Міністерство освіти і науки України
Державний університет «Одеська політехніка»
Навчально-науковий інститут комп'ютерних систем
Кафедра системного програмного забезпечення

Кіріязі Ігор Ігорович
студент групи АС-161

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

Програма для закріплення знань про алгоритми планування роботи процесів в операційній системі

Спеціальність:
121 – Інженерія програмного забезпечення
Освятня програма:
Інженерія програмного забезпечення

Керівник:
Зіноватна Світлана Леонідівна,
кандидат технічних наук, доцент

Одеса – 2021

ЗМІСТ

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ 4
ВСТУП 7
1 АНАЛІЗ ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ 12
1.1 Загальні відомості про навчальні програми 12
1.2 Планування роботи процесів в операційній системі 17
1.3 Огляд існуючих комп'ютерних навчальних програм 19
1.3 Висновки до розділу 25
2 ВИМОГИ ДО ПРОГРАМИ ДЛЯ ЗАКРІПЛЕННЯ ЗНАНЬ ПРО ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ 26
2.1 Опис користувачів 26
2.2 Опис варіантів використання 27
2.3 Нефункціональні вимоги 36
2.5 Висновки до розділу 37
3 ПРЕДСТАВЛЕННЯ ЗАВДАННЯ НА ЗАКРІПЛЕННЯ ЗНАНЬ ПРО АЛГОРИТМИ ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ 38
3.1 Загальний алгоритм виконання завдання в навчальній програмі ProcessStudy 38
3.2 Формалізоване представлення завдання для закріплення знань про планування роботи процесів 41
3.3 Висновки до розділу 42
4 ПРОЕКТУВАННЯ ПРОГРАМИ НАВЧАННЯ ВЗАЄМОДІЇ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ 43
4.1 Структура бази даних 43
4.2 Архітектура програмного продукту 55
4.3 Висновки до розділу 59
5 РЕАЛІЗАЦІЯ ПРОГРАМИ ДЛЯ НАВЧАННЯ ВЗАЄМОДІЇ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ 60
5.1 Опис методів класів 60

5.2 Опис інтерфейсу програми 65
5.3 Висновки до розділу 71
6 ТЕСТУВАННЯ ПРОГРАМИ ДЛЯ ОТРИМАННЯ ЗНАНЬ ПРО ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ 72
6.1 Опис тестових випадків 72
6.2 Експериментальне дослідження результатів роботи програми 75
6.3 Висновки до розділу 76
ВИСНОВКИ 77
СПИСОК ЛІТЕРАТУРИ 78
ДОДАТОК А КОД ПРОГРАМИ ..... **Ошибка! Закладка не определена.**

Міністерство освіти і науки України
Державний університет «Одеська політехніка»
Навчально-науковий інститут комп'ютерних систем
Кафедра системного програмного забезпечення

Рівень вищої освіти: другий (магістерський)
Спеціальність: 121 – Інженерія програмного забезпечення
Освятня програма: Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____проф. Любченко В.В.
"_____" _____ 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Кіріязі Ігору Ігоровичу, АС-161
(прізвище, ім'я та по батькові)

1. Тема проекту (роботи) Програма для закріплення знань про алгоритми планування роботи процесів в операційній системі
Керівник проекту Зіноватна Світлана Леонідівна, канд. техн. наук, доцент
затверджені наказом ректора від "25"жовтня 2021 р.№374-в
2. Строк подання студентом проекту (роботи) 30.11.2021
3. Вихідні дані по проекту (роботі) Технічне завдання
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належіть розробити)
Вступ. Вивчення особливостей предметної області. Вимоги до програми для закріплення знань про планування роботи процесів в операційній системі. Предствлення завдання на закріплення знань про алгоритми планування роботи процесів. Проектування програми навчання взаємодії процесів в операційній системі. Реалізація програми для навчання взаємодії процесів в операційній системі. Тестування програми для отримання знань про планування роботи процесів. Список літератури
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Мета та задачі роботи. Порівняння функціональних характеристик продуктів-аналогів. Діаграма варіантів використання. Схема алгоритму виконання контрольного завдання. Схема визначення оцінки за виконання завдання. Таблиця для відображення поведінки процесів. Схема визначення оцінки за виконання завдання. Концептуальна модель даних. Реляційна модель даних. Архітектура програмного продукту. Схема інтерфейсу відповідно архітектурі. Приклади віконних форм програми. Результати експериментального дослідження. Висновки

6. Консультанти розділів проекту (роботи)

Розділ	Консультант	Підпис, дата	
		Завдання видав	Результат прийняв

7. Дата видачі завдання 28.08.2021

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1.	Вивчення особливостей предметної області	07.09.2021	виконано
2.	Опис вимог	21.09.2021	виконано
4.	Створення бази даних	30.09.2021	виконано
5.	Створення алгоритмів для реалізації функцій програми	13.10.2021	виконано
6.	Кодування функцій програми	02.11.2021	виконано
7.	Тестування програмного продукту	12.11.2021	виконано
8.	Оформлення пояснювальної записки	15.11.2021	виконано

Здобувач вищої освіти _____ *І. І. Кіріязі*

Керівник роботи _____ *С. Л. Зіноватна*

РЕФЕРАТ

Метою роботи є скорочення часу на отримання знань про способи планування роботи процесів в операційній системі за рахунок автоматизованого тестування з використанням специфічних завдань.

В роботі досліджені особливості предметної області, описані вимоги до програмного продукту у вигляді варіантів використання з детальними сценаріями кожного з них. Спроектовано структуру даних та реалізовані засоби взаємодії програми та користувача.

Інтерфейс користувача реалізований засобами мов PHP та JavaScript. Інформація про завдання та результати їх виконання зберігаються у базі даних MySQL.

Ключові слова: автоматизований тест, планування роботи процесів, навчальна програма.

ABSTRACT

The aim of the work is to reduce the time to gain knowledge about how to plan the work of processes in the operating system through automated testing using specific tasks.

The peculiarities of the subject area are investigated in the work, the requirements to the software product in the form of use cases with detailed scenarios of each of them are described. The data structure is designed and the means of interaction between the program and the user are implemented.

The user interface is implemented using PHP and JavaScript. Information about tasks and their results are stored in a MySQL database.

Keywords: automated test, process planning, computer training program.

ВСТУП

Використання нових комп’ютерних технологій є одним з компонентів реформ у галузі освіти, який підвищує рівень якості освіти.

При використанні сучасних програмних засобів, мультимедійних технологій можливо «створювати електронні підручники, електронні лекційні курси, проводити дистанційне навчання в режимі реального часу» [2].

В [2] визначені чотири путі створення автоматизованих програм для навчання:

– пряме програмування на мовах високого рівня;

– використання інструментальних систем, за допомогою яких можна створити програмні засоби навчального призначення викладачу, незнайомому із програмуванням;

– використання готових навчальних програм для конкретної дисципліни;

– замовлення спеціалізованим організаціям на виготовлення навчальної програми.

В [1] перелічені вимоги до інструментальних систем у класі задач автоматизованого навчання:

– наявність засобів розробки тестів;

– наявність засобів проведення тестування, в локальній мережі або на віддаленому комп’ютері.

Тестування з використання комп’ютерних програм є важливим методом контролю отриманих знань студентів. «Використання комп’ютерів для контролю знань є економічно вигідним і забезпечує підвищення ефективності навчального процесу» [3].

Також в [3] зазначено, що «комп’ютерне тестування успішності дає можливість реалізувати основні дидактичні принципи контролю навчання: принцип індивідуального характеру перевірки й оцінки знань; принцип

системності перевірки й оцінки знань; принцип тематичності; принцип диференційованої оцінки успішності навчання; принцип однаковості вимог викладачів до студента».

В [4] зазначено, що автоматизований контроль знань:

1) дає економію часу для викладача, який за рахунок отриманого зворотного зв'язку не повторює положення, які вже засвоєні студентами, і, навпаки, може викласти положення, які засвоєні не в повній мірі;

2) постійний контроль рівня знань підвищує якість навчання за рахунок акцентування на складних положеннях дисципліни та відповідальність студентів за результати самостійної роботи.

Також переваги комп'ютерного тестування визначені в [5]:

– скорочення часу, витраченого на перевірку значного обсягу матеріалу для великих груп студентів;

– регулювання складність питань;

– виконання студентами самооцінку при підготовці до підсумкового контролю;

– формування об'єктивної оцінки;

– наявність зворотного зв'язку між студентом і викладачем

– формування статистичної інформації про результати контролю.

Таким чином, автоматизація контролю рівня знань дозволяє виключити людський фактор, але вона припускає формалізацію процесу, що не завжди можливо.

Можна виділити класичні види тестових завдань:

1) вибір одного правильного варіанта відповіді із кількості можливих;

2) встановлення відповідності;

3) встановлення послідовності;

4) завдання з короткою відповіддю.

Однак класичне тестування, навіть перенесене в електронне середовище, не завжди здатне оцінити професійні навички та вміння для предмета тестування [6].

В [7] перелічені класи комп’ютерних навчальних програм:

– спеціалізовані навчальні програми;

– комп'ютерні моделі;

– автоматизовані навчальні системи.

Спеціалізовані навчальні програми спеціально написані для надання допомоги учням і викладачеві в навчанні, призначені для навчання окремому питанню навчального курсу, вивчення якого-небудь окремого поняття, явища, факту [7]. Достоїнства: програма готова до застосування. Недоліки: велика відповідальність розроблювача через необхідність ураховувати освітні стандарти й типові навчальні програми; відсутність гнучкості, неможливість ніяких доробок.

З використанням комп'ютерної моделі вивчається деяке навчальне явище, наукове положення й т. д. Достоїнства: відсутність жорсткої послідовності дій, що дозволяє студентові виявити ініціативу в проведенні експериментів з моделлю. Недоліки: залежність від того, наскільки правильно модель відображає знання про предметну область.

Автоматизовані навчальні системи являють собою програмну оболонку, що припускає заповнення її предметним змістом. Достоїнства: можливість участі викладача у формуванні навчального матеріалу. Недоліки: жорсткі рамки створення завдань для перевірки знань тільки певного типу.

Таким чином, для закріплення та контролю знань в області поведінки операційної системи при планування роботи процесів необхідно мати програму, яка поєднує особливості таких типів навчальних програм, як комп'ютерні моделі та автоматизовані навчальні системи. Така програма повинна моделювати використання різних алгоритмів планування одночасної роботи процесів у комп’ютерній системі та дозволяти формувати викладачу завдання різного рівня складності.

Тобто задача автоматизованого тестування в області знань з внутрішньої роботи операційної системи є актуальною задачею.

Метою роботи є скорочення часу на отримання знань про способи планування роботи процесів в операційній системі за рахунок автоматизованого тестування з використанням специфічних завдань.

Для досягнення мети в роботі потрібно вирішити наступні задачі:

– визначити проблеми предметної області навчання питанням роботи операційної системи комп’ютера;

– проаналізувати можливості автоматизованих програм для навчання;

– розробити модель специфічного завдання для отримання знань про планування роботи процесів;

– реалізувати базу даних для зберігання необхідної інформації;

– розробити програмний інтерфейс для створення специфічних завдань та проведення тестування з використанням таких задач.

Об'єктом дослідження є програми для автоматизованого навчання.

Предметом дослідження є засоби формування та виконання завдань для автоматизованого навчання.

Методи дослідження. Модель специфічного завдання для отримання знань про планування роботи процесів базуються на принципах теорії множин.

Науково-практична цінність полягає в тому, що користувач отримує можливість формування специфічних завдань для оцінки знань про поведінку операційної системи при планування одночасної роботи декількох процесів.

Апробація результатів роботи. Результати роботи доповідалися й обговорювалися на XIV Міжнародної науково-практичної конференції «Інформаційні технології і автоматизація – 2021», Одеса, 21 - 22 жовтня 2021 р.

У першому розділі розглянуто поняття навчальної програми, описані її функції та можливості. Проаналізовано існуючі програмні засоби для організації навчання та контролю знань.

В другому розділі визначені три типи користувачів, які мають працювати з програмою. Для кожного з них писаний загальний функціонал. Наданий детальний опис варіантів використання відповідно вимогам до опису сценаріїв. Також описані характеристики системи у вигляді нефункціональних вимог.

В третьому розділі описано, яким чином відбувається обробка специфічного завдання для контролю знань студентом поведінки системи планування. Алгоритм визначає загальні кроки роботи програми для формування віконної форми, завдяки якої можливо сформувати відповідь студента. Також надано формальне представлення завдання в цілому з точки зору поведінки процесів, включених до завдання відповідно зазначеному алгоритму планування.

В четвертому розділі описано предметну область у вигляді концептуальної моделі даних, яка містить множину сутностей, їх атрибутів та взаємозв’язків. На основі отриманої моделі даних спроектовано та реалізовано реляційну базу даних, наданий детальний опис таблиць. Надано обґрунтування вибору шаблону архітектури та описано загальну схему реалізації функцій програми.

В п’ятому розділі описані методи окремих класів для обробки даних, які описують тести та їх складові. Також надані приклади віконних форм програми та описані їх основні компоненти.

У шостому розділі виконано тестування програми та експериментально підтверджено скорочення часу на отримання знань про планування роботи процесів в операційній системі.

Впровадження розробленої програми надасть можливість додавати нові завдання в базу даних задач для перевірки знань студентів про алгоритми планування роботи процесів в операційній системі та проводити тестування з використанням таких задач.

1 АНАЛІЗ ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальні відомості про навчальні програми

Відповідно [8] комп'ютерна навчальна програма - це програмний засіб, призначений для вирішення певних педагогічних задач, що має предметний зміст і орієнтоване на взаємодію з тим, яких навчають,

Також в [8, 9] перелічені основні педагогічні задачі, розв'язувані за допомогою електронних засобів навчання:

1) початкове ознайомлення із предметною областю, освоєння її базових понять і концепцій;

2) базова підготовка на різних рівнях глибини й детальності;

3) вироблення вмінь і навичок вирішення типових практичних завдань у даній предметній області;

4) вироблення умінь аналізу й прийняття рішень у нестандартні (нетипових) проблемних ситуаціях;

5) розвиток здатності до певних видів діяльності;

6) проведення учбово-дослідницьких експериментів з моделями досліджуваних об'єктів, процесів і середовища діяльності;

7) відновлення знань, умінь і навичок (для ситуацій, які нечасто зустрічаються, завдань і технологічних операцій);

8) контроль і оцінювання рівнів знань і вмінь.

Комп'ютерна навчальна програма повинна включати навчальний матеріал по певній предметній області (дисципліні, курсу, розділу, темі). Навчальний матеріал - це інформація, яка може мати декларативний характер (опис, ілюстрація), форму задач для контролю знань і вмінь або форму моделі, алгоритму, які представляють досліджувані об'єкти й процеси [8].

