_2	$s_{21} \in S$	$s_{22} \in S$	z_2	$s''_{03} \in S_0$‘	a_3	
a_{yi-1}	$s_{yi-11} \in S$	$s_{yi-12} \in S$	z_{ri-1}	$s''_{ri} \in S_0$‘	a_{yi}	
a_{yi}	$s_{yi-1} \in S$	$s_{yi-2} \in S$	z_{ri}	$s''_{11} \in S_1$ ‘	a_{yi-1}	$S_2' = S_I' \cup \Delta S_I'$
$a_{yi+y2-1}$	$s_{yi+y2-1;1} \in S$	$s_{yi+y2-1;2} \in S$	$z_{ri+r2-1}$	$s''_{1ri} \in S_1$‘	a_{yi+y2}	
a_k	$s_{k1} \in S$	$s_{k2} \in S$	z_k	$s''_{n-1;1} \in S_{n-1}$ ‘	a_{k+1}	$S = S_n'$ $= S_{n-I}' \cup \Delta S_{n-I}'$
a_{k+yn-1}	$s_{k+rn-1;1} \in S$	$s_{k+rn-1;2} \in S$	z_{k-rn-1}	$s''_{n-1;rn} \in S_{n-1}$‘	a_{k+yn}	
a_{k+yn}	$s_{k+rn-1} \in S$	$s_{k+rn-2} \in S$	-	-	-	

Контроль відповіді зводиться до того, що кожна буква a_i слова a порівнюється з результатом правильного рішення $s_i \in S$ і відповідь a_i

набуває рейтингу $r_i=g(s_i;a_i)\in Z$. У підсумку на виході А формується слово $r = r_1 r_2 \dots r_n$, за яким встановлюється загальний результат виконання тесту z, цей результат доводиться до А′ і на цьому процедура тестування завершується.

В українському освітньому просторі в процедурах комп'ютерного тестування в основному переважають тести з відкритою або закритою формою завдань [19; 20]. При закритій формі тестових завдань (формат multiple choice) кожне таке завдання забезпечено набором відповідей, з яких зазвичай тільки одна вірна відповідь, і студенту пропонується вибрати по одній відповіді з кожного такого набору. У завданнях відкритої форми відповіді з вибором не передбачаються і студент повинен сам подати відповідь, яка свідчить про наявність або відсутність у нього необхідних знань по заданому питанню. При компонуванні тестових батарей завдання із закритою формою зазвичай розташовують на початку (частина А), а більш складні завдання зазвичай припускають відкриту форму відповіді і розміщуються ближче до кінця тесту (частина Б).

Алгоритм обробки тестів [21]. Нехай, як і вище, тест z представляється словом $z = z_1 z_2 \dots z_n$, у якого перший k букв $z_i\ i = \overline{L;k}, k < n$ – описують завдання із закритою формою, а інші літери $z_i, i = \overline{k+\mathrm{I}}, n$ – це завдання з відкритою формою відповідей. Тоді вихідна безліч Z’ автомата А′ *(i,* відповідно, вхідної безлічі A автомата А) можна представити у вигляді розбиття $Z' = Z'_3 \cup Z'_0$ (відповідно, $A' = A'_3 \cup A'_0$), де індекси *3; 0* - приписуються завдань з закритою і відкритою формою відповідей, відповідно.

Нехай при виконанні тестових завдань із закритою формою випробуваному по кожному окремому завданню слід вибрати одну відповідь з $l > 1$ запропонованих відповідей. Тоді підмножину $Z'_3 \cup Z'$ автомата А′ можна визначити базисним набором l- мірних ортонормованих векторів. У цьому випадку $g'(s'_i; z_i) = \overrightarrow{a_i}$, де $s'_i \in S$ – варіант рішення, запропонований

тестованим за завданням $z_{i,}$ $i = \overline{L;k}$, на підставі якого обрана відповідь $\overrightarrow{a_\iota}$ *(A* $_{i1;}$ *...;* $a_{il)} \in Z_3'$, Так, що одна з координат $a_{ij,}$ $j = i = \overline{L;I}$, що відповідає обраній відповіді у запропонованому наборі, дорівнює *1,* а решта координати нульові. З векторів $\overrightarrow{a_\iota}$, сформуємо стовпець *(* $\overrightarrow{a_\iota}$,… $\overrightarrow{a_k}$*)**, де * - означає операцію транспонування, після чого, розгортаючи цей стовпець за координатами векторів $\overrightarrow{a_k}$, отримуємо матрицю відповідей $A_{kl.}$

Матриця A_{kl} представляє вхідний сигнал для автомата А, який реалізує контроль тесту. Для цього введемо вектор $\overrightarrow{g_\iota}$ *(Q* $_{i1;}$ *...;* $q_{il)} \in A_3 = Z_3'$, що представляє вектор-ключ до завдання $z_{i,}$ $i = \overline{L;k}$, у якого одна з координат $q_{ij,}$ $j = \overline{L;I}$, відповідна правильній відповіді у запропонованому наборі, дорівнює *1,* а решта координати - нульові. З векторів $\overrightarrow{g_\iota}$ сформуємо рядок *(* $\overrightarrow{g_\iota} \ldots \overrightarrow{g_k}$ *)* і, потім, розгортаючи координати $\overrightarrow{g_\iota}$ в стовпці, будуємо ключ-матрицю $Q_{lk.}$

Легко переконатися, що для матриці $C_{kk} = A_{kl}\ Q_{lk}$ діагональні елементи $c_{ii} \in \{0,1\}$, причому, $c_{ii} = 0,$ якщо дана невірна відповідь до завдання z_i і $c_{ii} = 1$ - в іншому випадку. Тому сума діагональних елементів Sp *(*$C_{kk)}$ - це кількість правильних відповідей серед завдань відкритої форми $z_{i,}$ $i = \overline{L;k}$ тесту *z.* Тоді, якщо R_{kk} = diag *(*$r_{1;}$*...;* $r_{k)}$ - скалярна матриця рейтингів завдань $z_{1;}$ *...;* $z_{k,}$ то Sp *(*$C_{kk}\ R_{kk)}$ - визначає загальний рейтинг, що оцінює результати виконання тесту *z* в частині завдань із закритою формою відповіді.

При відкритій формі тестових завдань $z_{i,}$ $i = \overline{L;k,n}$, Відповідь до кожного завдання подається у вигляді деякого числа, так, що для підмножини $Z_0' \subset Z'$ можна покласти $Z_0' = R$. У цьому випадку $a_i = g'(s_i', z_i)$, де $a_i \in R$ − чисельна відповідь до завдання z_i у відкритій формі, і можна визначити вектор $\bar{a}$ *(A* $_{k + 1;}$ *...;* $a_{n)}$.Нехай тепер $\bar{q}$ *(Q* $_{k + 1;}$ *...;* $q_{n)}$ - вектор-ключ для тестових завдань $z_{i,}$ $i = \overline{L;k,n}$, так, що $q_i \in R$ − правильна відповідь для завдання $z_{i.}$ Складемо різницю $\Delta\vec{g} = \vec{g} - \vec{a},$ Причому, координати вектора $\Delta\vec{g}$ запишемо за правилом:

$$\Delta g = g_i - a_i = \begin{cases} I, a_i = g_i \\ 0, a_i \neq g_i \end{cases} \quad (2.1)$$

Тоді, якщо $\bar{y}$ $(R_{k+1;} ...; r_{n)}$ − є рейтинг-вектор, де $r_i \in R$ − рейтинг завдання $z_{i,}$ то скалярний добуток $\Delta\vec{g},\vec{r}$ визначає загальний рейтинг виконання тестових завдань z_i з відкритою формою відповіді.

Загальний рейтинг при виконанні тесту $z = z_1 z_2 ... z_n$ можна визначити сумою Sp $(C_{kk} R_{kk})$ + $\Delta\vec{g},\vec{r}$, з якою пов'язується певна оціночна шкала *P*, що встановлює результати тестування, що доводяться до студента [22].

2.3 Програмне забезпечення програмного додатку

Технічне забезпечення – комплекс технічних засобів, призначених для роботи інформаційної системи, а також відповідна документація на ці засоби і технологічні процеси [26].

Комплекс технічних засобів складають:

- комп'ютери будь-яких моделей;
- пристрої збору, накопичення, обробки, передачі та виведення інформації;
- пристрої передачі даних і ліній зв'язку;
- оргтехніка та пристрої автоматичного знімання інформації;
- експлуатаційні матеріали і ін.

Документацією оформляються попередній вибір технічних засобів, організація їх експлуатації, технологічний процес обробки даних, технологічне оснащення. Документацію можна умовно розділити на три групи:

1. Загальносистемну, що включає державні й галузеві стандарти з технічного забезпечення.

2. Спеціалізовану, що містить комплекс методик по всіх етапах розробки технічного забезпечення.

3. Нормативно-довідкову, використовувану при виконанні розрахунків з технічного забезпечення.

До теперішнього часу склалися дві основні форми організації технічного забезпечення (форми використання технічних засобів): централізована і частково або повністю децентралізована.

Централізоване технічне забезпечення базується на використанні в інформаційній системі великих ЕОМ та обчислювальних центрів [27].

Децентралізація технічних засобів передбачає реалізацію функціональних підсистем на персональних комп'ютерах безпосередньо на робочих місцях.

Перспективним підходом слід вважати, вочевидь, частково децентралізований підхід – організацію технічного забезпечення на базі розподілених мереж, що складаються з персональних комп'ютерів і великий ЕОМ для зберігання баз даних, загальних для будь-яких функціональних підсистем [28].

Математичне та програмне забезпечення – сукупність математичних методів, моделей, алгоритмів і програм для реалізації цілей і завдань інформаційної системи, а також нормального функціонування комплексу технічних засобів [20].

До засобів математичного забезпечення відносяться:

- засоби моделювання процесів управління;
- типові завдання управління;
- методи математичного програмування, математичної статистики, теорії масового обслуговування та ін.

До складу програмного забезпечення входять загальносистемні і спеціальні програмні продукти, а також технічна документація.

До загальносистемного програмного забезпечення відносяться комплекси програм, орієнтованих на користувачів і призначених для

вирішення типових задач обробки інформації. Вони служать для розширення функціональних можливостей комп'ютерів, контролю і управління процесом обробки даних.

Спеціальне програмне забезпечення являє собою сукупність програм, розроблених при створенні конкретної інформаційної системи. У його склад входять пакети прикладних програм (ППП), які реалізують розроблені моделі різного ступеня адекватності, що відображають реального функціонування об'єкта [29].

Вибір того чи іншого програмного засобу визначається як специфікою розробки програмного забезпечення та його популярністю, так і фінансовими можливостями розробника.

У якості мови програмування обрано C #. Мові програмування C # в червні 2020 року виповнилося 20 років, але вона і не планує здавати свої позиції.

Мова програмування C # є однією з найбільш затребуваних мов в IT-галузі, використовується для створення як настільних (десктопних) програм, так і для веб-додатків. C # була створена спеціально для роботи з фреймворком (платформою) Microsoft .NET, що надає безліч можливостей і що спрощує створення додатків. Одним з основних компонентів пакету Microsoft .NET Framework є Common Language Runtime (CLR) загальномовне виконуюче середовище, яке компілює код програми на мові MSIL (в який компілюється код на мові C #) під час його виконання, а також надає MSIL-програмами (а отже , і програмами, написаним на мовах високого рівня, що підтримують .NET Framework) доступ до бібліотеки класів .NET Framework.

За допомогою C # і .NET Framework можна створювати класичні настільні Windows Forms і Windows Presentation Foundation додатки, веб-додатки (використовуючи технологію ASP.NET), компоненти для розподілених додатків і доступу до баз даних. C # надає засоби для розробки практично будь-якого типу програмного забезпечення для платформи Windows.

У C # розроблявся дружній інтерфейс, для роботи з системою тестування. Також створення робочої області типу панелі управління «кнопок».

Для того, щоб програма виконувала наказані їй дії, наприклад, обчислювала, виводила результат, реагувала на дії користувача, наприклад, на натискання кнопок, вибір рядків зі списку, необхідний програмний код.

Програмний код – це набір слів і символів мови програмування. Слова і символи повинні бути записані строго за правилами мови, без орфографічних і пунктуаційних помилок [31]. Саме точне написання дозволить комп'ютеру однозначно зрозуміти і виконати програму.

В якості інструментального середовища проектування використовується Rational Software Architect. Для опису моделі використовується мова UML.

Діаграма компонентів (Рис. 2.2) показує структурні відносини між компонентами системи. У UML 2 компоненти є автономними інкапсульованими одиницями всередині системи або підсистеми, які забезпечують один або кілька інтерфейсів. Тому діаграма компонента дозволяє архітекторові переконатися в тому, що компоненти реалізують задану функціональність системи.