В [10] надано таке визначення: «Навчальна програма - це специфічний навчальний посібник, призначений для самостійної роботи учнів. Воно повинне сприяти максимальній активізації тих, хто навчається, індивідуалізуючи їхню роботу й надаючи їм можливість самим управляти

своєю пізнавальною діяльністю». Також відзначено, що навчальна програма, яка є частиною всієї системи навчання, повинна бути пов'язана з усім навчальним матеріалом , виконувати окремі специфічні функції.

Навчальні програми мають бути складені з урахуванням таких принципів програмованого навчання [10]:

– наявність мети навчальної роботи й алгоритму досягнення цієї мети;

– розподіленість навчальної роботи на кроки, пов'язані з відповідними дозами інформації, які забезпечують здійснення кроку;

– завершення кожного кроку самоперевіркою й можливим коригувальним впливом;

– використання автоматичного пристрою;

– індивідуалізація навчання (у достатніх і доступних межах).

За допомогою правильно побудованої навчальної програми можливе досягнення наступних результатів:

– уникнення монотонності завдань, враховувати зміну діяльності по її рівнях: дізнавання, відтворення, застосування;

– надання можливості успішної роботи з програмою студентам з різними рівнями підготовки;

– врахування фактору пам'яті (оперативної, короткочасної й довгострокової).

Відповідно [11], автоматизована навчальна система – це комп'ютерний педагогічний програмний засіб, який має такі характеристики:

– призначений для пред'явлення нової інформації;

– призначений для засвоєння навичок і вмінь,

– призначений для проміжного й підсумкового тестування;

– володіє розвитою системою допомоги, як по самій навчальній програмі, так і по досліджуваному предметі;

– володіє можливістю налаштування до того, кого навчають (рівня знань, швидкості й шляхів просування по досліджуваному матеріалі й т.д.);

– володіє розвитою системою збору й обробки статистичної інформації про кожному окремому студенту, групу або потік;

– накопичує інформацію про помилки, які часто зустрічаються при роботі з навчальною системою або по досліджуваній темі або дисципліні.

Відповідно [12] комп'ютерною навчальною програмою називається програма багаторазового застосування, спеціально розроблена або адаптована для реалізації педагогічної функції навчання або навчання при взаємодії з тим, кого навчають,. Програми такого типу чітко орієнтовані на комп'ютерну підтримку процесу одержання інформації й формування знань у якій-небудь області, закріплення навичок і вмінь, контролю або тестування знань. Навчальна програма повинна забезпечити реалізацію наступних педагогічних цілей: демонстрацію навчального матеріалу; тренінг у певній області; тестування й діагностику з метою контролю за ходом процесу навчання; властиво навчання.

Використання комп'ютерних технологій навчання в освіті призводить до підвищення ефективності викладання, а також може бути корисним для індивідуальної освіти осіб, що не мають по різних причинах можливості відвідування занять. Особливо важливе значення комп'ютерні технології можуть мати в післядипломному підвищенні кваліфікації й перепідготовці. «Цей вид освіти спирається на конкретні потреби професійно підготовлених фахівців, що займаються конкретною справою, що знають свої потреби й активно бажаючих підвищувати рівень освіти» [13].

Для забезпечення високої ефективності навчання до програмних навчальних засобів повинні пред'являтися певні вимоги [13]:

– до інтерфейсу користувача, який має враховувати зручне «розташування матеріалу на екрані, колірні та яскравістні виділення, способи і обсяги подання інформації в кожний момент часу й т. п. і визначальне сприйняття, стомлюваність того, якого навчають, і т. д.»;

– до можливостей настроювання навчання на індивідуальні особливості особи, що вчиться, можливість «встановлювати темп подання навчального

матеріалу з урахуванням можливостей його засвоєння, проводити розгалуження навчання для компенсації окремих індивідуальних пробілів»;

– до можливості використання програми для «рішення різних по глибині завдань розглянутого напрямку: від знайомства з ідеями, закладеними в об'єкт вивчення, до детального дослідження всіх особливостей (це дозволяє індивідуально давати тому, кого навчають, стільки, скільки він «може взяти», навіть трохи виходячи за рамки навчального плану для здатних учнів без додаткових витрат викладача)»,

– до структури побудови, що «забезпечує знайомство з основними теоретичними положеннями досліджуваної теми, активну практику, що дозволяє проводити не ознайомлювальну, а дослідницьку, аналітичну, творчу роботу, можливості самоконтролю одержаних знань і навичок»,

– до розумного сполучення фізичних і інтелектуальних дій того, кого навчають, що залежить від конкретних цілей і завдань при навчанні.

Моделювання - один із самих потужних методів пізнання навколишнього світу. цей метод застосовувався давно: при будуванні споруджень, для передбачення явищ природи, встановлення законів і т.п. Важко зараз назвати область діяльності людини, де б не застосовувалося моделювання [15].

Модель – це штучно створений об'єкт, що відтворює в певному виді реальному об'єкту-оригіналу.

Модель - це якийсь новий об'єкт, що відбиває істотні особливості досліджуваного об'єкта, явища або процесу. Загальні властивості моделей [16].

1) адекватність - це ступінь відповідності моделі тому реальному явищу (об'єкту, процесу), для опису якого вона будується,

2) кінцевість - модель відображає оригінал лише в кінцевому числі його відносин і, крім того, ресурси моделювання кінцеві,

3) спрощеність - модель відображає тільки істотні сторони об'єкта,

4) повнота - враховані всі необхідні властивості,

5) приблизність - дійсність відображається моделлю грубо або приблизно,

6) інформативність - модель повинна містити достатню інформацію про систему - у рамках гіпотез, прийнятих при побудові моделі,

7) потенційність - передбачуваність моделі й її властивостей.

В [16] надана класифікація моделей за різними ознаками, зокрема класифікація за областю використання є такою:

– навчальні - наочні приладдя, навчаючі програми, тренажери;

– дослідні моделі - зменшені або збільшені копії проектованого об'єкта, використовуються для дослідження об'єкта й прогнозування його майбутніх характеристик;

– науково-технічні моделі - для дослідження процесів і явищ;

– ігрові - військові, економічні, спортивні, ділові ігри.

– імітаційні моделі - не тільки відбивають реальність із певним ступенем точності, але імітують її, експеримент або багаторазово повторюється, щоб вивчити й оцініть наслідки яких-небудь дій на реальну обстановку, або проводиться одночасно з багатьма іншими схожими об'єктами, але поставленими в різні умови.

Комп'ютерна модель - подання інформації про моделюєму систему засобами комп'ютера.

Крім того, можливе оновлення існуючих програмних продуктів для вирішення завдань навчання [17].

Очевидно, що під час світової пандемії COVID-19 та повсюдного переходу на дистанційну форму освіти наявність комп’ютерних навчальних програм для будь-яких специфічних областей знань в окремих дисциплінах з можливістю моделювання та візуалізації поведінки реальних об’єктів є надзвичайно важливим фактором освіти.

Програма, розроблена в кваліфікаційній роботі, надає можливість відтворити одночасну роботу кількох процесів в операційній системі в рамках використання одного процесору для різних алгоритмів планування роботи процесів. Таким чином, розроблена програма відноситься до навчальних

програм з можливістю контролю знань, яка використовує комп’ютерне моделювання.

1.2 Планування роботи процесів в операційній системі

Планування - це робота з визначення того, в який момент перервати виконання одного процесу і якому процесу надати можливість виконуватися [18]. Тобто, задачами планування є:

– визначення моменту часу для зміни поточного активного потоку;

– вибір для виконання потоку з черги готових потоків;

Надалі описано можливих алгоритмів планування, які мають бути включені в модель навчальної програми [18, 19].

First-Come, First-Served (FCFS) - найпростіший алгоритм планування є алгоритм, коли процес переходить в стан готовність, він поміщається в кінець черги, вибір нового процесу для виконання здійснюється з початку.

Алгоритм здійснює планування, яке не витісняє.

Процес, що отримав у своє розпорядження процесор, займає його до закінчення поточного CPU burst. Після цього для виконання вибирається новий процес з початку черги.

Перевага алгоритму: легкість його реалізації.

Недолік: процеси з короткими CPU burst можуть довго очікувати черги.

Round Robin (RR) є модифікацією алгоритму FCFS. Кожен процес отримує від процесора фіксований квант часу. Якщо час безперервного використання процесора, необхідного процесу (залишок поточного CPU burst), менше або дорівнює тривалості кванта часу, то процес по звільняє процесор до закінчення кванта часу, на виконання надходить новий процес з початку черги. Якщо тривалість залишку поточного CPU burst процесу більше, ніж квант часу, то після закінчення цього кванта процес переривається таймером і поміщається в кінець черги процесів, готових до виконання, а процесор виділяється для використання процесу, що знаходиться в її початку.

Алгоритм здійснює планування, яке не витісняє.

На продуктивність алгоритму RR сильно впливає величина кванта часу.

Shortest-Job-First (SJF) передбачає вибір для виконання процес з мінімальною тривалістю CPU burst. Якщо ж таких процесів більше одного, то для вибору використовується алгоритм FCFS. Квантування часу при цьому не застосовується.

SJF-алгоритм короткострокового планування може бути витісняючим та невитісняючим. При невитісняючому SJF-планування процесор надається обраному процесу на весь необхідний йому часі. При витісняючому SJF-планування враховується поява нових процесів у черги готових до виконання (з числа знову народилися або розблокованих) під час роботи обраного процесу. Якщо CPU burst нового процесу менше, ніж залишок CPU burst у виконуваного, то виконуваний процес витісняється новим.

Гарантоване планування гарантує, що кожен із користувачів матиме в своєму розпорядженні частину процесорного часу, яка є залежною від кількості користувачів, що працюють у системі. для процесів кожного користувача обчислюється значення коефіцієнта справедливості. Черговий квант часу надається готовому процесу з найменшою величиною коефіцієнта справедливості.

Недоліки алгоритму: неможливість передбачити поведінку користувачів, користувач може не запускати процеси довгий час, потім його процеси будуть отримувати багато процесорного часу.

Пріоритетне планування - кожному процесу присвоюється певне числове значення - пріоритет, у відповідності з яким йому виділяється процесор. Процеси з однаковими пріоритетами плануються в порядку FCFS.

Планування з використанням пріоритетів може бути витісняючим та невитісняючим. При витісняючому плануванні процес з більш високим пріоритетом, який з'явився в черзі готових процесів, витісняє виконуваний процес з більш низьким пріоритетом. У разі невитісняючого планування процес стає на початок черги готових процесів.

Недолік алгоритму пріоритетного планування полягає в тому, що процеси з низькими пріоритетами можуть не запускатися довгий час. Рішенням такої проблеми може бути збільшення з часом значення пріоритету процесу, що перебуває у стані готовність, що гарантує, що процес в розумні терміни отримує право на виконання.

1.3 Огляд існуючих комп'ютерних навчальних програм

Платформа ITVDN. Містить велику кількість курсів для вивчення різних направлень в галузі IT (рис.1.1). Платформа ITVDN - це освітній online ресурс для IT-спеціалістів. Мета проекту - навчання мовам програмування й інформаційним технологіям усіх, хто хоче стати професіоналом у сфері розробки програмного забезпечення, проектування складних програмних систем, веб-розробки й у суміжних областях [11].

Навчання на ITVDN проводиться у формі окремих курсів, що дозволяє проходити навчання послідовно з нуля або вибірково по окремих технологіях для підвищення рівня кваліфікації фахівців.

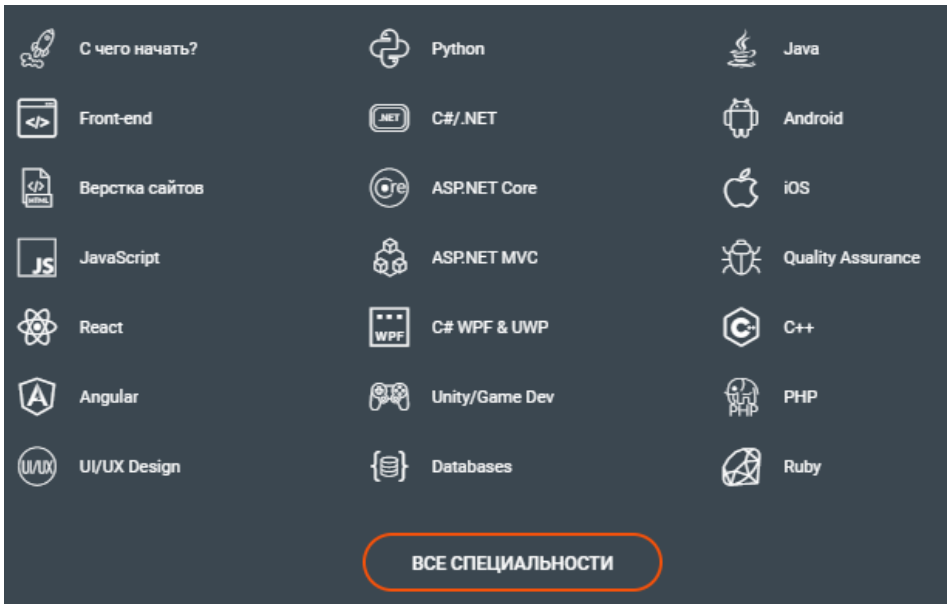

Рисунок 1.1 – Пропозиції напрямків IT на платформі ITVDN

Навчання передбачає перегляд відеокурсів, навчання з тренером, інтерактивний практикум, вебінари, навчання співробітників компаній.

До кожного уроку дається три спроби на проходження тестування й дві спроби на підсумковий тест за всім курсом. Поурочне тестування призначене для тренування й розуміння того, наскільки засвоєний пройдений матеріал. При успішній здачі підсумкового тесту (більше 70%) формується електронний сертифікат.

Платформа є комерційною, має досить високу вартість (рис.1.2) [11].

ЧТО ВХОДИТ В ПОДПИСКУ	1 КУРС	СТАРТОВЫЙ	БАЗОВЫЙ	ПРЕМИУМ
Доступ к видеокурсам ITVDN	1 курс	Все специальности и курсы	Все специальности и курсы	Все специальности и курсы
Время доступа	30 дней	3 мес	6 мес	12 мес
Скачивание учебных материалов	✓	✓	✓	✓
Практикум	✓	✓	✓	✓
Тестирование	1 тест	10 тестов	16 тестов	24 тестов
Электронный сертификат	✓	✓	✓	✓
Проверка домашнего задания	–	5 заданий	10 заданий	20 заданий
Консультация с тренером	–	30 мин	60 мин	120 мин
Возможна оплата частями	–	–	✓	✓
СТОИМОСТЬ	264.93 UAH	1325.73 UAH	2386.53 UAH	4508.13 UAH

Рисунок 1.2 – Вартість навчання на платформі ITVDN

Інструменти Microsoft. Компанія Microsoft надає інструменти та безкоштовні ресурси Microsoft, які допомагають у підготовці до занять, викладанні, оцінюванні, веденні обліку та проведенні аналізу. До таких засобів відносяться Sway, OneNote та Teams [20].

Викладач та учні можуть створювати у Sway високоякісні, інтерактивні цифрові історії за короткий час, використовувати при цьому свої зображення, тексти, відео та інші мультимедійні матеріали. Застосунок допомагає створювати професійні інтерактивні презентації на основі будь-якого контента [21]. Sway є безкоштовним для приватних користувачів з функціями

професійного оформлення та анімації, простого додавання контенту, необмеженої кількості презентацій. Також Sway входить до Office 365.

Блокнот OneNote дозволяє організувати роботу для класу, вирізняється персональними робочими областями, бібліотекою контенту та середовищем, яке заохочує учнів до співпраці. OneNote дозволяє упорядкувати плани уроків і матеріали курсів у власному цифровому блокноті, щоб викладач міг відстежувати завдання та занотовувати свої думки, плани, ідеї [22].

В [23] представлені переваги долучення блокноту до навчальної діяльності (рис.1.3):

– організація – OneNote є універсальним й адаптується до будь-яких переваг ведення заміток;

– доступ, спільна робота й обмін контентом - учні можуть вчитися після закінчення уроків;

– представлення контента - можна додавати текст, зображення, аудіо, відео й цифрові рукописні дані на будь-яку сторінку;

– можливість гнучкого стилю навчання - вибір найкращого методу навчання для конкретної ситуації;

– безладдя із цифровим чорнилом - даються інструменти у вільній формі;

– бездротова презентація - можливість вільно переміщатися по кімнаті, забезпечуючи більш ефективне й гнучке середовище навчання;

– навчання на різних пристроях - доступ до своїх заміток на різних пристроях;

– пошук і додавання тегів - можливість витягати інформацію за бажанням, коли й де вона потрібна, оскільки замітки доступні для пошуку; також можна позначити свої замітки готовими або особисто створеними тегами, щоб швидко знаходити потрібні замітки;

– суперблокнот із записних книжок - не потрібно носити із собою фізичні папки;

– можливість вчити світ - можливість писати й малювати цифровим чорнилом, вести особисту записну книжку, поділитися записною книжкою з іншими людьми, записати звуковий коментар для студента, написати складне математичне вираження "від руки", вмонтувати онлайн-відео для уроку, індивідуалізувати інструкції за допомогою Microsoft Forms і настроїти тест із автоматичною оцінкою для перевірки, вбудовувати мультимедійний контент.

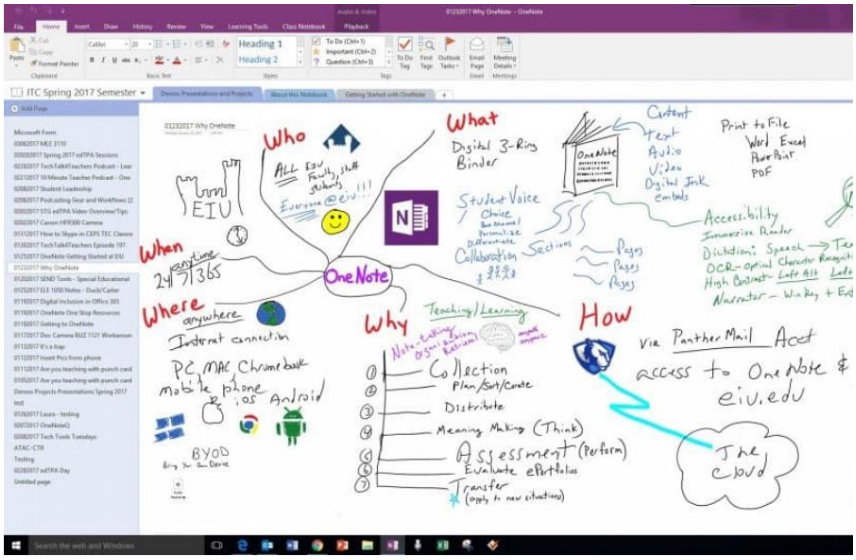

Рисунок 1.3 – Приклад використання OneNote [19]

Керувати роботою в класі з долученням кожного учня можливо з використанням з Microsoft Teams, який дозволяє [24]:

– збирати на зустрічі до 300 студентів або учасників спільноти безкоштовно;

– отримувати доступ до постійного чату для забезпечення зв’язку для навчання чи роботи;

– організовувати заняття й завдання, співпрацювати, обмінюватися файлами, користуватися матеріалами класу в єдиному рішенні;

– користуватися аналітикою для установи й групи.

Всі ці продукти дають можливість оптимізувати організацію завдань та виставлення оцінок. Може надавати аналітику даних для оцінювання

успішності учнів, виявлення проблемних учнів, покращення навчання та оптимізації ресурсів.

Інструмент Google Class Room. Google Клас допомагає викладачам розподіляти завдання й ефективно взаємодіяти з учнями. Цей сервіс доступний у веб-інтерфейсі й на мобільних пристроях. У Класі також настроєна інтеграція з багатьма сервісами Google, наприклад Gmail, Google Документами й Google Календарем [25].

Можливості інструменту Google Class Room наведені у табл.1.1.

Таблиця 1.1 – Можливості класу [25]

Категорія користувачів	Можливості
Викладачі	Проведення відеозустрічей. Створення курсів, завдань і керування ними, робота з оцінками в режимі онлайн. Додавання матеріалів до завдань, наприклад відео YouTube, форми Google, опитування й інші об'єкти з Диска. Надання коментарів і відкликань учням у режимі реального часу. Публікація оголошень і питань для учнів у стрічці курсу.
Учні	Відстеження й виконання завдань. Перевірка робіт на унікальність, одержання коментарів і оцінок викладача. Обмін інформацією й спілкування в стрічці курсу або по електронній пошті.
Адміністратори	Захист даних і настроювання дозволів для користувачів. Настроювання курсів і списків учнів. Додавання й видалення учнів і викладачів у рамках курсів. Цілодобова підтримка.

Проаналізовано можливості роботи програмних засобів для проведення навчання та виконано порівняння їх характеристик з характеристиками програми, яка розроблена в кваліфікаційній роботі (табл. 1.2).

Таблиця 1.2 – Порівняння характеристик програмних засобів для проведення навчання

Характеристики	Програма			
	ITVDN	Microsoft	Google Class Room	ProcessStudy
Тестування знань	+	+	+	+
Можливість створення власних завдань	-	+	+	+
Можливість формування спеціальних задач з моделлю	-	-	-	+
Режим роботи	веб, мобільні пристрої	веб, мобільні пристрої	веб, мобільні пристрої	веб, мобільні пристрої
Цілодобова праця	+	+	+	+
Спеціалізація	Програмування	Універсальна	Універсальна	Операційні системи
Вартість	Платна	Платна	-	-

Таким чином, можна зробити висновок, що розробка навчальної програм з можливістю контролю знань з відтворенням моделі вузько спеціалізованої предметної області, є актуальною задачею.

1.3 Висновки до розділу

В розділі розглянуто поняття навчальної програми, описані її функції та можливості.

Проаналізовано існуючі програмні засоби для організації навчання та контролю знань.

Таким чином, метою роботи є скорочення часу на отримання знань про способи планування роботи процесів в операційній системі за рахунок автоматизованого тестування з використанням специфічних завдань.

Для досягнення мети в роботі потрібно вирішити наступні задачі:

– визначити проблеми предметної області навчання питанням роботи операційної системи комп’ютера;

– проаналізувати можливості автоматизованих програм для навчання;

– розробити модель специфічного завдання для отримання знань про планування роботи процесів;

– реалізувати базу даних для зберігання необхідної інформації;

– розробити програмний інтерфейс для створення специфічних завдань та проведення тестування з використанням таких задач.

2 ВИМОГИ ДО ПРОГРАМИ ДЛЯ ЗАКРІПЛЕННЯ ЗНАНЬ ПРО ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ

2.1 Опис користувачів

Працювати з системою можуть три типи користувачів: Адміністратор, Викладач, Студент.

Користувач має бути ідентифікований в системі за допомогою імені та паролю.

На початку роботи системи існує єдиний користувач типу Адміністратор з заданим за замовчуванням іменем та паролем.

Після впровадження системи ім’я та пароль адміністратора має бути зміненими.

Адміністратор може додавати в систему нових користувачів типу Викладач.

Викладач може додавати в систему нових користувачів типу Студент.

Викладач може створювати групи студентів, до яких додаються конкретні студенти. Один студент може входити до кількох груп.

Викладач може створювати тест, до якого може бути включено кілька питань. Питання можуть бути різного типу: з одним варіантом відповіді, з кількома варіантами відповіді, з короткою відповіддю, специфічна задача на планування роботи процесів. Питання може входити до кількох тестів.

До тесту можуть входити питання, створенні різними викладачами. Викладач може дозволити використовувати свої запитання спеціальною дією.

Для тесту можуть бути вказані групи та терміни дії. Тест може бути навчаючим або контролюючим. Для контролюючого тесту може бути завдана кількість спроб.

Для кожного питання може бути задана кількість балів за правильну відповідь. Після проходження тесту система розраховує отриманий студентом бал, який може бути переглянутий студентом та викладачем. Для

опублікування балу студентам може бути вказаний термін – відразу після проходження тесту або після закриття тесту.

Студент може ввійти в систему для виконання тесту, який відкритий для групи. до якої студент входить.

Викладач може переглянути бали. отримані конкретним студентом, отримати відомість для всієї групи та отримати статистику по всім або окремим групам відповідей: за питанням, за тестом, за групою.

2.2 Опис варіантів використання

Варіант використання - це письмовий опис того, як користувачі будуть виконувати завдання в програмі. Він описує, з погляду користувача, поводження системи при відповіді на запит. Кожний варіант використання представлений як послідовність простих кроків, починаючи з мети користувача й закінчуючи її досягненням. Варіант використання надає список цілей, якому можна використати для визначення вартості й складності системи [26].

Відповідно [27], варіант використання - це методологія, використовувана в системному аналізі для виявлення, уточнення й організації системних вимог. Варіант використання складається з набору можливих послідовностей взаємодій між системою й користувачем в конкретному середовищі й пов'язаний з конкретною метою. У результаті створюється документ, у якому описані всі кроки, початі користувачем для виконання дії.

Сценарії використання зазвичай пишуться бізнес-аналітиками й можуть використатися на декількох етапах розробки програмного забезпечення:

- планування системних вимог;
- перевірка дизайну;
- тестування програмного забезпечення;
- створення схеми для інтерактивної довідки й посібника користувача.

Документ варіанта використання може допомогти групі розроблювачів визначити й зрозуміти, де можуть виникнути помилки під час транзакції, щоб вони могли їх виправити.

Кожний варіант використання містить три основних елементи [27]:

– Актор- користувач системи, може бути одна людина або група людей, взаємодіючих із процесом;

– Ціль - остаточний успішний результат, що завершує процес;

– Система - процес і кроки, початі для досягнення кінцевої мети, включаючи необхідні функціональні вимоги і їхнє очікуване поводження.

На рис. 2.1 відображені варіанти використання для програми для закріплення знань з поведінки системи планування роботи процесів операційної системи.

Після створення первісного візуального списку учасників і варіантів використання у вигляді діаграми, можна створити початкову сітку варіантів використання, у якій кожний варіант використання має різні атрибути, що ставляться як до самого варіанта використання, так і до проекту. На рівні проекту ці атрибути включають, наприклад, складність, статус і пріоритет [28]. В табл. 2.1 виконано індексацію варіантів використання (ВВ), де складність визначається трьома категоріями: висока (В), середня (С), низька (Н), пріоритет є вищим для чисел з меншим значенням.

Таблиця 2.1 - Індексація варіантів використання

Ідентифі-катор ВВ	Назва ВВ	Актор	Склад-ність	Пріо-ритет
1.	Додавання викладача	Адміністратор	С	1
2.	Авторизація	Адміністратор, Викладач, Студент	С	1
3.	Редагування даних викладача	Адміністратор	С	2

Продовження табл. 2.1.

Ідентифікатор ВВ	Назва ВВ	Актор	Складність	Пріоритет
4.	Зміна паролю	Викладач, Студент	Н	2
5.	Додавання тесту	Викладач	С	1
6.	Проходження тесту	Студент	В	1
7.	Перегляд оцінки	Студент	С	2
8.	Редагування тесту	Викладач	С	3
9.	Видалення тесту	Викладач	Н	2
10.	Додавання питання	Викладач	В	1
11.	Видалення запитання з тесту	Викладач	Н	2
12.	Редагування дозволу на використання питань	Викладач	Н	3
13.	Додавання студента	Викладач	С	1
14.	Додавання запитання до тесту	Викладач	Н	1
15.	Видалення запитання	Викладач	Н	2
16.	Редагування запитання	Викладач	В	3
17.	Фільтр результатів тесту	Викладач	С	2
18.	Формування статистики для тестів	Викладач	В	2
19.	Перегляд результатів тесту	Викладач	С	1
20.	Видалення студента	Викладач	Н	2
21.	Призначення балу за питання	Викладач	Н	1
22.	Перегляд результату тесту викладачем	Викладач	В	2

Кожен варіант використання має визначений сценарій. Сценарії окремих варіантів представлено в табл. 2.2-2.7.