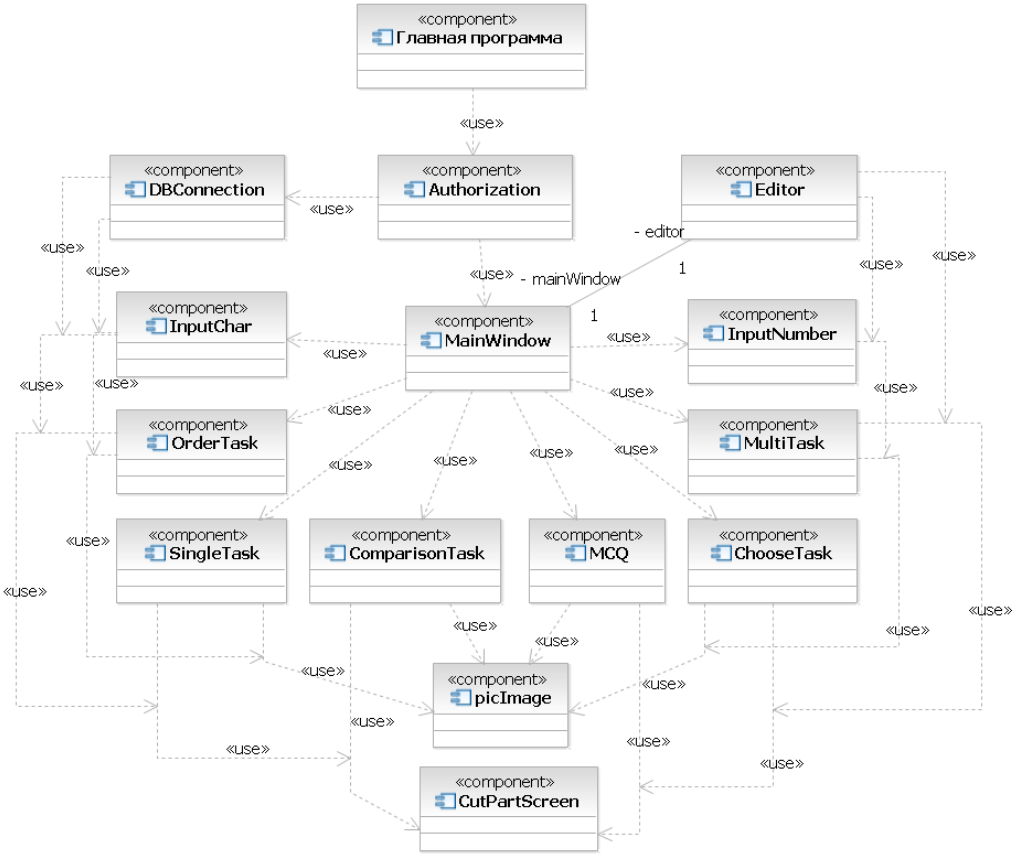


Рисунок 2.2 Діаграма компонентів

Діаграма класів

Діаграми класів є центральною ланкою методології об'єктно-орієнтованого аналізу і проектування.

Діаграма класів показує класи і їхні відносини, тим самим представляючи логічний аспект проекту. На стадії аналізу діаграми класів використовуються, щоб виділити загальні ролі та обов'язки сутностей, що забезпечують необхідну поведінку системи. На стадії проектування діаграми класів використовуються, щоб передати структуру класів, що формують архітектуру системи.

Кожен клас повинен мати ім'я, причому воно повинно бути унікально в проекті, що містить його.

На діаграмі класів зображуються також атрибути класів, операції та обмеження, які накладаються на зв'язки між об'єктами.

На (Рис. 2.3) зображена Діаграма класів розроблюваної системи.

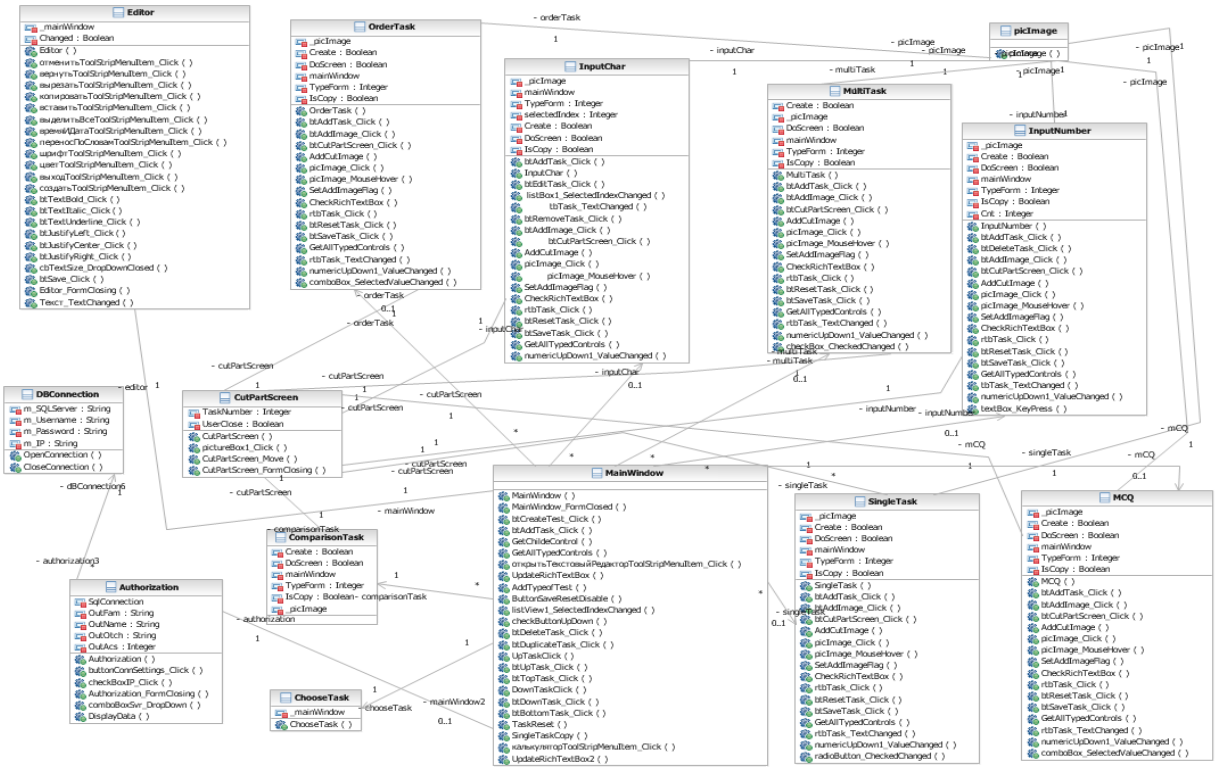


Рисунок 2.3 Діаграма класів

Висновки до розділу 2

Другим розділом даної дипломної роботи передбачено висвітлення питань проектної частини роботи. Наведено інформаційну модель системи, що розробляється, яка включає в себе: сервіс викладача, формування завдання, банк питань, базу тестових завдань, банк відповідей, блок управління, блок формування оцінок.

Проводиться вибір технічного та програмного забезпечення. Наведено необхідну конфігурацію ПК та описано мову програмування застосовану для розробки системи тестування. Здійснено проектування діаграми компонентів та діаграми класів. В якості інструментального середовища проектування використовується Rational Software Architect. Для опису моделі використовується мова UML.

Виконано розробку програмного продукту, в межах отриманого результату, варто зазначити, що в рамках програми є підтримка текстового введення в якості відповіді (без вказівки альтернатив), редагування тестів, обґрунтування відповіді та ін. Програма може бути використана в освітніх цілях.

РОЗДІЛ 3.
ОПИС РОБОТИ ПРОГРАМНОГО ДОДАТКУ ДЛЯ НАВЧАННЯ З КУРСУ «АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ»

3.1 Інструкція користувача

Система, що розробляється здійснює створення тестів і проведення комп'ютерного тестування, збору та аналізу результатів, виставлення оцінки.

Програма легка і зручна у використанні і освоєнні. Надає можливість відповіді на питання та обґрунтування свого вибору. Для створення тестів є дуже зручний редактор тестів з дружнім інтерфейсом. Будь-який викладач, який навіть володіє комп'ютером на початковому рівні, може легко скласти свої тести і використовувати їх під час тестування.

3.1.1 Системні вимоги до клієнтської машини

Мінімальні вимоги до обладнання клієнта наступні:

Процесор Intel Pentium III з частотою від 700 МГц або вище;

Оперативна пам'ять, доступна операційній системі, не менш 64Мб;

Монітор з підтримкою відеорежиму 1024x768 при 256 кольорах;

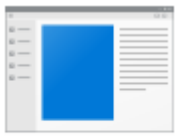
Маніпулятор типу "миша", пристрій введення – клавіатура.

Вимоги до програмного забезпечення клієнта наступні:

- ОС Windows 2010
- Visual Studio 2019
- C#
- БД SQL 2012.

3.1.2 Порядок роботи з системою тестування

За для запуску розробленої програми варто натиснути на іконці



, після чого на екрані з’явиться головне вікно програми (рис. 3.1)

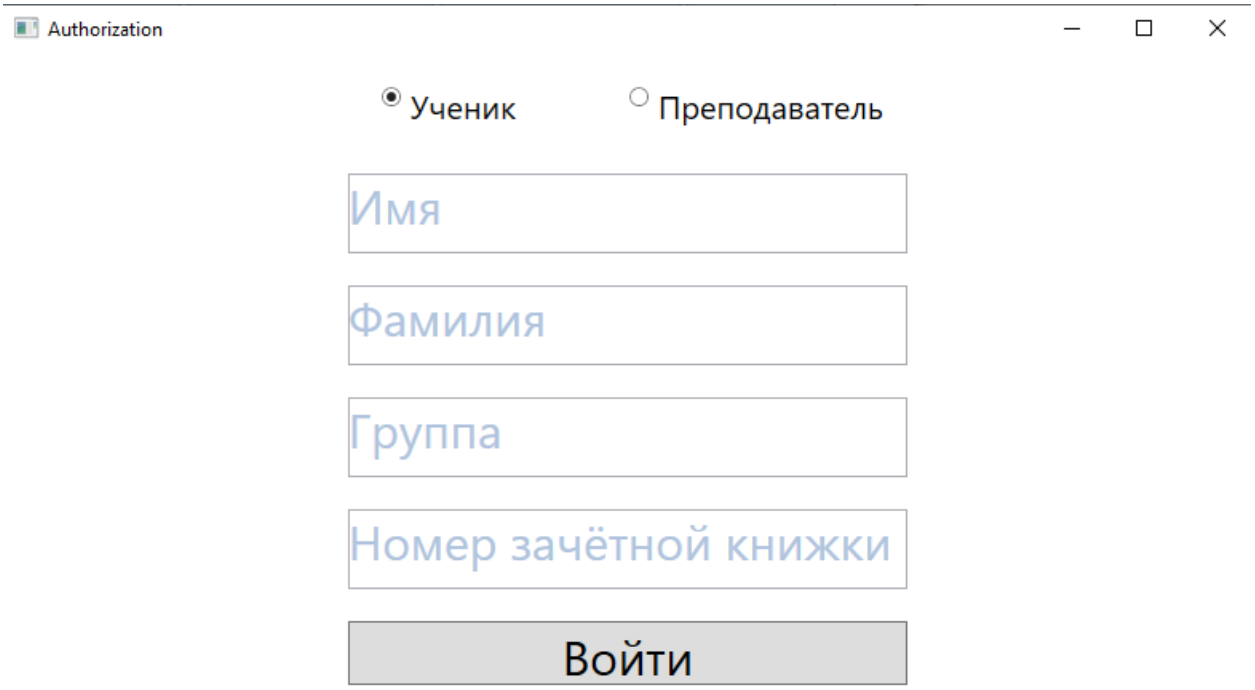


Рисунок 3.1 Головне вікно програми

Після чого користувач повинен обрати у якості кого він входить у систему:

– студент;
– викладач.