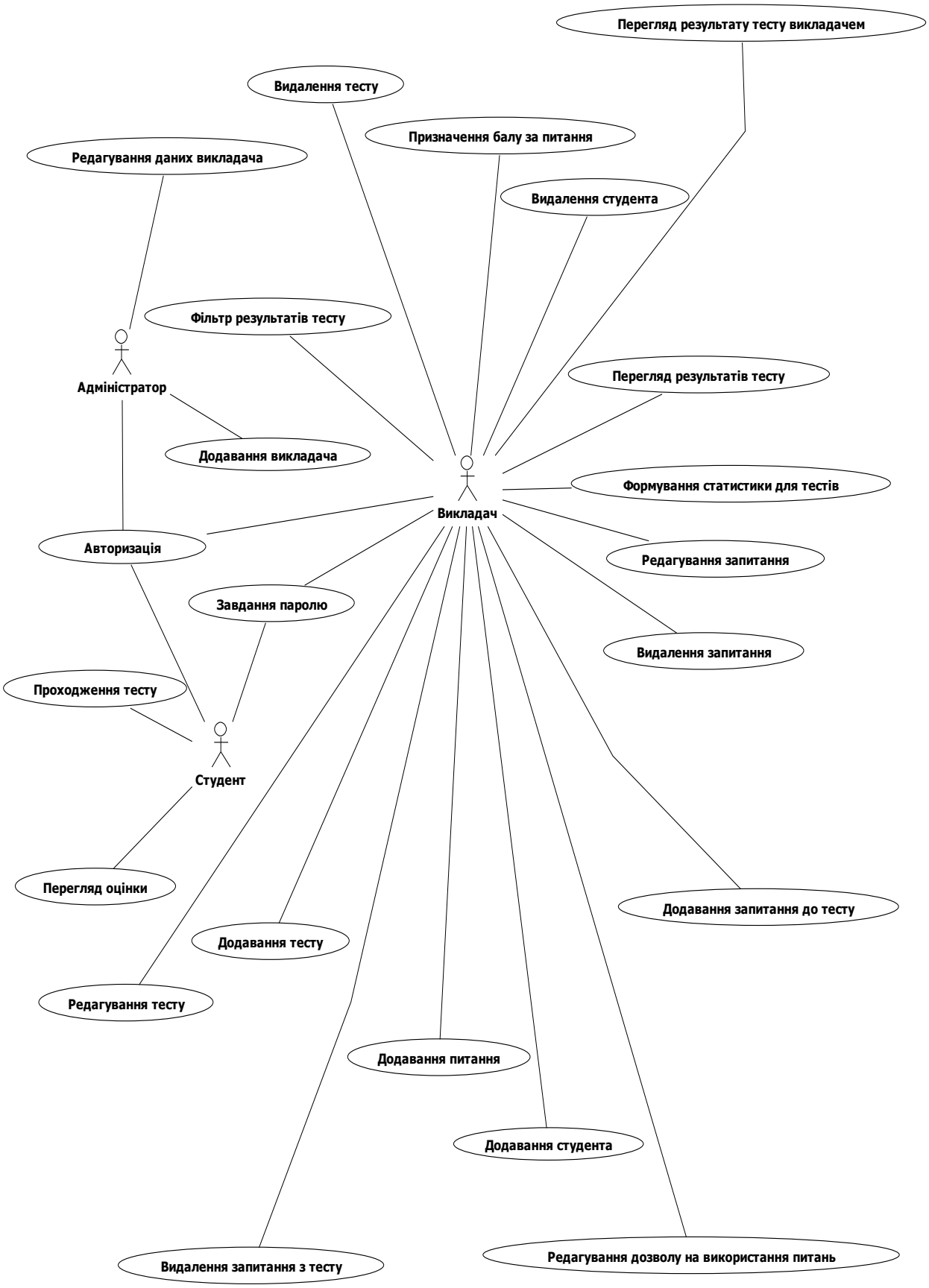

Рисунок 2.1 – Діаграма варіантів використання

Таблиця 2.2 – Варіант використання Ид1

ВВ	Додавання викладача
Актори	Адміністратор
Ціль	Мати актуальний список викладачів
Передумова	Користувач авторизований та переглядає список викладачів
Основний сценарій	
1	Користувач хоче додати нового викладача
2	Система виводить форму для внесення даних про викладача (прізвище, логін, пароль)
3	Користувач вносить дані та хоче зберегти їх
4	Система перевіряє, що необхідні дані внесені, такого логіну немає та зберігає дані
5	Система відображає список викладачів, доданий викладач виділений кольором
Альтернативний сценарій 1	
3а	Користувач не хоче вносити дані та відміняє дію
4а	Система повертається до списку викладачів
Альтернативний сценарій 2	
4б	Користувач не вніс всі необхідні дані
5б	Система виводить повідомлення про необхідність внести дані та повертається до форми внесення даних
Альтернативний сценарій 3	
4в	Користувач вказав логін, який вже присутній в програмі
5в	Система виводить повідомлення про необхідність змінити логін та повертається до форми внесення даних
Результат	Створено обліковий запис для нового викладача

Таблиця 2.3 – Варіант використання Ид2

ВВ	Авторизація
Актори	Адміністратор, Викладач, Студент
Ціль	Надати користувачу можливість виконувати свої функції
Передумова	Користувач внесений до програми
Основний сценарій	
1	Користувач хоче почати працювати з програмою
2	Система виводить форму для внесення даних логіну та паролю
3	Користувач вносить дані та хоче ввійти у систему
4	Система перевіряє, чи існує користувач з вказаними логіном та паролем і формує екран в залежності від статусу користувача
Альтернативний сценарій 1	
4а	Користувач з вказаними логіном та паролем не існує
5а	Система виводить повідомлення про невірний логін або пароль та повертається до форми введення ідентифікуючих даних
Результат	Користувач отримує форму з переліком доступних йому функцій

Таблиця 2.4 – Варіант використання Ид3

ВВ	Редагування даних викладача
Актори	Адміністратор
Ціль	Мати коректні дані про викладача
Передумова	Авторизований Адміністратор, виведений список викладачів
Основний сценарій	
1	Користувач хоче змінити дані про заданого викладача
2	Система виводить форму з даними викладача
3	Користувач змінює дані та хоче зберегти їх
4	Система зберігає дані та відображає список викладачів, викладач, для якого виконувалися зміни, виділений кольором

Продовження табл. 2.4

Альтернативний сценарій 1	
4а	Користувач не вказав необхідні дані
5а	Система виводить повідомлення про невірні дані та повертається до форми з даними про викладача
Альтернативний сценарій 1	
1б	Користувач не вказав викладача, для якого хоче змінити дані
2б	Система виводить повідомлення про відсутність вибору викладача
Результат	Користувач має актуальний список викладачів

Таблиця 2.5 – Варіант використання Ид4

ВВ	Зміна паролю
Актори	Адміністратор, Викладач, Студент
Ціль	Користувач має пароль, який йому зручно використовувати
Передумова	Користувач авторизований
Основний сценарій	
1	Користувач хоче змінити свій пароль
2	Система виводить форму для введення нового паролю та його підтвердження
3	Користувач вносить дані та хоче зберегти їх
4	Система перевіряє, чи відповідає пароль правилам, чи співпадають дві версії паролю та зберігає новий пароль
Альтернативний сценарій 1	
4а	Версії паролю не співпадають
5а	Система виводить повідомлення про неспівпадіння
Альтернативний сценарій 2	
4б	Пароль не задовольняє правилам
5б	Система виводить повідомлення про невірні дані та повертається до форми з даними про викладача
Результат	Користувач має актуальний список викладачів

Таблиця 2.6 – Варіант використання Ид5

ВВ	Додавання тесту
Актори	Викладач
Ціль	Створити новий тест
Передумова	Користувач авторизований та переглядає список тестів
Основний сценарій	
1	Користувач хоче додати новий тест
2	Система виводить форму для внесення даних про тест (назва, тип)
3	Користувач вносить дані та хоче зберегти їх
4	Система перевіряє, що необхідні дані внесені, такого тесту немає та зберігає дані
5	Система відображає список тестів, доданий тест виділений кольором
Альтернативний сценарій 1	
3а	Користувач не хоче вносити дані та відміняє дію
4а	Система повертається до списку тестів
Альтернативний сценарій 2	
4б	Користувач не вніс назву теста
5б	Система виводить повідомлення про необхідність внести дані та повертається до форми внесення даних
Альтернативний сценарій 3	
4в	Користувач вказав назву, яка вже присутня в програмі
5в	Система виводить повідомлення про необхідність змінити назву та повертається до форми внесення даних
Результат	Створено новий тест

Таблиця 2.7 – Варіант використання Ид6

ВВ	Проходження тесту
Актори	Студент
Ціль	Мати уявлення про знання студента
Передумова	Користувач авторизований та переглядає список доступних тестів
Основний сценарій	
1	Користувач хоче пройти тест
2	Система створює запис про проходження тесту
3	Система виводить питання тесту відповідно його типу
4	Користувач вносить відповідь
5	Система перевіряє, що необхідні дані внесені та зберігає дані
6	Система повторює пункти 2-4, стільки разів, скільки питань є у тесті
7	Після закінчення питань система пропонує зберегти результати
8	Користувач підтверджує збереження
9	Система формує оцінку та фіксує спробу проходження тесту
Альтернативний сценарій 1	
3а-4а	Користувач хоче припинити виконання проходження тесту
5а	Система перепитує, чи потрібно зберегти результати тестування
6а	Користувач хоче зберегти дані
7а	Система формує оцінку, фіксує спробу та повертається до списку доступних тестів
Альтернативний сценарій 2	
3б-4б	Користувач хоче припинити виконання проходження тесту
5б	Система перепитує, чи потрібно зберегти результати тестування
5б	Користувач не хоче зберегти дані
6б	Система видаляє збережені відповіді, запис про проходження тесту та повертається до списку доступних тестів

Продовження табл. 2.7

Альтернативний сценарій 3	
6в	Тест є контролюючим і час на проходження завершився
5в	Система формує оцінку, фіксує спробу та повертається до списку доступних тестів
Результат	Студент отримав оцінку за проходження тесту

2.3 Нефункціональні вимоги

Нефункціональні вимоги - це специфікація, яка описує можливості роботи системи й обмеження, що поліпшують її функціональність, тобто вимоги, які описують робочі якості, а не поводження продукту [28].

В [28] перелічені основні групи нефункціональних вимог:

– продуктивність і масштабованість - як швидко система повертає результати, наскільки зміниться ця продуктивність при більше високих навантаженнях;

– переносимість і сумісність - на якому устаткуванні, операційних системах, браузерах і їхніх версіях працює програмне забезпечення; чи конфліктує програма з іншими застосунками й процесами в цих середовищах;

– надійність, доступність, ремонтопридатність - як часто в системі трапляються критичні збої; скільки часу в користувачів займають простої;

– безпека - як система і її дані захищені від атак;

– локалізація - чи відповідає система місцевій специфіці;

– юзабіліті - наскільки легко клієнтові користуватися системою.

Продуктивність – система дає відклик на запит користувача протягом не більше 2 сек, при одночасній роботі до 150 користувачів на персональних комп’ютерах в браузері Chrome.

Переносимість і сумісність – система працює за допомогою браузера Chrome, на операційній системі Wihdows 7 та вище.

Надійність, доступність, ремонтопридатність – програма має працювати не менше 98% часу на протязі доби.

Безпека – захист даних забезпечується авторизацією доступу, збереженням паролів у шифрованому вигляді, архівацією даних.

Локалізація – інтерфейс програми має бути створений на українській мові відповідно закону України про освіту, формат дати повинен бути наступним: місяць, дата, рік.

Юзабіліті – елементи керування мають бути однотипними на різних формах, не повинно бути елементів, призначення яких може трактуватися двозначно, кольори форм мають відповідати вимогам контрасту; дизайн елементів керування має відповідати вимогам ієрархії (елементи повинні відрізнятися друг від друга по пріоритетності залежно від виконуваних завдань), дистанціювання (між елементами повинні спостерігатися порожні простори, що поліпшують читаність контента), вирівнювання (елементи повинні бути вирівняні відносно один одного).

2.5 Висновки до розділу

В розділі визначені три типи користувачів, які мають працювати з програмою. Для кожного з них писаний загальний функціонал.

Наданий детальний опис варіантів використання відповідно вимогам до опису сценаріїв.

Також описані характеристики системи у вигляді нефункціональних вимог.

3 ПРЕДСТАВЛЕННЯ ЗАВДАННЯ НА ЗАКРІПЛЕННЯ ЗНАНЬ ПРО АЛГОРИТМИ ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ

3.1 Загальний алгоритм виконання завдання в навчальній програмі ProcessStudy

На рис.3.1 представлено схему алгоритму виконання контрольного завдання, пов’язаного з відтворенням поведінки підсистеми планування операційної системи.

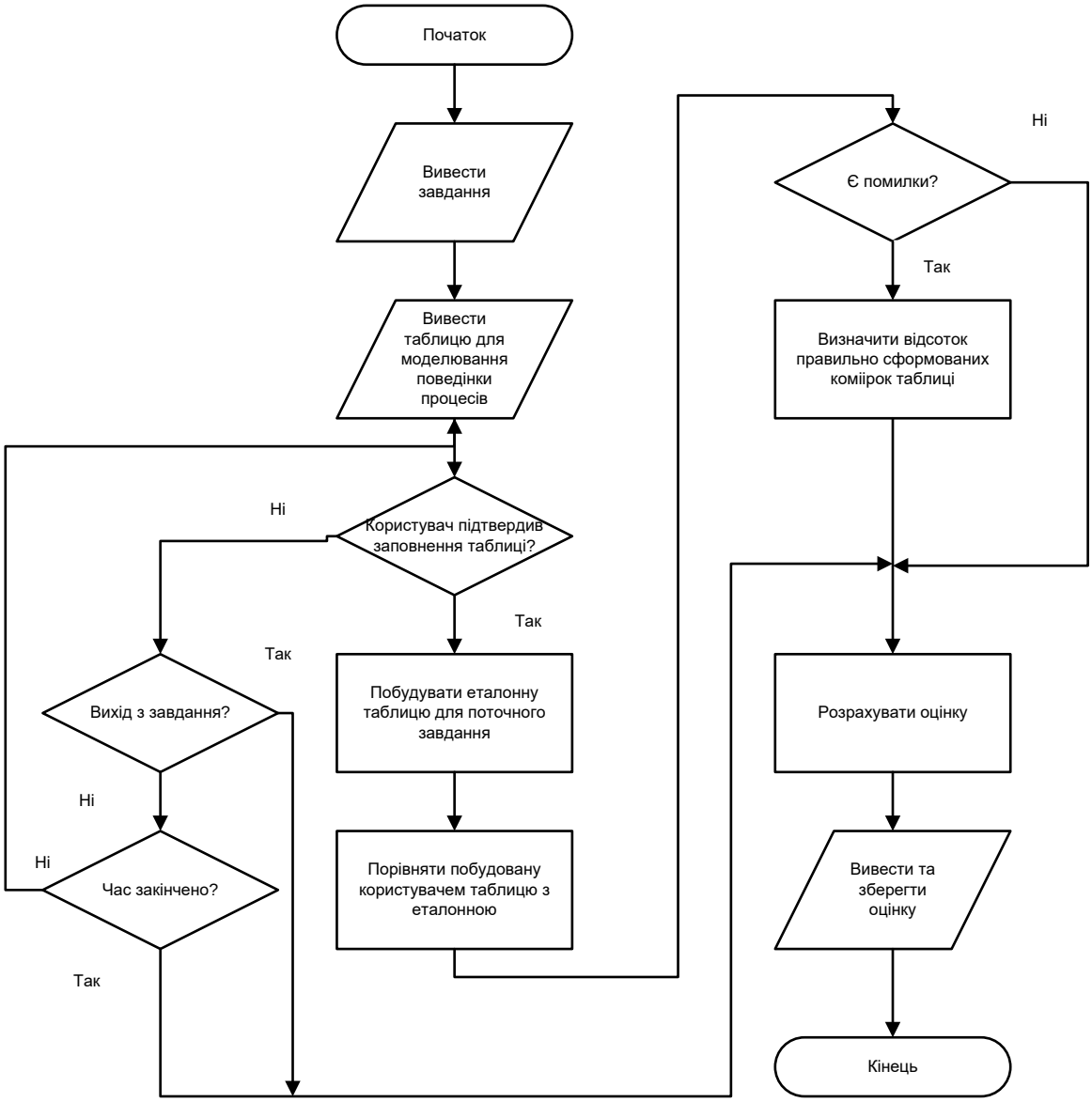

Рисунок 3.1 - Схема алгоритму виконання контрольного завдання

Алгоритм включає наступні блоки.

Блок виведення завдання передбачає вибір задачі з бази даних завдань відповідно до поточного користувача. Завдання передбачає вказівку алгоритму планування. набору процесів з зазначенням часу входу у систему та тривалості виконання.

Блок формування таблиці для відтворення роботи підсистеми планування передбачає створення віконної форми, яка буде містити структуру таблиці з кількістю рядків та стовпці, відповідних завданню, але без підказки рішення, тобто кількість стовпців та рядків має бути збільшена о реального вирішення. але достатньою для відображення необхідних даних.

Кожен рядок таблиці відповідає окремому процесу. Кожен стовпець відповідає одиниці часу.

Кожна комірка таблиці повинна надавати можливість вказати стан відповідного процесу у відповідний момент часу (рис.3.2).

Час / Процес		Моменти часу						
		0	1	…	*i*	...		
Процеси	1							
	2							
	…							
	j				Пустачи Очікуваннячи Виконання			
	...							

Рисунок 3.2 – Таблиця для відображення поведінки процесів

Можливі стани для комірки таблиці:

– пуста, процес ще не ввійшов до системи, або вже вийшов;

– очікування, процес готовий до виконання. але процесор зайнятий іншим процесом;

– виконання, процесор виконує відповідний процес у відповідний момент часу.

Після обрання стану комірка має отримати колір відповідно обраному стану.

Блок побудови еталонної таблиці передбачає отримання масиву даних *ME* розмірністю, відповідною роботі процесів для поточного завдання, *RE* × *CE,* де *RE* – кількість рядків у *ME, CE* – кількість стовпців, відповідно. Значення елементу масиву може бути 1, 2 чи 3, відповідно стану.

Коли користувач підтверджує завершення виконання завдання або скінчився час, відведений на виконання, система обробляє дані, введені користувачем, перетворюючи заповнену інтерактивну таблицю на масив цілих чисел *MU*, розмірність визначається за останніми заповненими рядками/стовпцями (рядок/стовпець містить хоча б один стан Очікування або Виконання), *RU* × *CU.*

Блок порівняння введеної користувачем таблиці з еталонною таблицею передбачає порівняння значення кожного елементу масиву даних *ME*, відповідного еталонній таблиці, з масивом даних, відповідному таблиці користувача *MU*. Обчислюється кількість неспівпадаючих значень *NF* у частині масиву *MU,* яка відповідає за розміром розміру масиву *ME*. Також визначається кількість надлишкових рядків *RF* та стовпців *CF* у *MU* (рис.3.3).

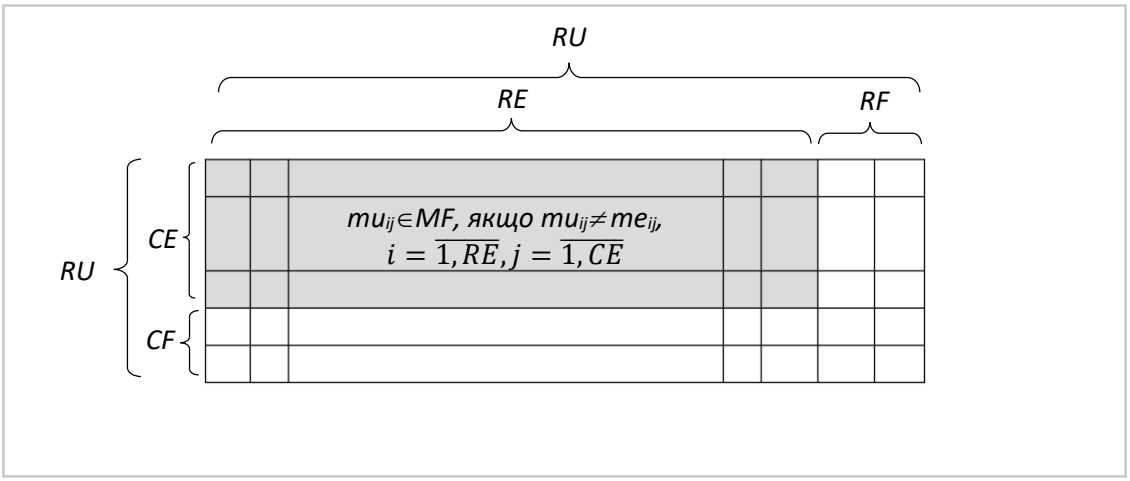

Рисунок 3.3 – Схема визначення оцінки за виконання завдання

На основі отриманих значень обчислюється оцінка

$$p = \frac{RE \cdot CE - NF - RF - CF}{RE \cdot CE} \cdot pmax,$$

де *pmax* – максимально можлива оцінка за виконання завдання.

3.2 Формалізоване представлення завдання для закріплення знань про планування роботи процесів

Кожне завдання складається з двох основних компонентів:

– алгоритм *Al*;

– множина процесів *PR*.

Алгоритм визначає подальшу поведінку процесів в системі відповідно роботі підсистеми планування обрання процесу для виділення процесору.

Елементи множини *PR* в свою чергу також включають кілька компонентів.

Кожен елемент *pr* має початкові дані t_{st}, *pl* та *pt*, де t_{st} відповідає часу надходження процесу у систему, *pl* - тривалості виконання процесу в тактах процесору, *pt* - пріоритету процесу.

В процесі обробки завдання, в залежності від алгоритму планування, для кожного процесу визначається послідовність часових відрізків для всього

вектору роботи системи планування (початком вважається час надходження в систему першого процесу з множини *PR*, кінцем – час закінчення роботи останнього процесу з множини *PR*. Тривалість усього періоду роботи системи, відповідного завданню, є *TT*.

Відрізки часу для кожного елементу *pr* можна описати у вигляді списку *TPR*. Кількість елементів у списку дорівнює *NTTR*. Елементи *TPR*, в свою чергу, мають такі складові: $t0_i$, stp_i, $tend_i$, i=1.. *NTTR*, де $t0_i$ – час початку відрізку часу, stp_i – стан процесу протягом відрізку часу, $tend_i$ – час закінчення відрізку часу

Можна вважати елементи упорядкованими за $t0$. Для першого елемент у списку $t0_1=0$, для останнього елементу $tend_{NTTR}=TT$.

Кількість елементів у списку *NTTR* може бути будь-якою, але не більше *TT*.

Очевидно, що при однакових *PR*, але для різних *Al* множина *TPR* може бути різною для процесів. що входять до *PR*.

3.3 Висновки до розділу

В розділі описано, яким чином відбувається обробка специфічного завдання для контролю знань студентом поведінки системи планування. Алгоритм визначає загальні кроки роботи програми для формування віконної форми, завдяки якої можливо сформувати відповідь студента.

Також описано формальне представлення завдання в цілому з точки зору поведінки процесів, включених до завдання відповідно зазначеному алгоритму планування.

4 ПРОЕКТУВАННЯ ПРОГРАМИ НАВЧАННЯ ВЗАЄМОДІЇ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ

4.1 Структура бази даних

Для предметної області можна виділити наступні сутності.

Сутність Персона містить список викладачів та студентів, має атрибути – Ідентифікатор та ПІБ .

Сутність Користувачі містить список зареєстрованих користувачів з визначеними логіном та паролем, має атрибути – Ідентифікатор, логін, пароль.

Сутність Група містить список груп, у які можуть бути поєднані студенти, має атрибути – Ідентифікатор, назва.

Сутність Тест містить список тестів, створених для студентів, має атрибути – Ідентифікатор, назва, чи є навчаючим або контролюючим.

Сутність Питання містить список питань, які можуть входити до тестів, має атрибути – Ідентифікатор, текст, чи є задачею.

Сутність Тип питання містить перелік типів питань, які можливі у програмі, в залежності від типу формується вікно для введення відповіді, можуть бути такими: питання з кількома варіантами відповіді та з одним правильним варіантом; питання з кількома варіантами відповіді та з кількома правильними варіантами; питання з відкритим текстом відповіді з еталонною відповіддю; питання типу есе, яке вимагає ручної перевірки; питання типу задача- специфічне для дисципліни «Операційні системи». Користувач не може змінювати екземпляри цієї сутності. Має атрибути – Ідентифікатор, назва.

Сутність Алгоритм містить список алгоритмів планування, для яких може бути створена задача. Користувач не може змінювати екземпляри цієї сутності. Має атрибути – Ідентифікатор, опис.

Сутність Задача містить перелік задач для виведення в тесті, має атрибути – Ідентифікатор, кількість квантів.

Сутність Відповідь містить список можливих відповідей для питань, має атрибути – Ідентифікатор, текст, чи є правильним.

Сутність Процес описує складові задачі з планування роботи процесів в операційній системі, має атрибути – Ідентифікатор, час надходження в систему, тривалість, пріоритет.

Сутність Результат містить список балів за відповідь, має атрибути – Ідентифікатор, дата та час виконання, бал за відповідь на питання.

Сутність Користувачі має зв'язок 1:M з сутністю Персона.

Сутність Персона має зв'язок 1:M з сутністю Група.

Сутність Питання має зв'язок 1:M з сутністю Відповідь.

Сутність Задача має зв'язок 1:1 з сутністю Питання.

Сутність Тип питання має зв'язок 1:M з сутністю Питання.

Сутність Алгоритм має зв'язок 1:M з сутністю Задача.

Сутність Задача має зв'язок 1:M з сутністю Процес.

Сутність Задача має зв'язок 1:M з сутністю Процес.

Сутність Тест має зв'язок M:N з сутністю Група. Зв'язок M:N між сутностями Тест та Група створює нову сутність Відкриття тесту, яка має власні атрибути – дата та час відкриття, дата та час завершення, кількість спроб, якщо кількість спроб дорівнює 0, то вважається необмеженою.

Сутність Тест має зв'язок M:N з сутністю Питання. Зв'язок M:N між сутностями Тест та Питання створює нову сутність Склад тесту, яка має власний атрибут – бал за конкретне питання у конкретному тесті.

Сутність Результат має зв'язок 1:M з сутністю Питання.

Сутність Результат має зв'язок 1:M з сутністю Відкриття тесту.

Сутність Результат має зв'язок 1:M з сутністю Персона.

Сутності, їх атрибути та зв’язки між сутностями відображені на рис.4.1 у вигляді концептуальної моделі даних.

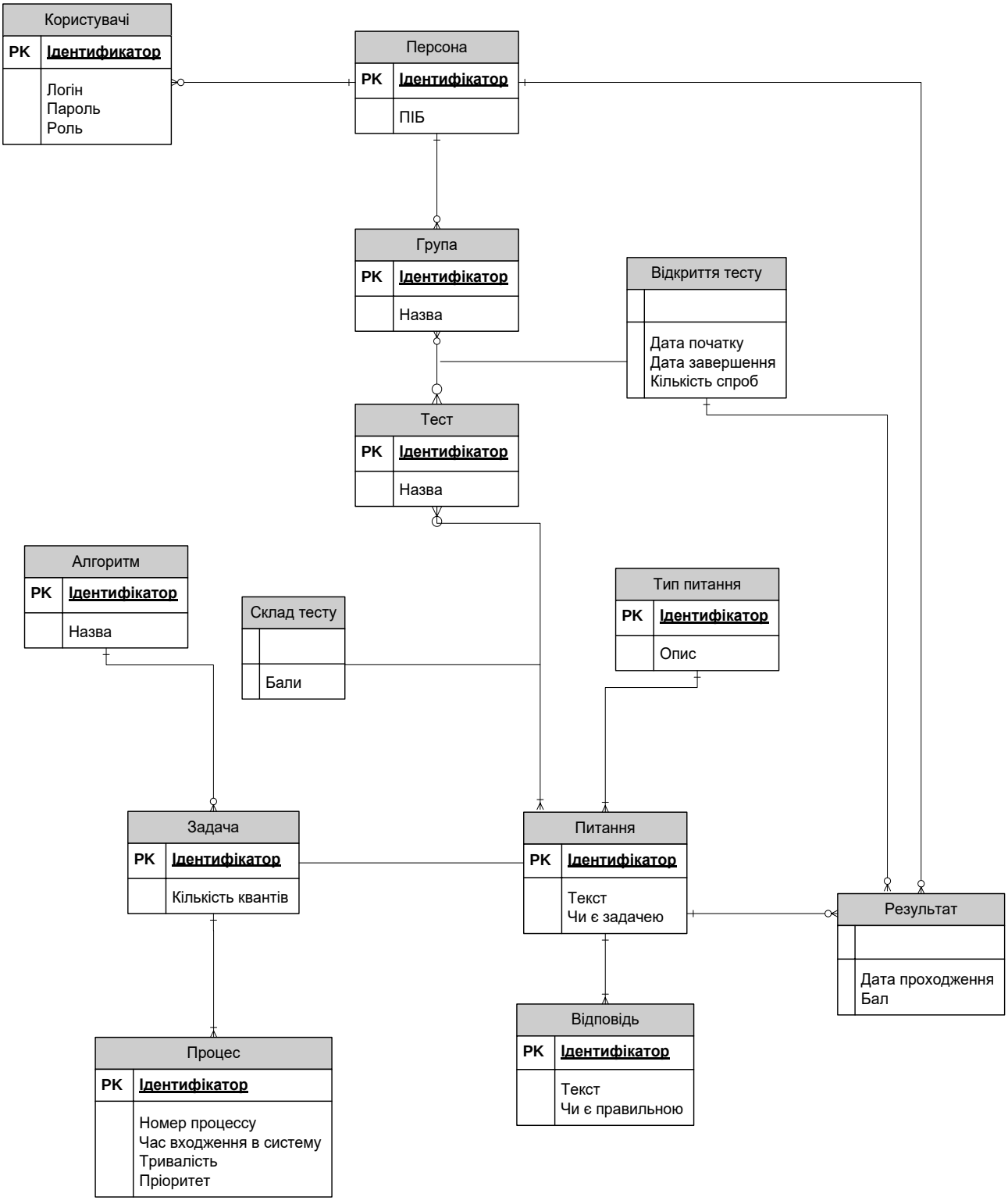

Рисунок 4.1 – Концептуальна модель даних

На основі спроектованої концептуальної моделі даних створена реляційна модель даних. Кожній сутності відповідає реляційна таблиця.