Якщо користувач входить у якості викладача, він вводить свої дані це логін та пароль (рис. 3.2)

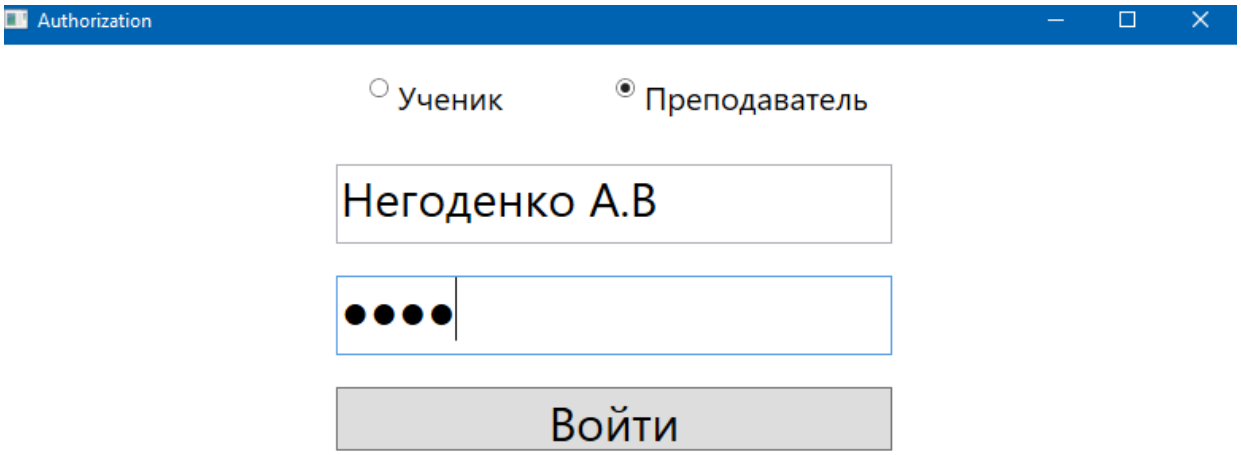


Рисунок 3.2 Авторизація

та потрапляє на сторінку викладача (рис. 3.3)

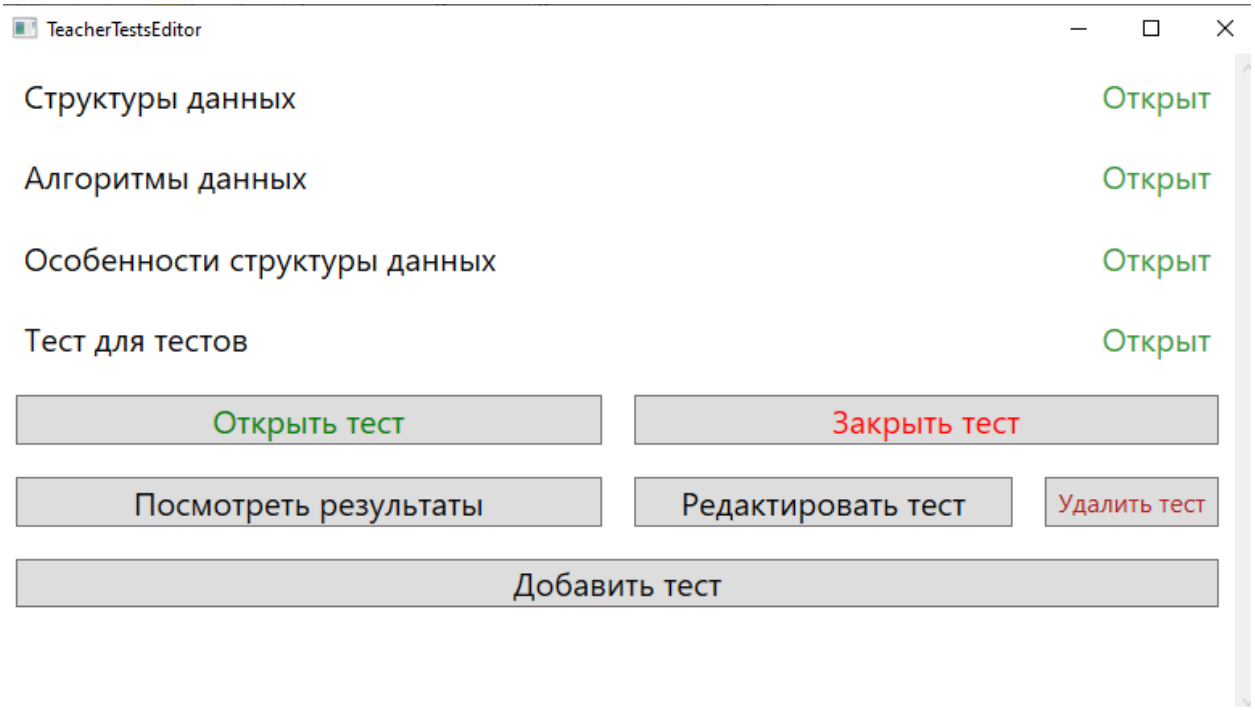


Рисунок 3.3 Сторінка викладача

Натискаємо на «Редагувати тест» та отримуємо тест у вигляді конструктора, де є можливість змінювати питання, додавати відповіді тощо (рис. 3.4).

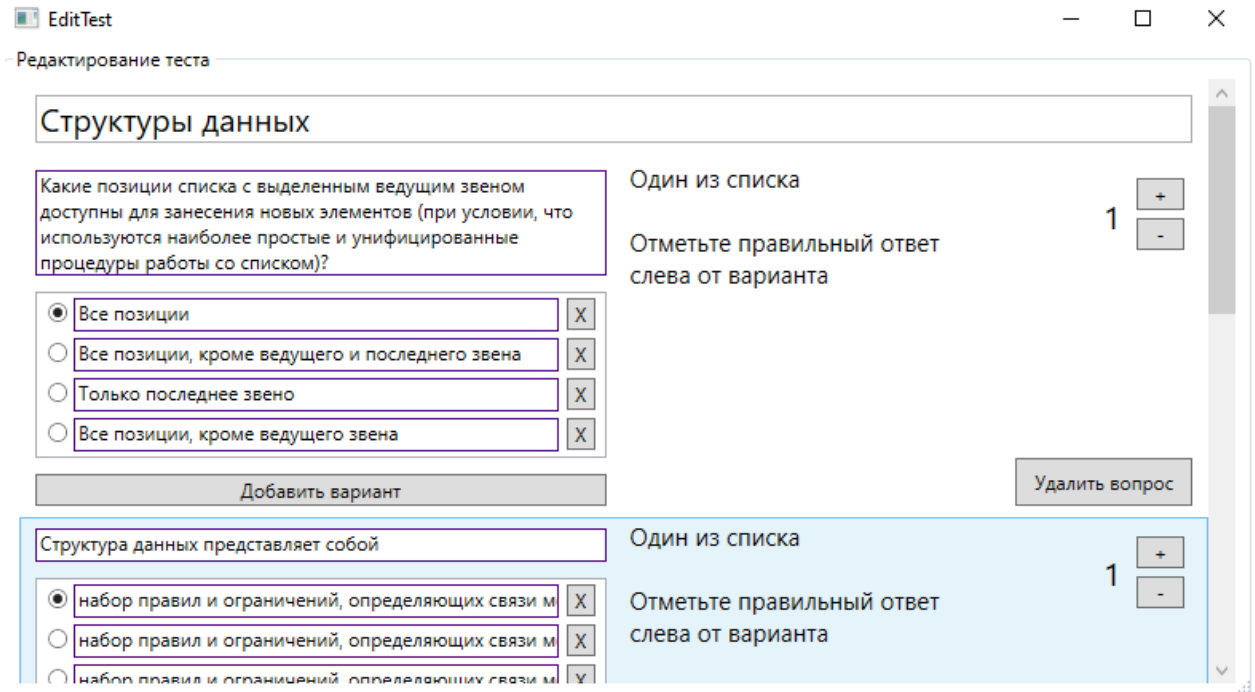


Рисунок 3.4 Сторінка викладача. Редагування тесту

Після внесення всіх необхідних змін, викладач натискає на кнопці зберегти та тест зберігається. При натисканні на кнопці «Переглянути результати», викладач бачить на екрані перелік студентів, які виконали тестування.

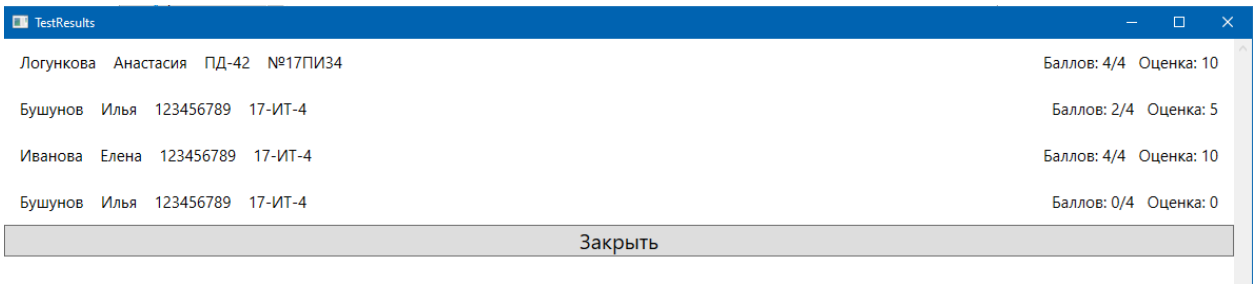


Рисунок 3.5 Сторінка викладача. Результати тестування

Якщо здійснити вхід у систему у якості студента, користувач має заповнити форму студента (рис. 3.6).

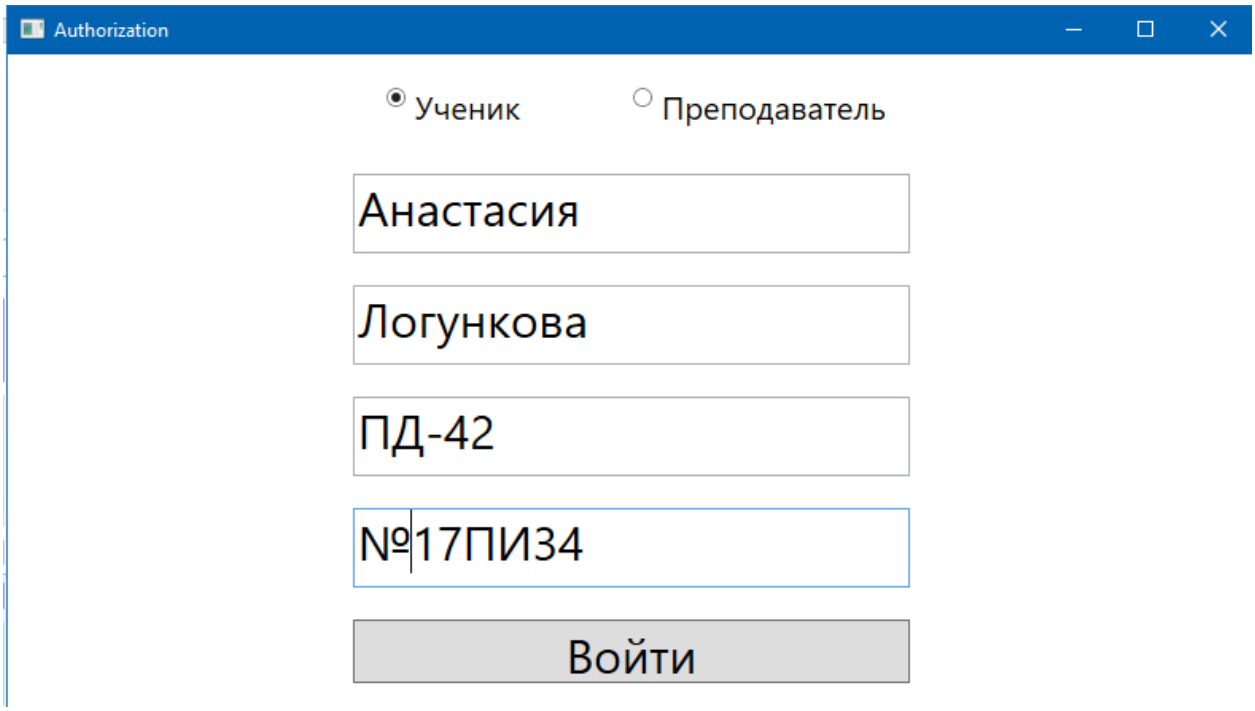


Рисунок 3.6 Авторизація

та потрапляє на сторінку студента (рис. 3.7)

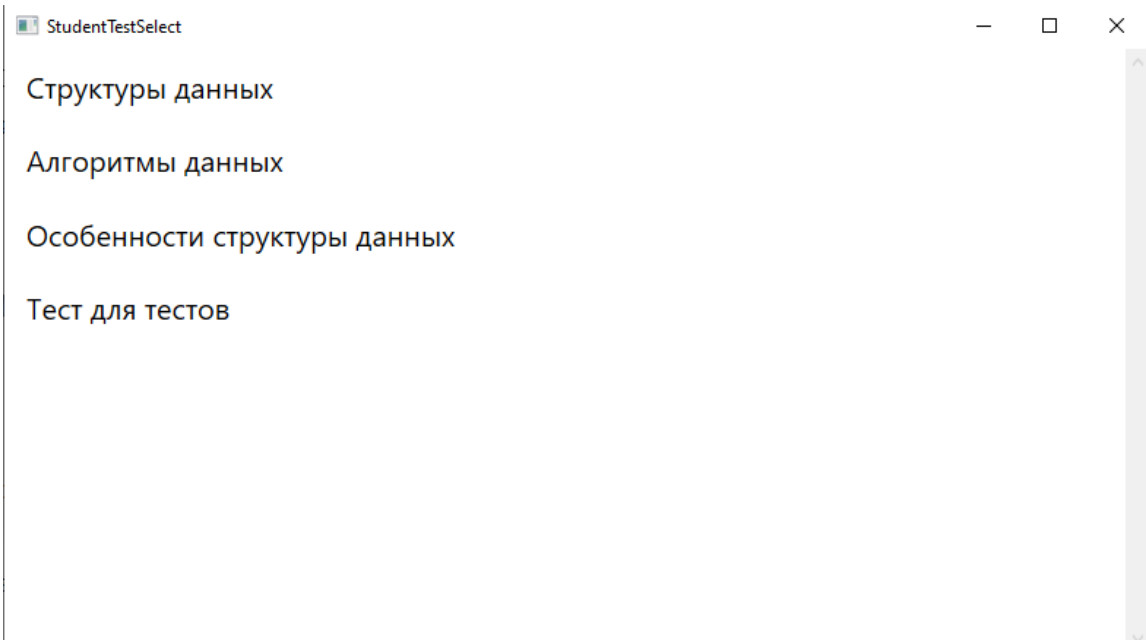


Рисунок 3.7 Сторінка студента

Користувач, що проходить тестування відповідає на питання встановленням крапки у відповідному осередку (рис. 3.8).