Схема реляційної бази показано на рис.4.2.

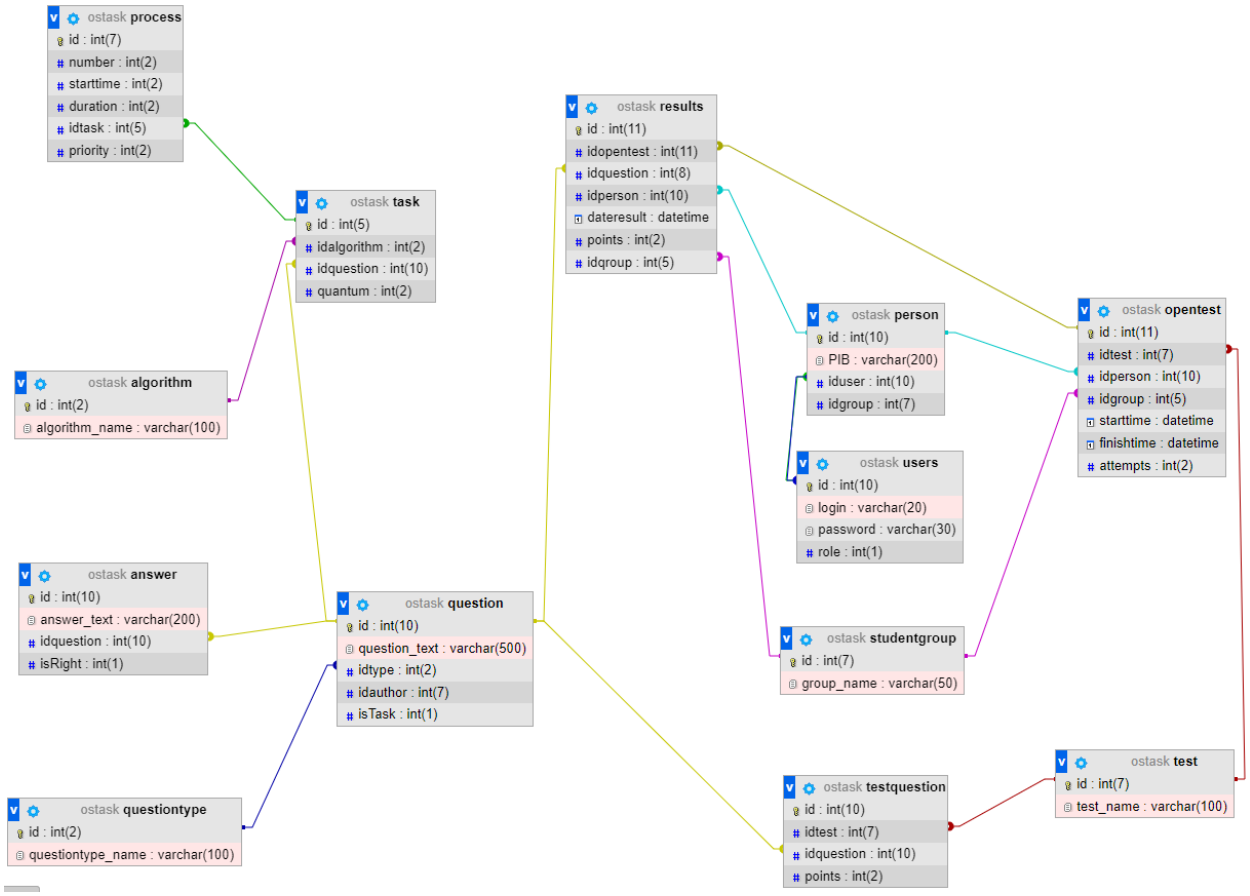

Рисунок 4.2 – Реляційна схема даних

База даних створена за допомогою PHPMyAdmin для СКБД MySQL.

Нижче наведений детальний опис усіх таблиць БД.

У всіх таблиць присутній атрибут id, який відповідає атрибуту Ідентифікатор для сутностей предметної області. Атрибут id зазначений у якості первинного ключа та має властивість автоінкременту, тобто СКБД сама формує значення для цього атрибуту

Таблиця algorithm має структуру, яка показана на рис.4.3.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(2)			Нет	*Нет*		AUTO_INCREMENT
☐	2	algorithm_name	varchar(100)	utf8mb4_unicode_ci		Нет	*Нет*		

Рисунок 4.3 – Структура таблиці algorithm

Атрибут algorithm_name містить назву алгоритму планування.

Таблиця answer має структуру, яка показана на рис.4.4.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(10)			Нет	*Нет*		AUTO_INCREMENT
☐	2	answer_text	varchar(200)	utf8mb4_unicode_ci		Нет	*Нет*		
☐	3	idquestion	int(10)			Нет	*Нет*		
☐	4	isRight	int(1)			Нет	*Нет*		

Рисунок 4.4 – Структура таблиці answer

Атрибут answer_text містить текст відповіді; атрибут isRight визначає, чи є відповідь правильною, 0 – відповідь невірна, 1 – відповідь є вірною; атрибут idquestion служить для зв’язку з таблицею question.

Для таблиці answer визначені наступні обмеження зовнішнього ключа (рис.4.5).

Ограничения внешнего ключа

Действия	Свойства ограничения	Столбец	Ограничение внешнего ключа (INNODB)		
			База данных	Таблица	Столбец
✕	answer_ibfk_1 ON DELETE RESTRICT ON UPDATE RESTRICT	idquestion + Добавить столбец	ostask	question	id

Рисунок 4.5 – Обмеження зовнішнього ключа для таблиці answer

Таблиця opentest має структуру, яка показана на рис.4.6.

Атрибут starttime визначає час відкриття тесту, атрибут finishtime визначає час закриття тесту, атрибут idtest служить для зв’язку з таблицею test, атрибут idgroup служить для зв’язку з таблицею studentgroup, атрибут idperson служить для зв’язку з таблицею person.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(11)			Нет	*Нет*		AUTO_INCREMENT
☐	2	idtest	int(7)			Нет	*Нет*		
☐	3	idperson	int(10)			Нет	*Нет*		
☐	4	idgroup	int(5)			Нет	*Нет*		
☐	5	starttime	datetime			Нет	*Нет*		
☐	6	finishtime	datetime			Нет	*Нет*		
☐	7	attempts	int(2)			Нет	*Нет*		

Рисунок 4.6 – Структура таблиці opentest

Для таблиці opentest визначені наступні обмеження зовнішнього ключа (рис.4.7).

Ограничения внешнего ключа

Действия	Свойства ограничения	Столбец	Ограничение внешнего ключа (INNODB)		
			База данных	Таблица	Столбец
✕	opentest_ibfk_1 ON DELETE RESTRICT ON UPDATE RESTRICT	idgroup + Добавить столбец	ostask	studentgroup	id
✕	opentest_ibfk_2 ON DELETE RESTRICT ON UPDATE RESTRICT	idperson + Добавить столбец	ostask	person	id
✕	opentest_ibfk_3 ON DELETE RESTRICT ON UPDATE RESTRICT	idtest + Добавить столбец	ostask	test	id

Рисунок 4.7 – Обмеження зовнішнього ключа для таблиці opentest

Таблиця person має структуру, яка показана на рис.4.8.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(10)			Нет	*Нет*		AUTO_INCREMENT
☐	2	PIB	varchar(200)	utf8mb4_unicode_ci		Нет	*Нет*		
☐	3	iduser	int(10)			Нет	*Нет*		
☐	4	idgroup	int(7)			Нет	*Нет*		

Рисунок 4.8 – Структура таблиці person

Атрибут PIB визначає прізвище, ім’я та по батькові персони, атрибут iduser служить для зв’язку з таблицею user, атрибут idgroup служить для зв’язку з таблицею studentgroup.

Для таблиці person визначені наступні обмеження зовнішнього ключа (рис.4.9).

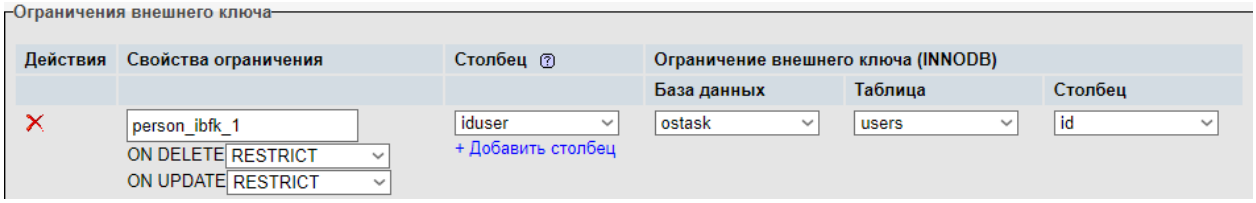

Рисунок 4.9 – Обмеження зовнішнього ключа для таблиці person

Таблиця process має структуру, яка показана на рис.4.10.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(7)			Нет	*Нем*		AUTO_INCREMENT
☐	2	number	int(2)			Нет	*Нем*		
☐	3	starttime	int(2)			Нет	*Нем*		
☐	4	duration	int(2)			Нет	*Нем*		
☐	5	idtask	int(5)			Нет	*Нем*		
☐	6	priority	int(2)			Нет	*Нем*		

Рисунок 4.10 – Структура таблиці process

Атрибут number визначає номер процесу (відповідає ідентифікатору процесу у межах задачі, відповідно. не може повторюватися у межах задачі, є послідовним числом, починаючи з 0), атрибут starttime визначає час надходження процесу у систему, відповідає кванту часу, є числом від 0 до максимальної кількості квантів для задачі, атрибут duration визначає, скільки квантів часу потрібно процесу на виконання; атрибут idtask служить для зв’язку з таблицею task, атрибут priority визначає пріоритет процесу.

Для таблиці process визначені наступні обмеження зовнішнього ключа (рис.4.11).

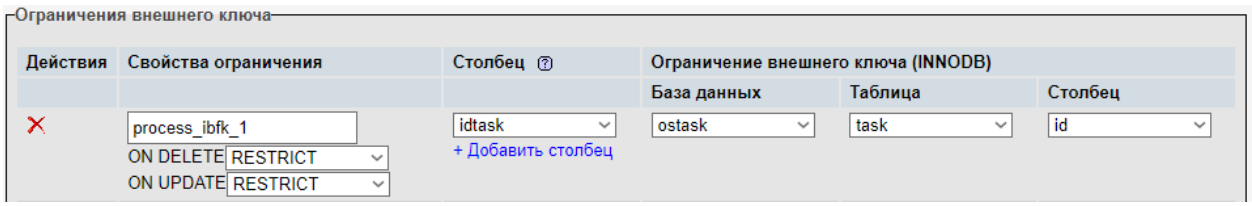

Рисунок 4.11 – Обмеження зовнішнього ключа для таблиці process

Таблиця question має структуру, яка показана на рис.4.12.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(10)			Нет	*Нем*		AUTO_INCREMENT
☐	2	question_text	varchar(500)	utf8mb4_unicode_ci		Нет	*Нем*		
☐	3	idtype	int(2)			Нет	*Нем*		
☐	4	idauthor	int(7)			Нет	*Нем*		
☐	5	isTask	int(1)			Нет	*Нем*		

Рисунок 4.12 – Структура таблиці question

Атрибут question_text містить текст питання, атрибут isTask визначає, чи є питання задачею, 0 – тестове питання. 1 – специфічна задача; атрибут idtype служить для зв’язку з таблицею questiontype, атрибут idauthor служить для зв’язку з таблицею person.

Для таблиці question визначені наступні обмеження зовнішнього ключа (рис.4.13).

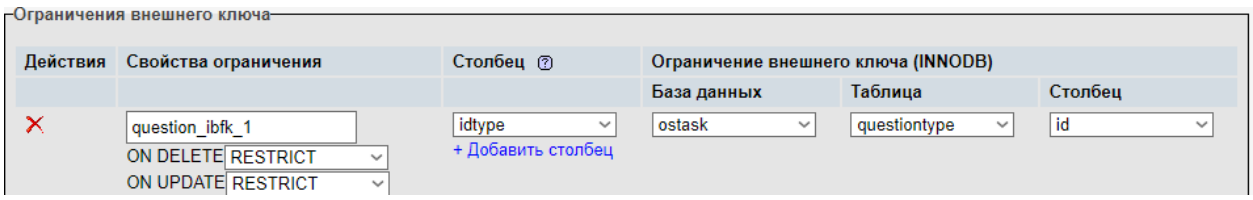

Рисунок 4.13 – Обмеження зовнішнього ключа для таблиці question

Таблиця questiontype має структуру, яка показана на рис.4.14.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(2)			Нет	*Нем*		AUTO_INCREMENT
☐	2	questiontype_name	varchar(100)	utf8mb4_unicode_ci		Нет	*Нем*		

Рисунок 4.14 – Структура таблиці questiontype

Атрибут questiontype_name містить назву типу питання.

Таблиця results має структуру, яка показана на рис.4.15.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(11)			Нет	*Нем*		AUTO_INCREMENT
☐	2	idopentest	int(11)			Нет	*Нем*		
☐	3	idquestion	int(8)			Нет	*Нем*		
☐	4	idperson	int(10)			Нет	*Нем*		
☐	5	dateresult	datetime			Нет	*Нем*		
☐	6	points	int(2)			Нет	*Нем*		
☐	7	idqroup	int(5)			Нет	*Нем*		

Рисунок 4.15 – Структура таблиці results

Атрибут dateresult містить час виконання тесту студентом, атрибут points визначає отриманий бал за питання; атрибут idopentest служить для зв’язку з таблицею opentest, атрибут idperson служить для зв’язку з таблицею person, атрибут idquestion служить для зв’язку з таблицею question.

Для таблиці results визначені наступні обмеження зовнішнього ключа (рис.4.16).