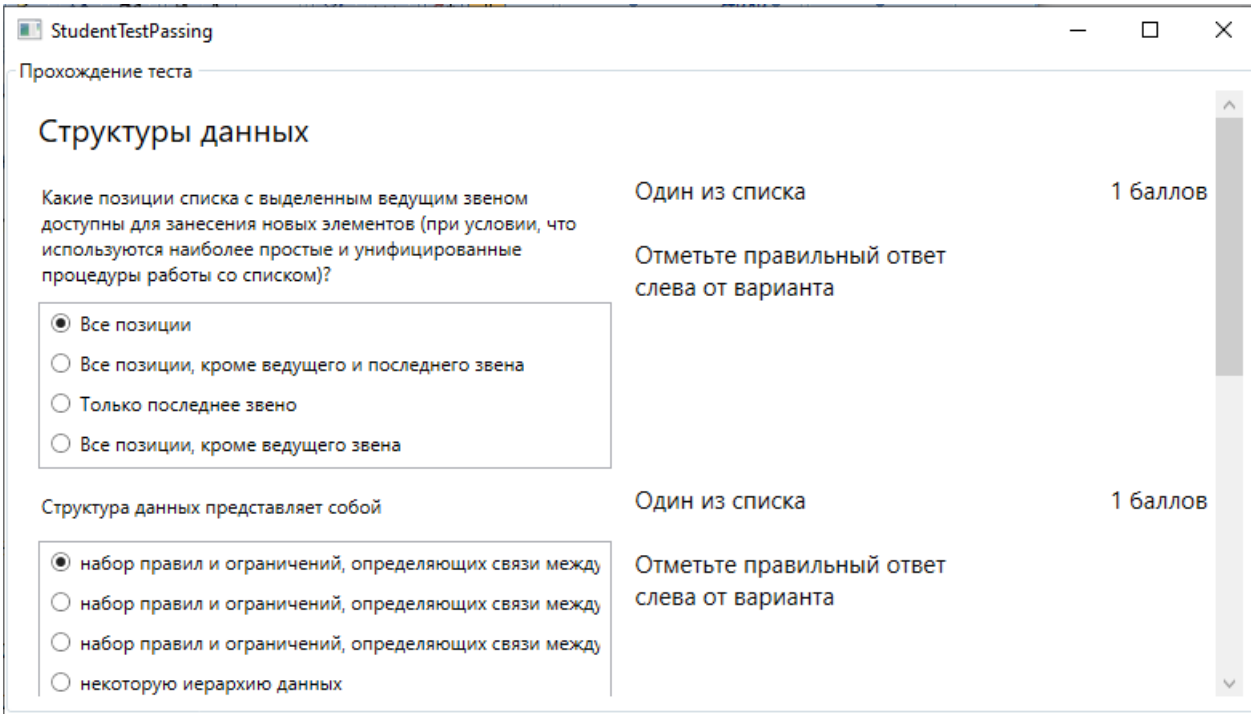
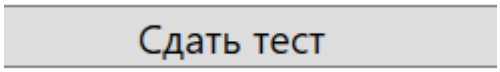


Рисунок 3.8 Сторінка студента. Проходження тесту

Нижньою частиною вікно передбачено здачу тесту. Після відповіді (рис 3.8), користувач натискає на кнопці 

та автоматично отримує результат (рис. 3.9).

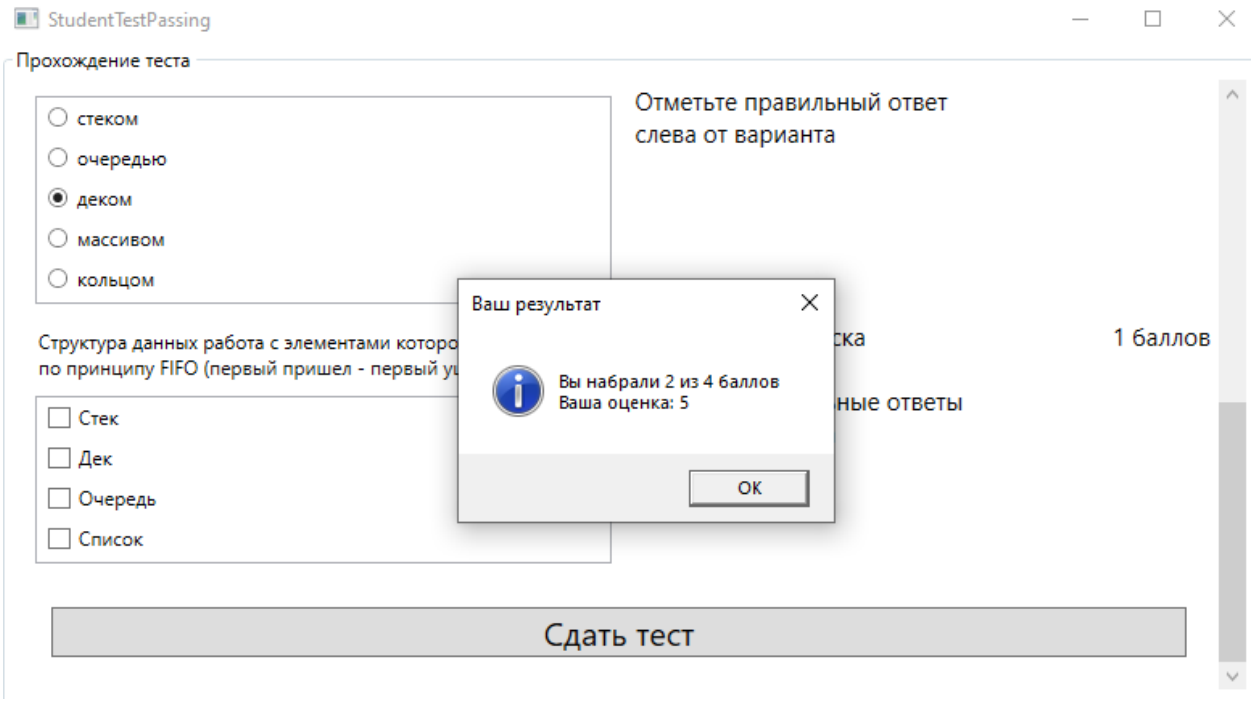
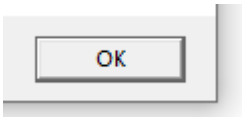


Рисунок 3.9 Сторінка студента. Результат тестування

Далі при натисканні на кнопці 

відбувається повернення до головного вікна програми (рис. 3.7), де є можливість обрати інший тест за темою (рис. 3.10).

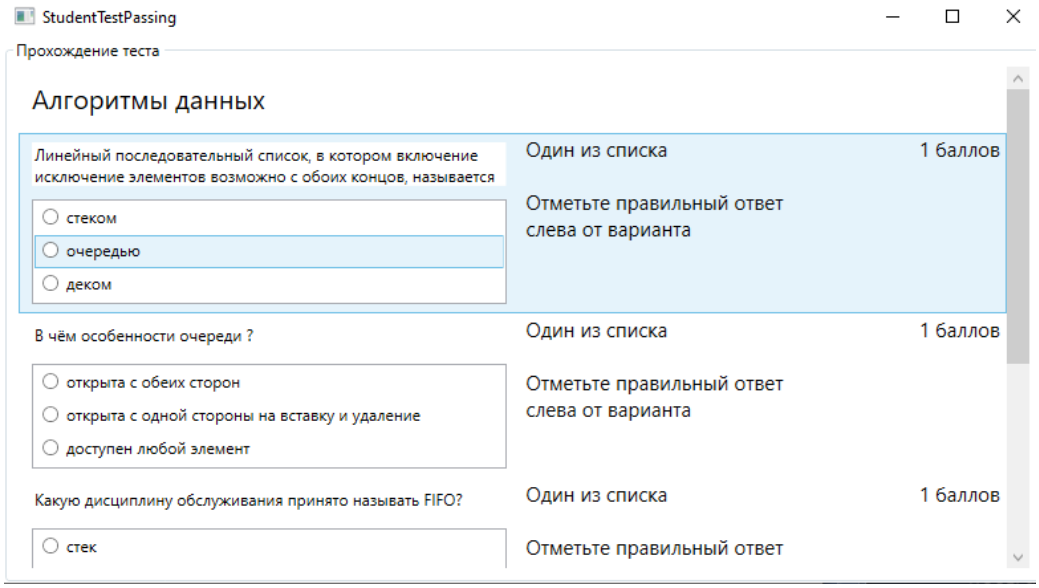


Рисунок 3.10 Інший тест

За для завершення роботи з програмою варто натиснути на  на головному вікні програмного додатку.

3.2 Тестування програмного додатку для навчання з курсу «Алгоритми та структури даних»

Тестування є одним з етапів життєвого циклу ПЗ, спрямованим на підвищення якісних характеристик. Програми, як об'єкти тестування, мають ряд особливостей, які відрізняють процес їх тестування від загальноприйнятого, застосовуваного при розробці апаратури та інших технічних виробів. Особливостями тестування програмних засобів є:

– складність програм і принципова неможливість вичерпного тестування;

– практична неможливість створення єдиної методики тестування (формалізація процесу тестування) в силу великої різноманітності програмних засобів за їх складності, функціонального призначенням, області використання.

3.2.1 Тестування програмних засобів

Налагодження даної програми проводилося у міру написання коду. На етапі компіляції виявлялися і усувалися синтаксичні помилки. Логічні помилки при реалізації усувалися тестуванням на етапі виконання.

3.2.2 Тестування програмних компонент

Для перевірки коректності програми в базу даних були внесені тестові завдання і була проведена перевірка правильності вибірки даних.

Результати тестування показали, що всі обчислення виконуються коректно, помилок при виконанні операцій не виявлено. При виникненні помилок при зверненні до бази даних, програма зберігає свою працездатність і повідомляє користувачеві про помилку.

Коректність при роботі з базою даних була перевірена шляхом виконання різних за типом і структурою запитів до бази даних та аналізом даних після виконання цих запитів.

3.3 Захист даних, резервне копіювання, захист від НДС

Основними принципами безпеки безпечних і захищених ПП можна назвати [3]:

Конфіденційність. Запобігання несанкціонованого доступу до інформації або програм.

Цілісність. Запобігання ушкодження, спотворення чи зміни інформації або програм.

Перевірка дійсності (аутентифікація). Посвідчення суб'єктів (користувачів або процесів) у мережі перед наданням доступу до даних, а також посвідчення об'єктів перед їх використанням.

Перевірка повноважень (авторизація). Забезпечення доступу до даних тільки тим суб'єктам, які були належним чином авторизовані і мають відповідні права (на читання, запис, зміна і т.д.).

Доступність. Забезпечення необхідною працездатністю та доступністю даних і програм.

Підзвітність (в деяких джерелах – доказ причетності, контроль). Встановлення зв'язку між користувачем (або об'єктом) і дією [5].

На практиці для забезпечення безпеки і захищеності ПП (в загальному випадку) застосовуються засоби і методи різних рівнів:

Нормативно-правовий. Рівень стандартів і законів, які регламентують відносини між особами, зацікавленими в розробці, експлуатації та розповсюдженні ПП [6]. Одним із засобів цього рівня можна назвати «ліцензійну угоду», з якою погоджується користувач ПП при її установці і використанні.

Адміністративно-організаційний. Даний рівень включає ряд заходів, в тому числі ті, які розробник ПП вважає за необхідне реалізувати в КМ користувача ПП, описує їх в документації до ПП (документи виду «Керівництво з безпечного встановлення та налаштування ПП») і передає користувачеві [3]. Виконання цих заходів, а також подальша підтримка продукту найчастіше лягає на плечі користувача. Адміністративно-організаційні заходи, як правило, дозволяють найбільш ефективно справлятися зі швидкістю зміни інформаційних технологій, зростанням числа вірусів, видів «хакерських» атак і т.д [8].

Економічний. Використання невисокої ціни для ПП достатньо часто є досить ефективним заходом для стримування обсягу «піратського» ринку. Багато користувачів, у випадку, якщо ціна ліцензійної копії продукту не багатьом перевершує вартість «піратської» копії, воліють купувати легальні копії продуктів [9].

Морально-етичний. Рівень включає в себе, наприклад, ряд заходів щодо посилення репутації легальних користувачів і дискредитації «чорних» - розповсюджувачі («піратів») неліцензійних копій ПП (приклади реалізації даних заходів: соціальна реклама в засобах масової інформації, конкурс компанії Microsoft на кращий літературний твір про чесно купленому програмному забезпеченні [4] і т.д.).

Фізичний. Рівень включає заходи фізичного розмежування доступу до ВУ, на яких інстальований ПП.

Програмно-технічний. Сюди відносяться різні електронні пристрої та спеціальні програми, які реалізують самостійно або в комплексі з іншими засобами наступні способи захисту [15]: ідентифікацію та аутентифікацію суб'єктів (користувачів, процесів і об'єктів), розмежування доступу до ресурсів КМ, контроль цілісності даних, забезпечення конфіденційності даних, реєстрацію та аналіз подій, що відбуваються в ПП.

Для попередження впливу загроз, слід застосовувати такі заходи:

а) для захисту інтересів споживачів від загроз з боку розробника ПП – незалежна компетентна експертиза (тестування) безпеки та захищеності рішень розробника;

б) для захисту інтересів розробника при поширенні власних програмних рішень – внутрішнє всебічне і компетентне тестування безпеки і захищеності ПП, до його публікації і розповсюдження.