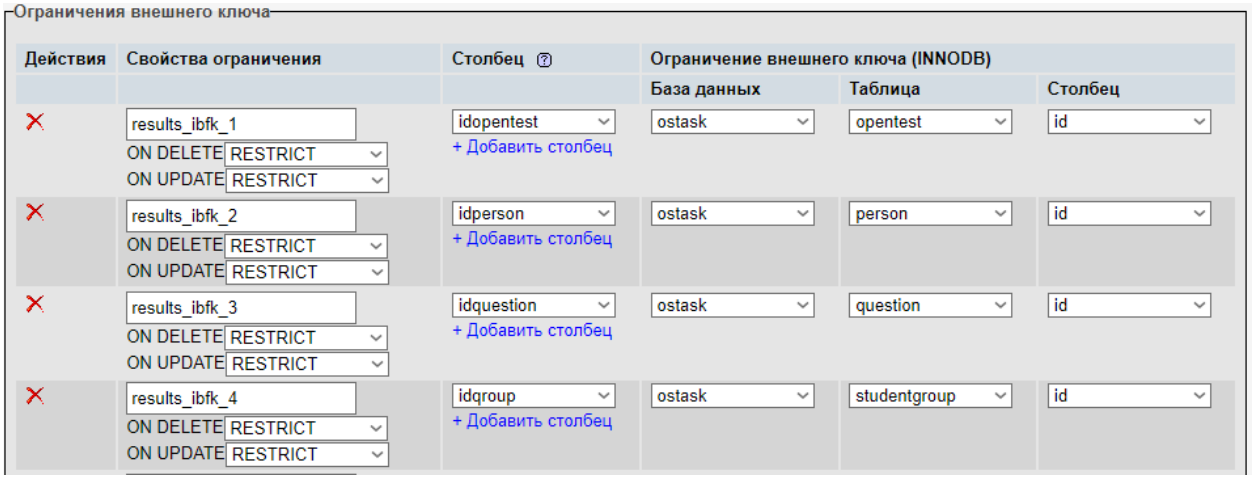

Рисунок 4.16 – Обмеження зовнішнього ключа для таблиці results

Таблиця studentgroup має структуру, яка показана на рис.4.17.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(7)			Нет	*Нем*		AUTO_INCREMENT
☐	2	group_name	varchar(50)	utf8mb4_unicode_ci		Нет	*Нем*		

Рисунок 4.17 – Структура таблиці studentgroup

Атрибут group_name містить назву групи.

Таблиця task має структуру, яка показана на рис.4.18.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(5)			Нет	*Нем*		AUTO_INCREMENT
☐	2	idalgorithm	int(2)			Нет	*Нем*		
☐	3	idquestion	int(10)			Нет	*Нем*		
☐	4	quantum	int(2)			Нет	*Нем*		

Рисунок 4.18 – Структура таблиці task

Атрибут quantum містить кількість квантів, вимагається для окремих алгоритмів планування; атрибут idalgorithm служить для зв’язку з таблицею algorithm, атрибут idquestion служить для зв’язку з таблицею question.

Для таблиці task визначені наступні обмеження зовнішнього ключа (рис.4.19).

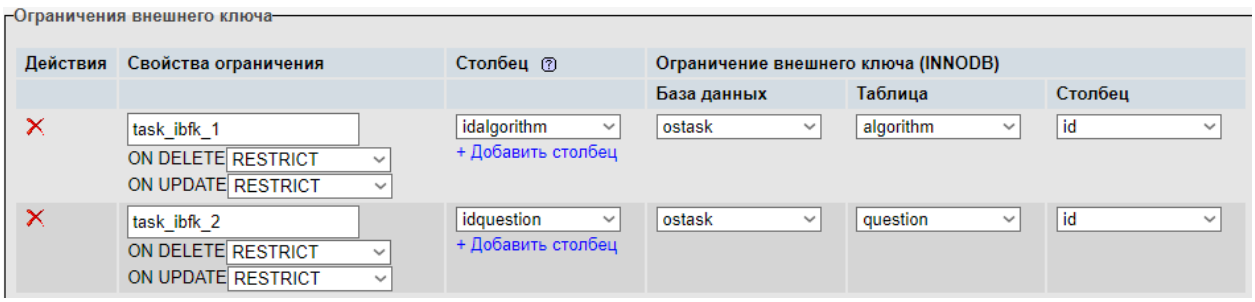

Рисунок 4.19 – Обмеження зовнішнього ключа для таблиці task

Таблиця test має структуру, яка показана на рис.4.20.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(7)			Нет	*Нем*		AUTO_INCREMENT
☐	2	test_name	varchar(100)	utf8mb4_unicode_ci		Нет	*Нем*		
☐	3	isControl	int(1)			Нет	*Нем*		

Рисунок 4.20 – Структура таблиці test

Атрибут test_name містить назву теста, атрибут isControl визначає, чи є тест контролюючим, 0 – навчальний тест, 1 – контролюючий тест.

Таблиця testquestion має структуру, яка показана на рис.4.21.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно
☐	1	id	int(10)			Нет	*Нем*		AUTO_INCREMENT
☐	2	idtest	int(7)			Нет	*Нем*		
☐	3	idquestion	int(10)			Нет	*Нем*		
☐	4	points	int(2)			Нет	*Нем*		

Рисунок 4.21 – Структура таблиці testquestion

Атрибут points визначає макимальний бал за питання; атрибут idtest служить для зв’язку з таблицею test, атрибут idquestion служить для зв’язку з таблицею question.

Для таблиці testquestion визначені наступні обмеження зовнішнього ключа (рис.4.22).

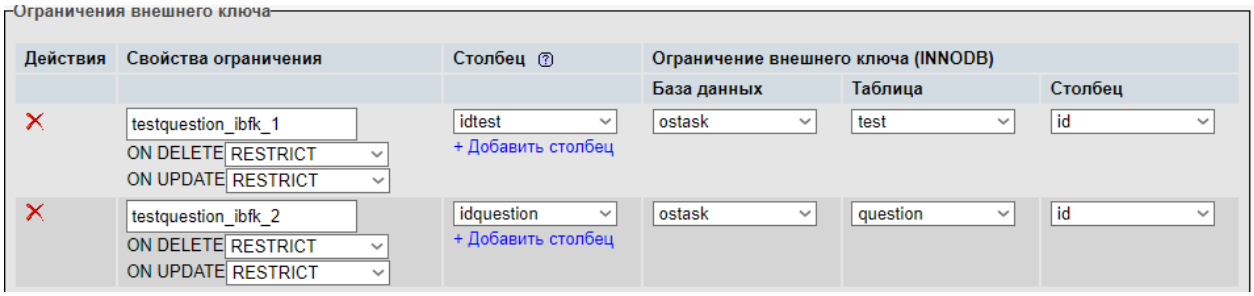

Рисунок 4.22 – Обмеження зовнішнього ключа для таблиці testquestion

Таблиця users має структуру, яка показана на рис.4.23.

	#	Имя	Тип	Сравнение	Атрибуты	Null	По умолчанию	Комментарии	Дополнительно	Действие
☐	1	id	int(10)			Нет	*Нет*		AUTO_INCREMENT	
☐	2	login	varchar(20)	utf8mb4_unicode_ci		Нет	*Нет*			
☐	3	password	varchar(30)	utf8mb4_unicode_ci		Нет	*Нет*			
☐	4	role	int(1)			Нет	*Нет*			

Рисунок 4.23 – Структура таблиці users

Атрибут login містить логін користувача, атрибут password визначає пароль користувача для входження в програму, атрибут role визначає права доступу користувача, 1 – викладач, 2 - студент.

Для таблиці users визначені наступні обмеження зовнішнього ключа (рис.4.24).

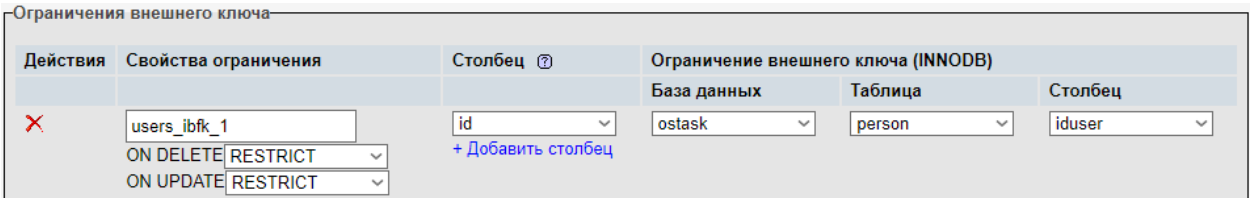

Рисунок 4.24 – Обмеження зовнішнього ключа для таблиці users

4.2 Архітектура програмного продукту

Архітектура застосунку включає [31]:

– вибір структурних елементів і їх інтерфейсів, за допомогою яких складена система, а також їхнього поводження в рамках співробітництва структурних елементів;

– з'єднання обраних елементів структури й поводження в більші системи;

– архітектурний стиль, що направляє всю організацію - всі елементи, їхні інтерфейси, їхнє співробітництво і їхнє з'єднання.

Для застосунку обраний шаблон MVC.

Архітектура MVC розділяє код застосунку на 3 частині: Модель (Model), Вид або Представлення (View) та Контролер (Controller). Поділ на частини дозволяє спростити великий за обсягом код. MVC не прив'язана до конкретної мови програмування, не вимагає обов'язкового використання об’єктно-орієнтованого програмування. На практиці модель часто займає основний обсяг застосунку, і представлена у вигляді великої кількості різнотипних класів - сутностей, сервісів, класів роботи із БД [30].

Модель містить у собі всю логіку застосунку, вона зберігає й обробляє дані, при цьому не взаємодіючи з користувачем прямо (звернутися до Моделі можна з коду, викликаючи її функції).

Представлення відображає дані, які йому передали. У веб- застосунку воно зазвичай складається з HTML-шаблонів сторінок або коду, що відповідає

за відображення інформації на екрані, відображення кнопок і інших елементів інтерфейсу.

Контролер відповідає за виконання запитів, що прийшли від користувача. Контролер відповідає за обробку натискань на кнопки й інші впливи від користувача.

Один Контролер може працювати з декількома Моделями, і навпаки, одна Модель може використатися в декількох Контролерах.

В застосунку навчаючої програми з дисципліни ОС виконаний наступний розподіл за частинами MVC (рис. 4.25).

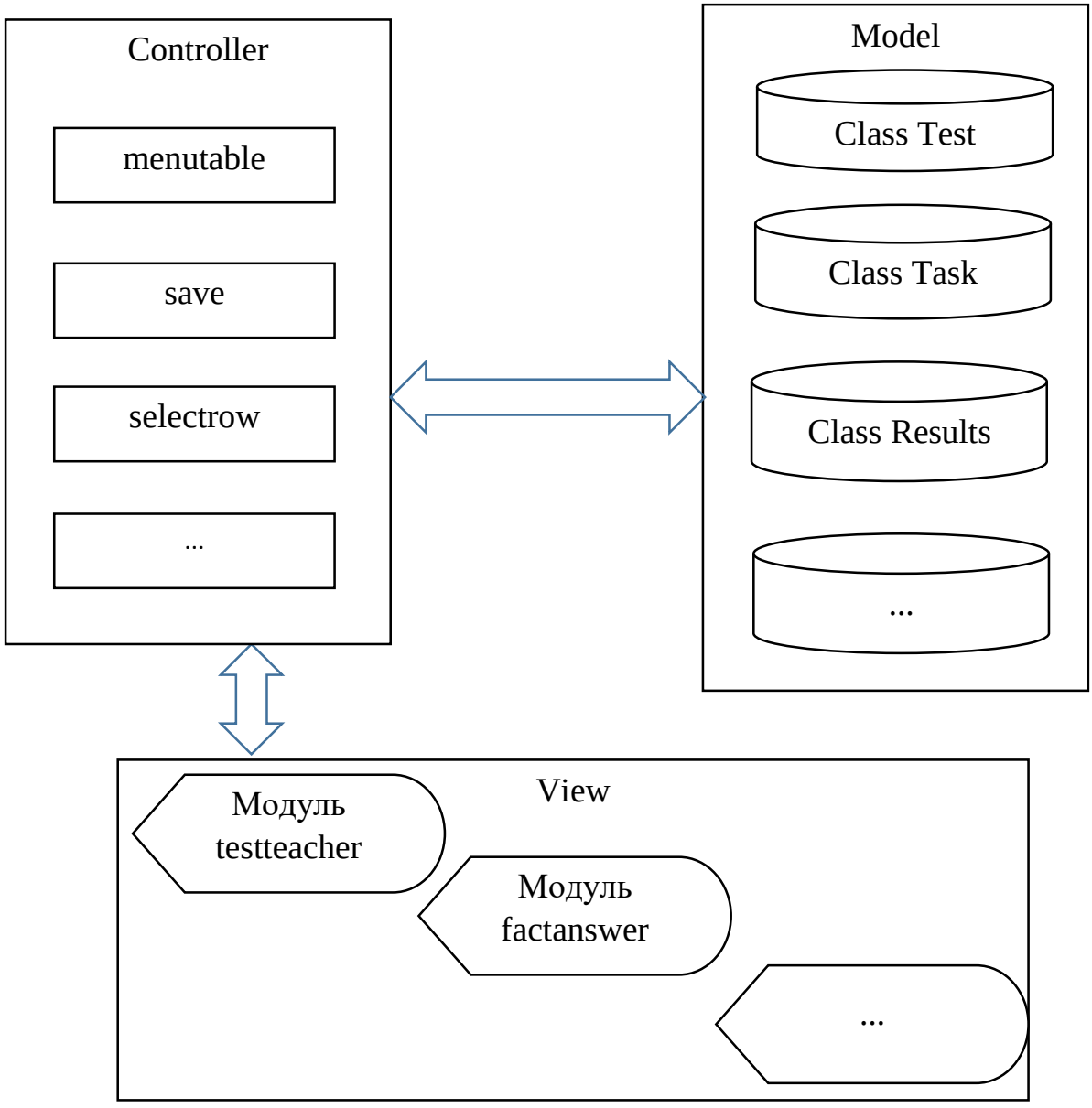

Рисунок 4.25 – Архітектура застосунку

В розділі Model містяться класи для роботи з БД. Є клас Total для виконання з’єднання з базою даних. Інші класи містять методи для виконання збереження даних у відповідну сутність предметної області, оновлення даних та видалення даних. Виконання дій забезпечується формуванням запиту на мові SQL в залежності від переданих значень. Також до складу методів класів входять методи для вибірки інформації, відповідної діям користувача при виконанні функцій програми.

В розділ View включені модулі для відображення списків, відповідних функціям програми. Під час виклику функції, наприклад, за рахунок натискання пункту меню або кнопки, викликаються потрібні методи для отримання інформації з бази даних, та отримана інформація відображається на екрані.