В якості незалежної експертизи може застосовуватися процедура добровільної атестації об'єктів інформатизації за вимогами безпеки інформації. Так само в мережі Інтернет можна знайти приклади використання послуг комерційних організацій.

Майбутнє питань незалежної атестації об'єктів інформатизації пов'язують з новим стандартом ДСТУ ISO / IEC 15408-1-3:2017 «Інформаційна технологія. Методи і засоби забезпечення безпеки. Критерії оцінки безпеки інформаційних технологій »(національна версія міжнародного стандарту ISO / IEC 15408, загальновідомого під назвою" Common criteria ").

Висновки до розділу 3

Третім розділом даної дипломної роботи передбачено виконання експериментальної частини. У межах розділу розроблено керівництво користувача. Проведено тестування програмного додатку розробленого у рамках даної дипломної роботи. Результати тестування показали, що всі обчислення виконуються коректно, помилок при виконанні операцій не виявлено. Можна зробити висновок про ефективність розробленого програмного продукту та можливість його використання у навчальних цілях.

ВИСНОВКИ

Робота присвячена розробці системи навчання та перевірки знань з курсу «Алгоритми та структури даних».

1. Проведено дослідження, котрі обґрунтовують актуальність роботи та наукову новизну. А саме, рівень попиту системи навчання та перевірки знань, аналіз існуючих застосунків для навчання та перевірки знань з курсу «Алгоритми та структури даних», їхні переваги та недоліки. Було визначено, що існуючі застосунки розраховані на надання користувачеві теоретичних знань, їх вартісну складову та невідповідність дисципліні.

2. Враховуючи переваги та недоліки існуючих застосунків, було розроблено алгоритм для ефективного навчання та перевірки знань з курсу «Алгоритми та структури даних». Було визначено вимоги до застосунку, кількість розділів та їхні функціональні можливості.

3. Описано програмні засоби, які було застосовано для розробки програмного забезпечення. Визначено, що оптимальним рішенням є використання офіційного середовища розробки Visual Studio та мови програмування C#. Встановлено, що C# вміщує в собі більшість переваг сучасних мов програмування, проте не поступається швидкодією та якістю створеного програмного забезпечення більш відомим мовам.

4. Якість та працездатність застосунку підтверджено результатами проведеного тестування.

5. Використання розробленої моделі дасть можливість підвищити роботу освітніх установ, а тим самим знизить кількість часу витраченого викладачем для тестування (опитування) студентів та підвищить якість роботи викладачів. Також, рекомендується використовувати застосунок як доповнення до лекцій з «Алгоритми та структури даних» у навчальних закладах, для кращого засвоєння матеріалу.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Бармак О. В. Застосування інформаційної технології гнучкого тестування рівня знань у середовищі moodle / О. В. Бармак, О. В. Мазурець, А. О. Матвійчук // Вісник Хмельницького національного університету. Технічні науки. – 2017. – № 2. – С. 103-114. – Режим доступу: http://nbuv.gov.ua/UJRN/Vchnu_tekh_2017_2_24.

2. Нові інформаційні технології в освіті [Електронний ресурс]. – Режим доступу: http://ittehnolog.com/statti/novi-informatsiyni-tehnologiyi-navchannya/.

3. Кучерова А.Ю. Интеллектуальное оценивание знаний на базе образовательных учереждений / А.Ю. Кучерова. – Киев. 2018. − 262 с.

4. Комп'ютерна програма тестування знань OpenTEST 2 [Електронний ресурс]. − Режим доступу: http://opentest.com.ua/kompyuternaya-programma-testirovaniya-znanij-opentest-2/

5. Контрольно-діагностична система Test-W2 [Електронний ресурс]. – Режим доступу: http://teachinf.at.ua/load/programi/testi/test_w2_kontrolno_diagnostichna_sistema/16-1-0-7

6. Аналіз та дослідження інструментів автоматизації тестування при розробці комплексу програм автоматизованої системи конструювання розкладу занять / С.В. Дуденко, В.В. Калачова, С.В. Алексєєв, М.М. Колмиков // Збірник наукових праць Харківського національного університету Повітряних Сил. – 2015. – № 4(45). – С. 167-173.

7. Снитюк В. Е. Интеллектуальное управление оцениванием знаний / В. Е. Снитюк, К. Н. Юрченко. − Черкассы, 2013. − 262 с.

8. Даревич Р. Р. Підвищення ефективності інтелектуального аналізу тесту шляхом зважування понять моделі онтології / Р. Р. Даревич // Штучний інтелект. – 2005. – № 3. – С. 571–577.

9. Зайцева Л. В. Моделі та методи адаптивного контролю знань / Л. В. Зайцева, Н. О. Прокофьєва // Educational Technology & Society. – 2004. – № 7 (4). – С. 265–277.

10. Катеринчук И. С. Интеллектуальная система автоматизированного контроля знаний студентов высших учебных заведений / И. С. Катеринчук, Р. В. Рачок, В. В. Кравчук, В. М. Кулик // Информационные технологии в образовании : сборник научных трудов. – Херсон : Изд-во ХГУ, 2009. – Выпуск 4. – С. 139–147.

11. Moodle – вільна енциклопедія [Електронний ресурс]. − Режим доступу : http://uk.wikipedia.org/wiki/Moodle

12. Поліщук А. О. Інформаційна технологія автоматизації аналізу відповідності тестових завдань лекційним матеріалам навчальних дисциплін / А. О. Поліщук, О. В. Мазурець // Збірник наукових праць за матеріалами дев'ятої міжнародної науково-технічної конференції «Актуальні проблеми комп'ютерних технологій 2015». – Хмельницький, 2015. – С. 207–218.

13. Бармак О. В. Інформаційна технологія автоматизованого визначення термінів у навчальних матеріалах / О. В. Бармак, О. В. Мазурець // Вимірювальна та обчислювальна техніка в технологічних процесах. – Хмельницький, 2015. – № 2. – С. 94–102.

14. Особливості розробки плагінів до Moodle [Електронний ресурс]. − Режим доступу : https://docs.moodle.org/dev/Main_Page

15. Матвійчук А. О. Інформаційна технологія гнучкого тестування рівня знань / А. О. Матвійчук, О. В. Мазурець // Збірник наукових праць за матеріалами дев'ятої міжнародної науково-технічної конференції «Актуальні проблеми комп'ютерних технологій 2015». – Хмельницький, 2015. – С. 158–169.

16. Крак Ю. В. Практичне дослідження ефективності інформаційної технології автоматизованого визначення семантичних термінів в контенті навчальних матеріалів / Ю. В. Крак, О. В. Бармак, О. В. Мазурець // Прикладне програмне забезпечення : збірник наукових праць за матеріалами

десятої міжнародної науково-практичної конференції по програмуванню «УкрПРОГ’2016». – К., 2016. – С. 237–245.

17. Работа в системе дистантного обучения Moodle. Инструкция для преподавателей ./ В.В. Гвоздев, В.В. Проскурин/ Тольятти, 2011. – 155 с.

18. The ASP.Net Site [Електронний ресурс] – Режим доступу до ресурсу: http://www.asp.net/.

19. Entity Framework [Електронний ресурс] – Режим доступу до ресурсу: http://www.asp.net/entity-framework.

20. JQuery Site [Електронний ресурс] – Режим доступу до ресурсу: https://jquery.com

21. NET Core and Open-Source [Електронний ресурс] – Режим доступу до ресурсу: https://msdn.microsoft.com/en-us/library/dn878908 (v=vs.110).aspx.

22. Справочник «Паттерны проектирования» [Електронний ресурс] – Режим доступу до ресурсу: http://design-pattern.ru/patterns/mvc.html.

23. ADO.NET Blog [Електронний ресурс] – Режим доступу до ресурсу: http://blogs.msdn.com/b/adonet/.

24. Introduction to ASP.NET Identity [Електронний ресурс] – Режим доступу до ресурсу: http://www.asp.net/identity/overview/gettingstarted/introduction-to-aspnet-identity.

25. Inversion of Control Containers and the Dependency Injection pattern [Електронний ресурс] – Режим доступу до ресурсу: http://martinfowler.com/articles/injection.html.

26. What is Code-First? [Електронний ресурс] – Режим доступу до ресурсу: http://www.entityframeworktutorial.net/code-first/what-is-code-first.aspx.

27. Code First Migrations [Електронний ресурс] – Режим доступу до ресурсу: https://msdn.microsoft.com/en-us/data/jj591621.aspx.

28. Create, Retrieve, Update and Delete (CRUD) [Електронний ресурс] – Режим доступу до ресурсу: https://www.techopedia.com /definition/25949/createretrieve-update-and-delete-crud.

29. Поняття архітектури програмного забезпечення. [Електронний ресурс]. — Режим доступу до ресурсу: http://www.lib.mdpu.org.ua/ebook/web/Lec10.html

30. Сравнение многоуровневых и клиент-серверных систем [Електронний ресурс]. — Режим доступу до ресурсу: http://msdn.microsoft.com/ruru/library/ee895340.aspx

ДОДАТКИ

Додаток А

Файл TeacherTestsEditor

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace TestingSystem
{
   /// <summary>
   /// Interaction logic for TeacherTestsEditor.xaml
   /// </summary>
   public partial class TeacherTestsEditor : Window
   {
      SqlConnection sqlConnection;
      public TeacherTestsEditor()
      {
         InitializeComponent();
         string connectionString = Connection.connectionString;
         sqlConnection = new SqlConnection(connectionString);
         LoadTestsListBox();
      }

      private async void LoadTestsListBox()
      {
         //Открытие подключения
         await sqlConnection.OpenAsync();

         //Чтение Тестов
         SqlCommand sqlCommandSELECT = new SqlCommand($"SELECT * FROM
[Tests]", sqlConnection);
         SqlDataReader dataReader = null;
         try
         {
            dataReader = await sqlCommandSELECT.ExecuteReaderAsync();
            while (await dataReader.ReadAsync())
            {
```

```
                    AddTestToTestsListBox(Convert.ToString(dataReader["Name"]),
Convert.ToString(dataReader["Status"]));
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, ex.Source, MessageBoxButton.OK,
MessageBoxImage.Error);
            }
            finally
            {
                dataReader.Close();
            }
            sqlConnection.Close();

        }

        private void AddTestToTestsListBox(string testName, string status)
        {
            status = status.Trim(' ');
            //Добавление элемента списка тестов
            Grid grid = new Grid();
            grid.ColumnDefinitions.Add(new ColumnDefinition());
            grid.ColumnDefinitions.Add(new ColumnDefinition());
            grid.ColumnDefinitions.Add(new ColumnDefinition());
            grid.ColumnDefinitions.Add(new ColumnDefinition());
            grid.ColumnDefinitions.Add(new ColumnDefinition());
            grid.HorizontalAlignment = HorizontalAlignment.Stretch;
            TextBlock nameTextBlock = new TextBlock();
            nameTextBlock.FontSize = 20;
            nameTextBlock.TextWrapping = TextWrapping.Wrap;
            nameTextBlock.Text = testName;
            nameTextBlock.Margin = new Thickness(10);
            Grid.SetColumn(nameTextBlock, 0);
            Grid.SetColumnSpan(nameTextBlock, 4);
            grid.Children.Add(nameTextBlock);
            TextBlock statusTextBlock = new TextBlock();
            statusTextBlock.FontSize = 20;
            statusTextBlock.Text = status;
            if (status == "Открыт")
            {
                statusTextBlock.Foreground = new SolidColorBrush(Colors.Green);
            }
            else
            {
                statusTextBlock.Foreground = new SolidColorBrush(Colors.Red);
            }
            statusTextBlock.Opacity = 0.8;
            statusTextBlock.HorizontalAlignment = HorizontalAlignment.Right;
            statusTextBlock.VerticalAlignment = VerticalAlignment.Center;
            statusTextBlock.Margin = new Thickness(10);
            Grid.SetColumn(statusTextBlock, 4);
```

```
            grid.Children.Add(statusTextBlock);
            TestsListBox.Items.Add(grid);
        }

        private void AddTest_Click(object sender, RoutedEventArgs e)
        {
            new EditTest(TestsListBox) { Owner = this }.Show();
        }

        private async void OpenTest_Click(object sender, RoutedEventArgs e)
        {
            if (TestsListBox.SelectedIndex >= 0)
            {
                if (TestsListBox.Items[TestsListBox.SelectedIndex] is Grid grid)
                {
                    if (grid.Children[1] is TextBlock statusTextBlock)
                    {
                        string status = "Открыт";
                        statusTextBlock.Text = status;
                        statusTextBlock.Margin = new Thickness(10);
                        statusTextBlock.Foreground = new SolidColorBrush(Colors.Green);

                        if (grid.Children[0] is TextBlock testNameTextBox)
                        {
                            string testName = testNameTextBox.Text;
                            //Открытие подключения
                            await sqlConnection.OpenAsync();

                            //Обновление статуса
                            SqlCommand sqlCommandUPDATE = new SqlCommand($"UPDATE
[Tests] SET [Status]=@Status WHERE [Name]=@Name", sqlConnection);
                            sqlCommandUPDATE.Parameters.AddWithValue("Name", testName);
                            sqlCommandUPDATE.Parameters.AddWithValue("Status", status);
                            await sqlCommandUPDATE.ExecuteNonQueryAsync();
                            sqlConnection.Close();

                        }
                    }
                }
            }
        }

        private async void CloseTest_Click(object sender, RoutedEventArgs e)
        {
            if (TestsListBox.SelectedIndex >= 0)
            {
                if (TestsListBox.Items[TestsListBox.SelectedIndex] is Grid grid)
                {
                    if (grid.Children[1] is TextBlock statusTextBlock)
                    {
                        string status = "Закрыт";
                        statusTextBlock.Text = status;
```

```
                    statusTextBlock.Margin = new Thickness(10);
                    statusTextBlock.Foreground = new SolidColorBrush(Colors.Red);

                    if (grid.Children[0] is TextBlock testNameTextBox)
                    {
                        string testName = testNameTextBox.Text;
                        //Открытие подключения
                        await sqlConnection.OpenAsync();

                        //Обновление статуса
                        SqlCommand sqlCommandUPDATE = new SqlCommand($"UPDATE
[Tests] SET [Status]=@Status WHERE [Name]=@Name", sqlConnection);
                        sqlCommandUPDATE.Parameters.AddWithValue("Name", testName);
                        sqlCommandUPDATE.Parameters.AddWithValue("Status", status);
                        await sqlCommandUPDATE.ExecuteNonQueryAsync();
                        sqlConnection.Close();

                    }
                }
            }
        }
    }

    private async void DeleteTest_Click(object sender, RoutedEventArgs e)
    {
        if (TestsListBox.SelectedIndex >= 0)
        {
            if (TestsListBox.Items[TestsListBox.SelectedIndex] is Grid grid)
            {
                if (grid.Children[1] is TextBlock)
                {
                    if (grid.Children[0] is TextBlock testNameTextBox)
                    {
                        string testName = testNameTextBox.Text;
                        if(MessageBox.Show($"Удалить тест '{testName}'?", "Удаление
теста", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)
                        {
                            EditTest EditTestWindow = new EditTest(TestsListBox) { Owner =
this };

                            await EditTestWindow.ClearOldTestData(testName);
                        }
                    }
                }
            }
        }
    }

    private async void EditTest_Click(object sender, RoutedEventArgs e)
    {
        if (MessageBox.Show($"Если вы сохраните тест в режиме редактирования,
вы не сможете посмотреть результаты прохождения студентов на текущую версию
теста.\nЕсли вам нужны результаты прохождения этой версии теста, для выхода из
```

```
режима редактирования используйте кнопку \"Отмена\"", "Удаление теста", MessageBoxButton.OK, MessageBoxImage.Information) == MessageBoxResult.OK)
                {
                    if (TestsListBox.SelectedIndex >= 0)
                    {
                        if (TestsListBox.Items[TestsListBox.SelectedIndex] is Grid grid)
                        {
                            if (grid.Children[1] is TextBlock)
                            {
                                if (grid.Children[0] is TextBlock testNameTextBox)
                                {
                                    string testName = testNameTextBox.Text;
                                    EditTest EditTestWindow = new EditTest(TestsListBox) { Owner = this };
                                    EditTestWindow.Show();
                                    await EditTestWindow.LoadTest(testName);
                                }
                            }
                        }
                    }
                }

                private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)
                {
                    new Authorization().Show();
                }

                private async void CheckResults_Click(object sender, RoutedEventArgs e)
                {
                    if (TestsListBox.SelectedIndex >= 0)
                    {
                        if (TestsListBox.Items[TestsListBox.SelectedIndex] is Grid grid)
                        {
                            if (grid.Children[1] is TextBlock)
                            {
                                if (grid.Children[0] is TextBlock testNameTextBox)
                                {
                                    string testName = testNameTextBox.Text;
                                    int Tests_id = 0;
                                    //Открытие подключения
                                    await sqlConnection.OpenAsync();

                                    //Чтение id добавленного ТЕСТА в базе данных для удаления ВОПРОСОВ
                                    SqlDataReader dataReader = null;
                                    SqlCommand sqlCommandSELECT = new SqlCommand($"SELECT [Id] From [Tests] WHERE Name=N'{testName}'", sqlConnection);
                                    try
                                    {
                                        dataReader = await sqlCommandSELECT.ExecuteReaderAsync();
```

```
                        await dataReader.ReadAsync();
                        Tests_id = Convert.ToInt32(dataReader["Id"]);
                        dataReader.Close();
                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message, ex.Source, MessageBoxButton.OK,
MessageBoxImage.Error);
                    }
                    finally
                    {
                        dataReader.Close();
                    }

                    sqlConnection.Close();

                    TestResults TestResultsWindow = new TestResults() { Owner = this };
                    TestResultsWindow.Show();
                    await TestResultsWindow.LoadTest(Tests_id);
                }
            }
        }
    }
}
```

Файл StudentTestPassing

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace TestingSystem
{
   /// <summary>
   /// Interaction logic for StudentTestPassing.xaml
   /// </summary>
   public partial class StudentTestPassing : Window
   {
      SolidColorBrush brushWatermarkBackground = new SolidColorBrush(Colors.White);
      SolidColorBrush transparent = new SolidColorBrush(Colors.White);
      int Tests_id = 0;
      int Student_id = 0;
      int radioIndex = 0;
      int allTestScore = 0;
      public ListBox TestsList = new ListBox();
      SqlConnection sqlConnection;

      public StudentTestPassing(ListBox TestsListbox, int Student_id)
      {
         InitializeComponent();
         TestsList = TestsListbox;
         string connectionString = Connection.connectionString;
         sqlConnection = new SqlConnection(connectionString);
         this.Student_id = Student_id;
      }

      public async Task LoadTest(string testName)
      {
         //Открытие подключения
         await sqlConnection.OpenAsync();

         //Чтение id добавленного ТЕСТА в базе данных для удаления ВОПРОСОВ
         SqlDataReader dataReader = null;
         SqlCommand sqlCommandSELECT = new SqlCommand($"SELECT * From [Tests] WHERE Name=N'{testName}'", sqlConnection);
```

```
                try
                {
                    dataReader = await sqlCommandSELECT.ExecuteReaderAsync();
                    await dataReader.ReadAsync();
                    Tests_id = Convert.ToInt32(dataReader["Id"]);
                    string Test_name = Convert.ToString(dataReader["Name"]);
                    TestName.Text = Test_name;
                    dataReader.Close();

                    sqlCommandSELECT = new SqlCommand($"SELECT * FROM [Questions] WHERE Tests_id=N'{Tests_id}'", sqlConnection);
                    dataReader = null;
                    List<int> Questions_ids = new List<int>();
                    List<string> Questions_names = new List<string>();
                    List<string> Questions_types = new List<string>();
                    List<int> Questions_points = new List<int>();
                    allTestScore = 0;
                    try
                    {
                        dataReader = await sqlCommandSELECT.ExecuteReaderAsync();
                        while (await dataReader.ReadAsync())
                        {
                            Questions_ids.Add(Convert.ToInt32(dataReader["Id"]));

Questions_names.Add((String)(Convert.ToString(dataReader["Name"])).Trim(' '));

Questions_types.Add((String)(Convert.ToString(dataReader["Type"])).Trim(' '));
                            Questions_points.Add(Convert.ToInt32(dataReader["Points"]));
                            allTestScore += Convert.ToInt32(dataReader["Points"]);
                        }
                    }
                    catch (Exception ex)
                    {
                        MessageBox.Show(ex.Message, ex.Source, MessageBoxButton.OK, MessageBoxImage.Error);
                    }
                    finally
                    {
                        dataReader.Close();
                    }
                    //цикл добавления вопросов
                    for (int i = 0; i < Questions_ids.Count; i++)
                    {
                        int QlistboxIndex = AddQuestion(Questions_types[i], Questions_points[i], Questions_names[i]);
                        ListBox answersList = null;
                        if (QListbox.Items[QlistboxIndex] is Grid grid)
                        {
                            if (grid.Children[1] is StackPanel stackPanel)
                            {
                                if (stackPanel.Children[0] is ListBox listBox)
                                {
```

```
                        answersList = listBox;
                    }
                }
            }
            string Text = null;
            string Correctness = null;

            //цикл добавления ответов для каждого вопроса
            sqlCommandSELECT = new SqlCommand($"SELECT * FROM [Answers]
WHERE Questions_id=N'{Questions_ids[i]}'", sqlConnection);
            dataReader = null;
            try
            {
                dataReader = await sqlCommandSELECT.ExecuteReaderAsync();
                while (await dataReader.ReadAsync())
                {
                    Text = Convert.ToString(dataReader["Text"]);
                    Correctness                                          =
(String)(Convert.ToString(dataReader["Correctness"])).Trim(' ');
                    if (Questions_types[i].Equals("Radio"))
                    {
                        if (answersList.Items.Count > 0)
                        {
                            if (answersList.Items[0] is Grid answerGrid)
                            {
                                if (answerGrid.Children[0] is RadioButton rb)
                                {
                                    answersList.Items.Add(AddRadioAnswer(rb.GroupName,
Text, Correctness));
                                }
                            }
                        }
                        else
                        {
                            answersList.Items.Add(AddRadioAnswer(radioIndex.ToString(),
Text, Correctness));
                            radioIndex++;
                        }
                    }
                    else if (Questions_types[i].Equals("Checkbox"))
                    {
                        answersList.Items.Add(AddCheckboxAnswer(Text, Correctness));
                    }
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message,     ex.Source,     MessageBoxButton.OK,
MessageBoxImage.Error);
            }
            finally
            {
```

```
                    dataReader.Close();
                }
            }
        }
        catch (Exception)
        {
        }

        sqlConnection.Close();
    }

    private int AddQuestion(string Type, int Points, string Text = "")
    {
        Grid grid = new Grid();
        grid.RowDefinitions.Add(new RowDefinition());
        grid.RowDefinitions.Add(new RowDefinition());
        grid.ColumnDefinitions.Add(new ColumnDefinition());
        grid.ColumnDefinitions.Add(new ColumnDefinition());

        Grid rightGrid = new Grid();
        Grid.SetRowSpan(rightGrid, 2);
        Grid.SetColumn(rightGrid, 1);

        if (Type.Equals("Radio"))
        {
            rightGrid.Children.Add(new TextBlock() { Text = "Один из списка\n\nОтметьте правильный ответ \nслева от варианта", Margin = new Thickness(10, 0, 10, 0), HorizontalAlignment = HorizontalAlignment.Left, FontSize = 15 });
        }
        else if (Type.Equals("Checkbox"))
        {
            rightGrid.Children.Add(new TextBlock() { Text = "Несколько из списка\n\nОтметьте правильные ответы \nслева от варианта", Margin = new Thickness(10, 0, 10, 0), HorizontalAlignment = HorizontalAlignment.Left, FontSize = 15 });
        }

        StackPanel stackPanelQ = new StackPanel();
        stackPanelQ.Margin = new Thickness(5);

        Grid stackPanelQGrid = new Grid();

        transparent.Opacity = 0;
        stackPanelQGrid.Background = brushWatermarkBackground;

        stackPanelQGrid.HorizontalAlignment = HorizontalAlignment.Stretch;
        stackPanelQGrid.VerticalAlignment = VerticalAlignment.Center;

        TextBlock textBlock = new TextBlock();
        textBlock.Margin = new Thickness(2, 1, 0, 0);
        textBlock.Text = Text;
        textBlock.MinHeight = 20;
        textBlock.TextWrapping = TextWrapping.Wrap;
```

```
            stackPanelQGrid.Children.Add(textBlock);
            stackPanelQ.Children.Add(stackPanelQGrid);
            grid.Children.Add(stackPanelQ);

            StackPanel stackPanelA = new StackPanel();
            stackPanelA.Margin = new Thickness(5);

            ListBox answersList = new ListBox();
            answersList.HorizontalContentAlignment = HorizontalAlignment.Stretch;
            answersList.SetValue(ScrollViewer.HorizontalScrollBarVisibilityProperty,
ScrollBarVisibility.Disabled);
            answersList.SelectionChanged += AnswersList_SelectionChanged;

            stackPanelA.Children.Add(answersList);
            grid.Children.Add(stackPanelA);
            Grid.SetRow(stackPanelA, 1);

            TextBlock points = new TextBlock();
            points.Text = Points.ToString() + " баллов";
            points.FontSize = 15;
            points.VerticalAlignment = VerticalAlignment.Top;
            points.HorizontalAlignment = HorizontalAlignment.Right;

            rightGrid.Children.Add(points);
            grid.Children.Add(rightGrid);

            QListbox.Items.Add(grid);
            return QListbox.Items.Count - 1;
        }

        private Grid AddCheckboxAnswer(string Text = "", string Correctness = "false")
        {
            Grid aGrid = new Grid();
            aGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width =
GridLength.Auto });
            aGrid.ColumnDefinitions.Add(new ColumnDefinition());

            CheckBox checkBox = new CheckBox();
            checkBox.Margin = new Thickness(0, 2, 2, 0);

            CheckBox gostCheckBox = new CheckBox();
            gostCheckBox.Margin = new Thickness(0, 2, 2, 0);
            gostCheckBox.Visibility = Visibility.Collapsed;
            if (Correctness.Equals("true"))
            {
                gostCheckBox.IsChecked = true;
            }

            TextBlock textBlock1 = new TextBlock();
            textBlock1.Margin = new Thickness(2, 1, 0, 0);
            textBlock1.Text = Text;
```

```
            textBlock1.MinHeight = 20;

            aGrid.Children.Add(checkBox);
            aGrid.Children.Add(textBlock1);
            aGrid.Children.Add(gostCheckBox);
            Grid.SetColumn(textBlock1, 1);

            return aGrid;
        }

        private Grid AddRadioAnswer(string group, string Text = "", string Correctness =
"false")
        {
            Grid aGrid = new Grid();
            aGrid.ColumnDefinitions.Add(new ColumnDefinition() { Width =
GridLength.Auto });
            aGrid.ColumnDefinitions.Add(new ColumnDefinition());

            RadioButton radioButton = new RadioButton();
            radioButton.Margin = new Thickness(0, 2, 2, 0);
            radioButton.GroupName = group;

            CheckBox gostCheckBox = new CheckBox();
            gostCheckBox.Margin = new Thickness(0, 2, 2, 0);
            gostCheckBox.Visibility = Visibility.Collapsed;
            if (Correctness.Equals("true"))
            {
                gostCheckBox.IsChecked = true;
            }

            TextBlock textBlock1 = new TextBlock();
            textBlock1.Margin = new Thickness(2, 1, 0, 0);
            textBlock1.Text = Text;
            textBlock1.MinHeight = 20;

            aGrid.Children.Add(radioButton);
            aGrid.Children.Add(textBlock1);
            aGrid.Children.Add(gostCheckBox);
            Grid.SetColumn(textBlock1, 1);

            return aGrid;
        }

        private async void SaveButton_Click(object sender, RoutedEventArgs e)
        {
            double studentScore = 0;

            //Подсчёт и сохранение результатов
            for (int i = 0; i < QListbox.Items.Count; i++)
            {
                ListBox answersList = null;
                int QPoints = 0;
```

```
                    if (QListbox.Items[i] is Grid grid)
                    {
                        if (grid.Children[1] is StackPanel stackPanel)
                        {
                            if (stackPanel.Children[0] is ListBox listBox)
                            {
                                answersList = listBox;
                            }
                        }
                        if (grid.Children[2] is Grid rightGrig)
                        {

                            if (rightGrig.Children[1] is TextBlock pointsTextBlock)
                            {
                                QPoints = Convert.ToInt32(pointsTextBlock.Text.Substring(0,
pointsTextBlock.Text.IndexOf(" ")));
                            }
                        }
                    }
                    bool addPoints = true;
                    for (int j = 0; j < answersList.Items.Count; j++)
                    {
                        if (answersList.Items[j] is Grid aGrid)
                        {
                            bool userAnswer = false;
                            bool correctAnswer = false;
                            if (aGrid.Children[0] is CheckBox checkBox)
                            {
                                userAnswer = (bool)checkBox.IsChecked;
                            }
                            else if (aGrid.Children[0] is RadioButton radioButton)
                            {
                                userAnswer = (bool)radioButton.IsChecked;
                            }
                            if (aGrid.Children[2] is CheckBox correctAnswerCheckBox)
                            {
                                correctAnswer = (bool)correctAnswerCheckBox.IsChecked;
                            }
                            if (userAnswer!= correctAnswer)
                            {
                                addPoints = false;
                                break;
                            }
                        }
                    }
                    if (addPoints)
                    {
                        studentScore += QPoints;
                    }
                }

                double mark = 10 * (studentScore / allTestScore);
```

```
                mark = Math.Round(mark);
                MessageBox.Show($"Вы набрали {studentScore} из {allTestScore} баллов \nВаша оценка: {mark} ", "Ваш результат", MessageBoxButton.OK, MessageBoxImage.Information);

                await sqlConnection.OpenAsync();
                SqlCommand sqlCommandINSERT = new SqlCommand("INSERT INTO [StudentsTestResults] (Tests_id, Students_id, StudentScore, AllTestScore, Mark)VALUES(@Tests_id, @Students_id, @StudentScore, @AllTestScore, @Mark)", sqlConnection);
                sqlCommandINSERT.Parameters.AddWithValue("Tests_id", Tests_id);
                sqlCommandINSERT.Parameters.AddWithValue("Students_id", Student_id);
                sqlCommandINSERT.Parameters.AddWithValue("StudentScore", studentScore);
                sqlCommandINSERT.Parameters.AddWithValue("AllTestScore", allTestScore);
                sqlCommandINSERT.Parameters.AddWithValue("Mark", mark);
                await sqlCommandINSERT.ExecuteNonQueryAsync();
                sqlConnection.Close();

                Owner.Show();
                this.Close();
            }

            private void AnswersList_SelectionChanged(object sender, SelectionChangedEventArgs e)
            {
                ListBox answersList = (ListBox)sender;
                if (answersList.Items[answersList.SelectedIndex] is Grid aGrid)
                {
                    if (aGrid.Children[0] is RadioButton radioButton)
                    {
                        radioButton.IsChecked = !radioButton.IsChecked;
                    }
                    else if(aGrid.Children[0] is CheckBox checkBox)
                    {
                        checkBox.IsChecked = !checkBox.IsChecked;
                    }
                }
            }

            private void Window_Closing(object sender, System.ComponentModel.CancelEventArgs e)
            {
                Owner.Show();
            }
        }
    }
```

Додаток В

ДИПЛОМНА РОБОТА

«РОЗРОБКА ПРОГРАМНОГО ДОДАТКУ ДЛЯ НАВЧАННЯ З КУРСУ «АЛГОРИТМИ ТА СТРУКТУРИ ДАНИХ» МОВОЮ C#»

Виконала: студентка 4 курсу,
групи ПД–42
спеціальності 121
Інженерія програмного забезпечення
Логункова А.І

Основні характеристики роботи

- **Метою** даної дипломної роботи виступає розробка програмного додатку для навчання з курсу «Алгоритми та структури даних».
- **Об'єктом дослідження** є підвищення ефективності вивчення та засвоєння навчального матеріалу з курсу «Алгоритми та структури даних».
- **Предметом дослідження** даної роботи є додаток для системи навчання та перевірки знань з курсу «Алгоритми та структури даних».

Актуальність дослідження

Тестування в педагогіці виконує три основні взаємопов'язані функції: діагностичну, навчальну і виховну:

- Діагностична функція полягає у виявленні рівня знань, умінь, навичок учня. Це основна, і найочевидніша функція тестування. За об'єктивності, широти і швидкості діагностування, тестування перевершує всі інші форми педагогічного контролю;
- Навчальна функція тестування полягає у мотивуванні учня до активізації роботи по засвоєнню навчального матеріалу. Для посилення навчальної функції тестування, можуть бути використані додаткові заходи стимулювання студентів, такі, як роздача викладачем примірного переліку питань для самостійної підготовки, наявність в самому тесті навідних запитань і підказок, спільний розбір результатів тесту.
- Виховна функція проявляється в періодичності й неминучості тестового контролю. Це дисциплінує, організовує і спрямовує діяльність учнів, допомагає виявити і усунути прогалини в знаннях, формує прагнення розвинути свої здібності.

Тестування – більш справедливий метод, воно ставить всіх учнів в рівні умови, як у процесі контролю, так і в процесі оцінки, практично, виключаючи суб'єктивізм викладача.

- Слід зазначити, що саме тестування поступово стає основною формою складання іспитів та виходить на перший план серед усіх рівнів навчання. Тому актуальність теми даного дослідження не викликає сумнівів.

Завдання дослідження

Для досягнення поставленої мети у роботі необхідно виконати низку завдань:

- здійснити дослідження об’єкта та виконати обґрунтування необхідності створення програмного додатку для навчання з курсу «Алгоритми та структури даних»;
- описати і обґрунтувати вибраний алгоритм щодо розв’язання задачі створення програмного додатку;
- провести опис-блок схеми вибраного алгоритму створення програмного додатку для навчання з курсу «Алгоритми та структури даних»; виконати вибір і обґрунтування вибору алгоритмічної мови програмування;

Завдання дослідження

- здійснити вибір технічного та програмного забезпечення;
- розробити проектні рішення по системі та її частинам;
- виконати проектування програмного додатку для навчання з курсу «Алгоритми та структури даних»;
- здійснити опис розробленого програмного додатку;
- розробити керівництво користувача;
- провести тестування розробленого програмного додатку для навчання з курсу «Алгоритми та структури даних».

Дослідження аналогів та прототипів

На сьогоднішній день існує безліч різноманітних програм, які виробляють перевірку знань учнів, по різних предметам навчання. Всі навчальні програми можу бути написані на різних мовах програмування: Delphi, C ++, C Sharp та ін.

На ринку навчальних програм представлена чимала кількість програмних засобів для вивчення різних предметів у тому числі іноземної мови. Більшість з них є у вільному доступі у Інтернет мережі. Розглянемо найпопулярніші з них:

- Сайт Premier Training Center пропонує пройти тест який приблизно оцінює рівень знань англійської граматики і словниковий запас.
- Наступною мережевою знахідкою став сайт NativeEnglish .За умовами сайту пропонується тест який перевіряє вміння правильно вживати неособисті форми дієслова (інфінітива) в англійській мові.
- Сайт Тесторіум «Тестування навчальних знань» навпаки є багатофункціональним та інтерактивним.
- Сайт «Тестування знань»- має вузьке направлення, загалом це тести з Javascript.

Постановка завдань дослідження

Метою роботи є розробка програмного додатку для навчання з курсу «Алгоритми та структури даних» мовою.

Використовувані на усних і письмових заліках та іспитах традиційні питання і завдання без переробки, як правило, не можуть бути використані в тестах. Для них спеціально розробляються завдання в тестовій формі. По трудомісткості, складності та важливості дану роботу можна порівняти з поетапною підготовкою підручника з нової дисципліни (написання лекцій, навчального посібника, підручника).

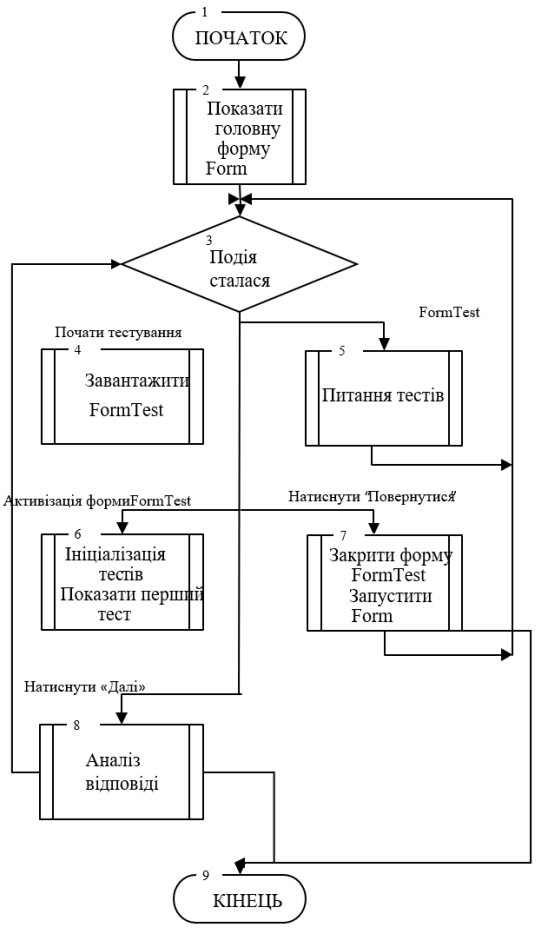
За змістом дій учня при контролі знань можна виділити завдання на:

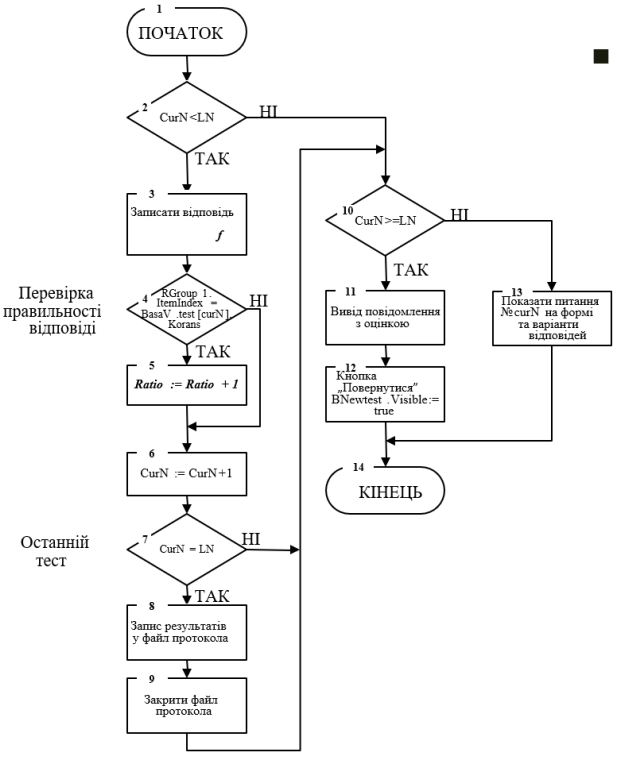
- вибір однієї відповіді;
- вибір декількох відповідей;
- встановлення (знаходження) відповідності між елементами двох множин;
- встановлення правильної послідовності в ряду пропонованих елементів;
- ранжування пропонованих елементів;
- заповнення пропусків, завершення пропозицій;
- підстановку;
- складання відповіді;
- обчислення відповіді;
- обчислення і вибір відповіді.

Тестові завдання можуть бути питальними, ствердними, текстовими, табличними, графічними (використовуються графіки, малюнки, діаграми).

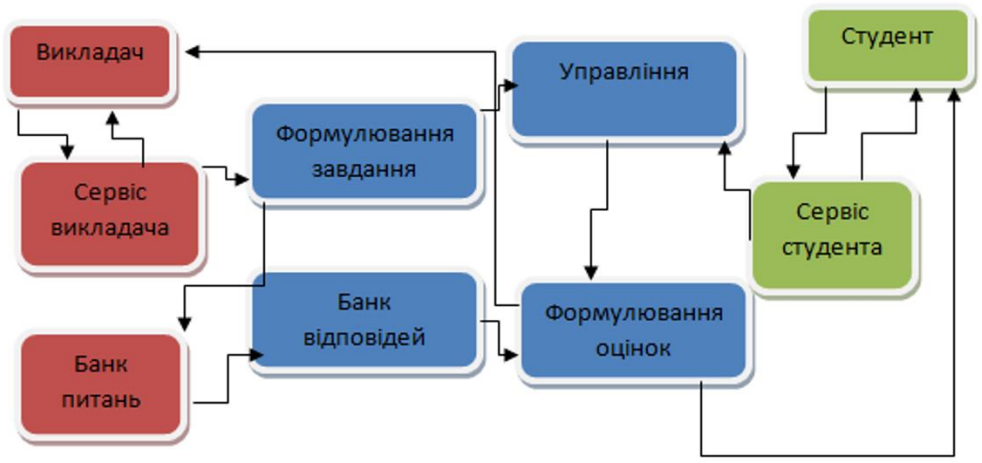
Алгоритм додатка

- Обробка подій екранних форм програми

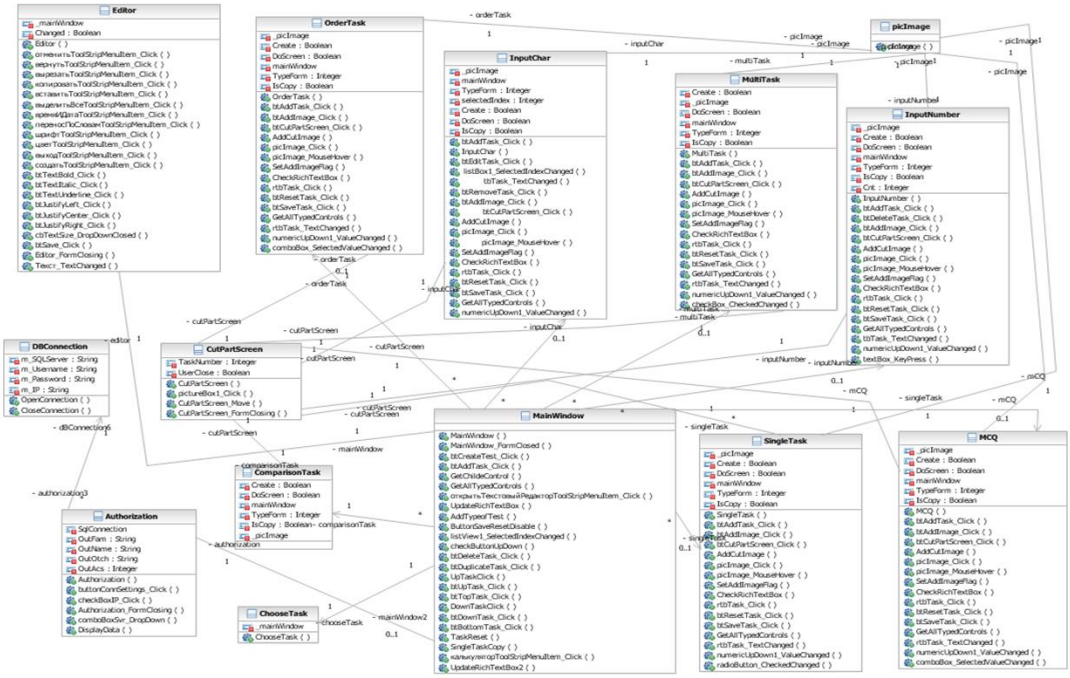




- Алгоритм процедури Click

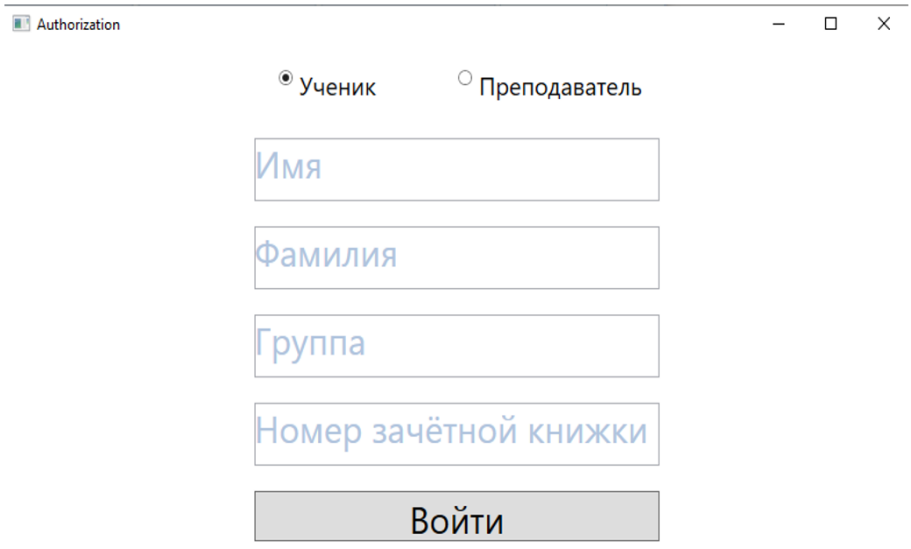


- Структурна модель системи тестування

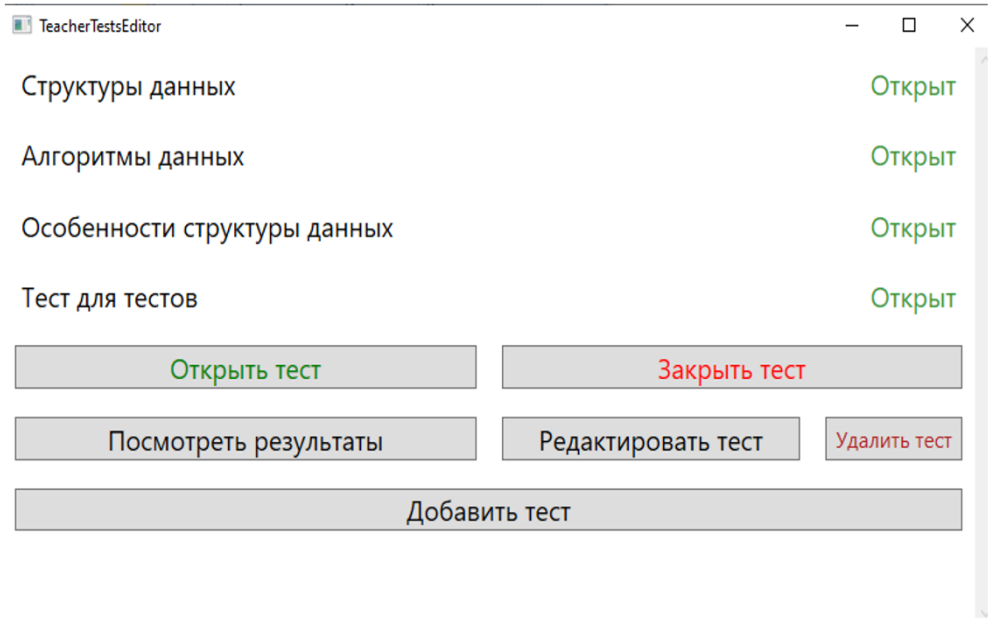


- Діаграма класів

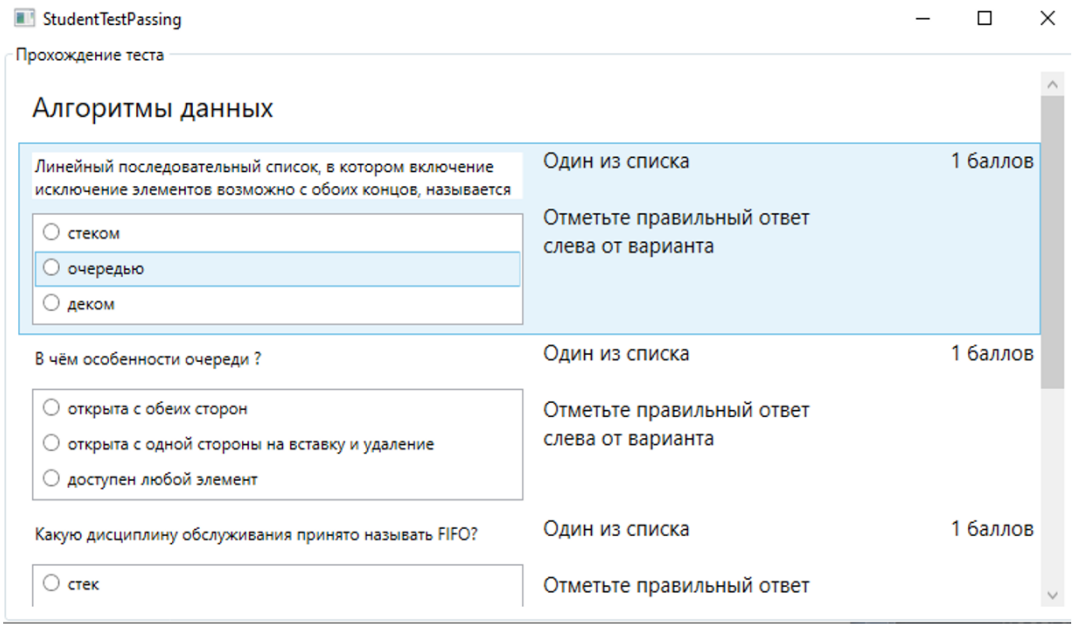
Результати роботи



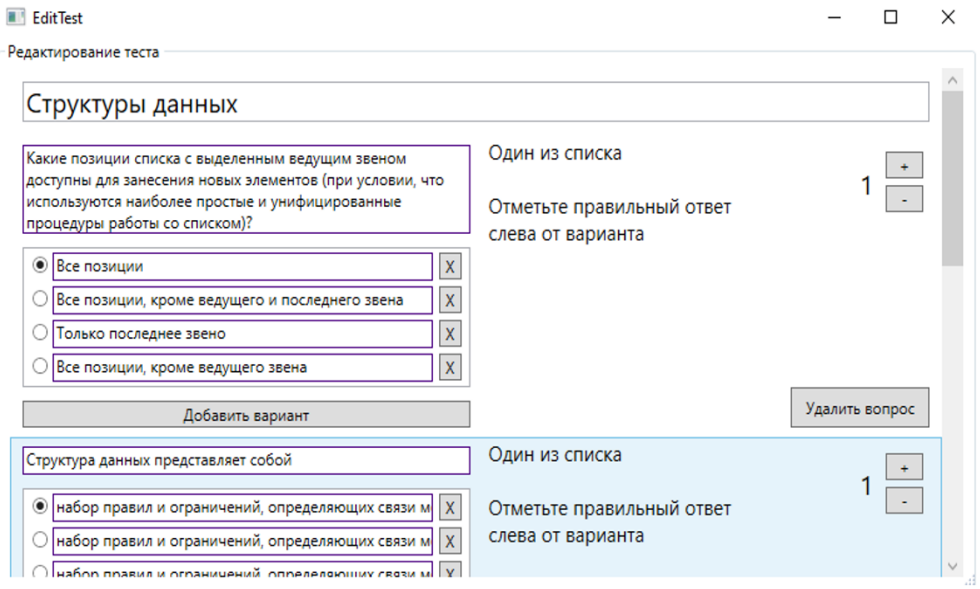
Результати роботи



Результати роботи



Результати роботи



ВИСНОВКИ

Робота присвячена розробці системи навчання та перевірки знань з курсу «Алгоритми та структури даних».

- Проведено дослідження, котрі обґрунтовують актуальність роботи та наукову новизну. А саме, рівень попиту системи навчання та перевірки знань, аналіз існуючих застосунків для навчання та перевірки знань з курсу «Алгоритми та структури даних», їхні переваги та недоліки.
- Враховуючи переваги та недоліки існуючих застосунків, було розроблено алгоритм для ефективного навчання та перевірки знань з курсу «Алгоритми та структури даних». Було визначено вимоги до застосунку, кількість розділів та їхні функціональні можливості.
- Описано програмні засоби, які було застосовано для розробки програмного забезпечення. Визначено, що оптимальним рішенням є використання офіційного середовища розробки Visual Studio та мови програмування C#.
- Якість та працездатність застосунку підтверджено результатами проведеного тестування.
- Використання розробленої моделі дасть можливість підвищити роботу освітніх установ, а тим самим знизить кількість часу витраченого викладачем для тестування (опитування) студентів та підвищить якість роботи викладачів.