Найбільш загальний алгоритм відображення даних є наступним:

1. Користувач натискає пункт меню для відображення потрібної частини даних (наприклад, список тестів).
2. Програма викликає метод getList отримання основного списку даних класу, пов’язаного з обраною частиною даних.
3. Програма викликає метод класу getFirstInList для отримання ідентифікатору екземпляру сутності, першого у списку, який сформований методом getList.
4. Програма формує приховане поле selectedid для зберігання ідентифікатору поточного екземпляру.
5. Програма формує кнопку для додавання нового екземпляру сутності.
6. Програма формує кнопку для редагування даних.
7. Програма формує кнопку для видалення даних.
8. Програма формує список екземплярів основного списку у вигляді таблиці.
9. Для кожного рядка таблиці задається обробник подвійного кліку по рядку. Обробник передбачає зміну значення поля selectedid та кольору виділеного рядка.

10. Якщо є дані, пов’язані з поточним екземпляром основного списку, то формується блок mainsub для відображення підлеглих записів, пов’язаних з поточним екземпляром основного списку.

11. Програма викликає метод getList отримання підлеглого списку даних класу, пов’язаного з підлеглою частиною обраних даних.

12. Програма викликає метод класу getFirstInList для отримання ідентифікатору екземпляру сутності, першого у підлеглому списку.

13. Програма формує приховане поле selectedidsub для зберігання ідентифікатору поточного екземпляру підлеглого списку.

14. Програма формує кнопки для додавання нового екземпляру підлеглої сутності, редагування та видалення існуючого екземпляру, пов’язаного зі значенням у полі selectedidsub.

15. При натисканні на кнопку додавання нового екземпляру сутності формується множина елементів керування (поле, список або прапорець) для внесення необхідних даних без значень або зі значеннями за замовчуванням. Користувач може внести дані та натиснути кнопку Зберегти або відмовитися від дії за допомогою кнопки Скасування.

16. При натисканні на кнопку редагування існуючого екземпляру сутності формується множина елементів керування (поле, список або прапорець) для внесення необхідних даних, які відображають значення для екземпляру, чий ідентифікатор міститься у полі selectedid (selectedidsub). Користувач може змінити дані та натиснути кнопку Зберегти або відмовитися від дії за допомогою кнопки Скасування.

17. При натисканні на кнопку видалення видаляється екземпляр, чий ідентифікатор міститься у полі selectedid (selectedidsub).

В розділ Controller включені модулі, які викликають відповідні методи класів, наприклад, модуль контролеру savesub викликає методи для додавання даних або збереження змінених даних класів, які відповідають даним поточного пункту меню.

На рис. 4.26 відображена описана схема інтерфейсу, пов’язаного з описаною архітектурою.

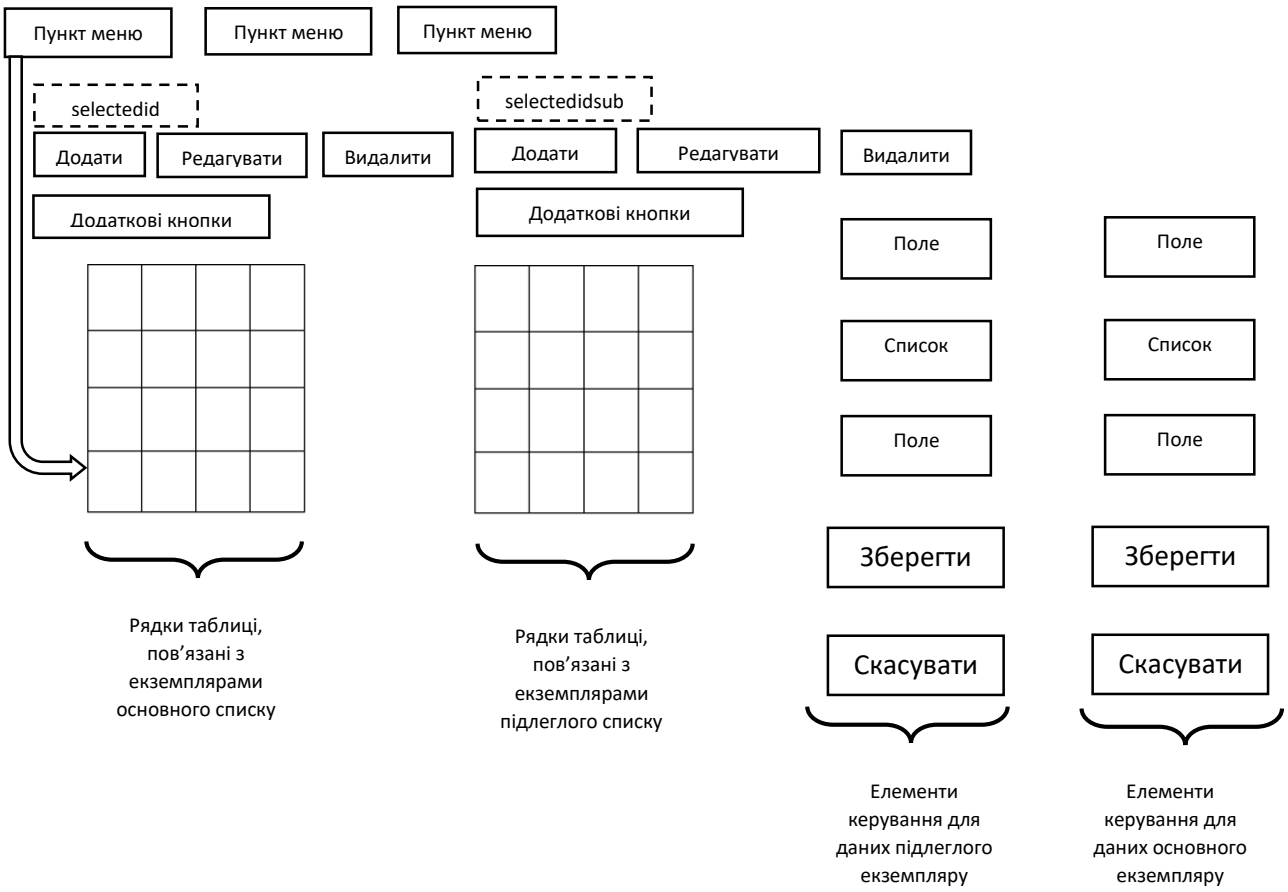

Рисунок 4.26 – Схема інтерфейсу відповідно архітектурі

4.3 Висновки до розділу

Описано предметну область у вигляді концептуальної моделі даних, яка містить множину сутностей, їх атрибутів та взаємозв’язків. На основі отриманої моделі даних спроектовано та реалізовано реляційну базу даних, наданий детальний опис таблиць. Надано обґрунтування вибору шаблону архітектури та описано загальну схему реалізації функцій програми.

5 РЕАЛІЗАЦІЯ ПРОГРАМИ ДЛЯ НАВЧАННЯ ВЗАЄМОДІЇ ПРОЦЕСІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ

5.1 Опис методів класів

Нижче описані методи класів, при проектированні яких враховані можливі зміни в пунктах сценаріїв вариантів використання [32].

Клас **Total** реалізує з’єднання з БД. Містить атрибут db, у якому зберігається змінна для звертання до бази даних.

Метод init задає складові для доступу до бази даних, сервер, назву, користувача.

Метод setconnection встановлює зв'язок з базою даних, встановлює значення атрибуту db.

Клас **User** призначений для роботи з користувачами програми.

Метод getStatus повертає статус користувача за вхідним параметром – ідентифікатор користувача.

Метод getName повертає ім’я користувача за вхідним параметром – ідентифікатор користувача.

Метод getUser повертає ідентифікатор користувача за вхідними параметрами – логін та пароль, у разі невірного логіну або паролю повертає 0.

Клас **Test** призначений для роботи з тестами.

Метод getList повертає список існуючих тестів, відсортований за назвою, вхідних параметрів немає.

Метод getFirstInList повертає ідентифікатор першого у списку тесту, вхідних параметрів немає.

Метод getListOpen повертає список тестів, відкритих для заданого користувача за вхідним параметром – ідентифікатор користувача.

Метод getListAllOpen повертає список груп, для яких був відкритий заданий тест, які були відкриті, за вхідним параметром – ідентифікатор тесту.

Метод saveNew створює новий тест, вхідні параметри – назва тесту, чи є контролюючим, ідентифікатор поточного користувача, повертає ідентифікатор створеного тесту.

Метод deleteRecord видаляє заданий тест, вхідний параметр – ідентифікатор тесту.

Метод saveEdit змінює заданий тест, вхідні параметри – назва тесту, чи є контролюючим, ідентифікатор поточного користувача, ідентифікатор тесту.

Метод getById повертає дані про заданий тест, вхідний параметр – ідентифікатор тесту.

Клас **Question** призначений для роботи з питаннями.

Метод getList повертає список існуючих питань, відсортований за текстом, вхідних параметрів немає.

Метод getFirstInList повертає ідентифікатор першого у списку питання, вхідних параметрів немає.

Метод saveNew створює нове питання, вхідні параметри – текст питання, ідентифікатор типу, ідентифікатор поточного користувача, повертає ідентифікатор створеного питання.

Метод deleteRecord видаляє задане питання, вхідний параметр – ідентифікатор питання.

Метод saveEdit змінює задане питання, вхідні параметри – текст питання, ідентифікатор типу, ідентифікатор поточного користувача, ідентифікатор питання.

Метод getById повертає дані про задане питання, вхідний параметр – ідентифікатор питання.

Метод getListTest повертає список питань для заданого тесту, вхідний параметр – ідентифікатор тесту. Використовується запит

select qts.id,question_text,points,questiontype_name

from Question q join questiontype qt join testquestion qts on qt.id=idtype and qts.idquestion=q.id

where idtest=$id order by 2

Метод getFirstInListTest повертає ідентифікатор першого в списку питання для заданого тесту, вхідний параметр – ідентифікатор тесту. Використовується запит

select qts.id,question_text,points,questiontype_name

from Question q join questiontype qt join testquestion qts on qt.id=idtype and qts.idquestion=q.id

where idtest=$id order by 2 limit 1

Метод getListNoTest повертає список питань, які не ввійшли у заданий тест, вхідний параметр – ідентифікатор тесту.

Метод getListNoTestPlusId повертає список питань, які не ввійшли у заданий тест, плюс ідентифікатор поточного питання у тесті, вхідний параметр – ідентифікатор поточного питання у тесті. Використовується запит

select q.id,question_text,questiontype_name

from (Question q join questiontype qt on qt.id=idtype)

left join (select idquestion,idtest from testquestion where idtest=$idtest)a on a.idquestion=q.id

where a.idtest is null

union select q.id,question_text,questiontype_name

from Question q join questiontype qt

on qt.id=idtype where q.id=$idquestion order by 2";

Клас **Studentgroup** призначений для роботи з групами студентів.

Метод getList повертає список існуючих груп, відсортований за назвою, вхідних параметрів немає.

Метод getFirstInList повертає ідентифікатор першої у списку групи, вхідних параметрів немає.

Метод saveNew створює нову групу, вхідні параметри – назва групи, повертає ідентифікатор створеної групи.

Метод deleteRecord видаляє задану групу, вхідний параметр – ідентифікатор групи.

Метод saveEdit змінює задану групу, вхідні параметри – назва групи, ідентифікатор групи.

Метод getById повертає дані про задану групу, вхідний параметр – ідентифікатор групи.

Клас **Answer** призначений для роботи з відповідями до питань.

Метод getListQuestion повертає список відповідей для заданого питання, вхідний параметр – ідентифікатор питання.

Метод getFirstInListAnswer повертає ідентифікатор першої відповіді у списку відповідей для заданого питання, вхідний параметр – ідентифікатор питання.

Метод getFactAnswer повертає список фактичних оцінок та максимально можливих оцінок для відповідей на питання заданої спроби тесту, вхідний параметр – ідентифікатор спроби. Використовується запит

```
select r.id, question_text,tq.points as maxpoints,points
from question q join results r join testquastion tq
on q.id=r.idquestion and tq.idquestion=r.idquestion
where idattempt=$idattempt
```

Метод getListQuestionProcess повертає список процесів для заданого питання-задачі, вхідний параметр – ідентифікатор питання.

Метод getAlgorithmQuestionProcess повертає назву алгоритму для заданого питання-задачі, вхідний параметр – ідентифікатор питання. Використовується запит

```
select algorithm_name
from algorithm a join task t
on a.id=t.idalgorithm
where idquestion=$idquestion
```

Метод saveNew створює нову відповідь, вхідні параметри – текст відповіді, чи є вірною, ідентифікатор питання, повертає ідентифікатор створеної відповіді.

Метод deleteRecord видаляє задану відповідь, вхідний параметр – ідентифікатор відповіді.

Метод saveEdit змінює задану відповідь, вхідні параметри – текст відповіді, чи є вірною, ідентифікатор питання, ідентифікатор відповіді.

Метод getById повертає дані про задану відповідь, вхідний параметр – ідентифікатор відповіді.

Клас **Statistic** призначений для формування статистики.

Метод getTestStatistic формує статистику у розрізі тестів. Використовується запит

```
select test_name,count(*) as c,avg(totalpoints) as a
from totalpoints
group by idtest,test_name
order by 1
```

Метод getStudentStatistic формує статистику у розрізі студентів. Використовується запит

```
select pib,count(*) as c ,avg(totalpoints) as a
from totalpoints
group by idperson,pib
order by 1
```

Клас **Results** призначений для роботи з результатами тестування.

Метод getList повертає формує загальні результати тестування на основі відповідей на окремі питання. Використовується запит

```
select a.id,test_name,a.idperson,pib,group_name,dateattempt,
sum(points) as a
from test t join opentest ot join attempt a join results r
```

```
join person p join studentgroup g
on t.id=ot.idtest and a.id=r.idattempt
and ot.id=a.idopentest
and a.idperson=p.id and g.id=p.idgroup
group by ot.id,test_name,a.idperson,pib,group_name,dateattempt
order by test_name,dateattempt,pib
```

5.2 Опис інтерфейсу програми

Робота з програмою починається з ідентифікації користувача (рис.5.1).

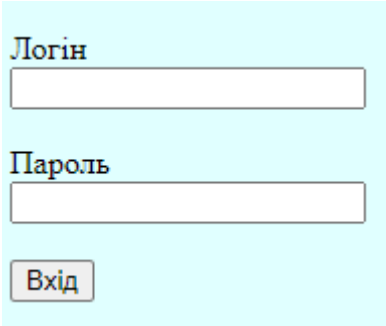

Рисунок 5.1 – Форма ідентифікації користувача

Якщо введені правильні логін та пароль, то відкривається вікно з функціями, відповідними статусу користувача (визначається автоматично). Для користувача. який має статус викладача, відкривається вікно виду, показаного на рис. 5.2.

Форма містить логотип та ім’я поточного користувача в в верхній частині.

Далі знаходиться множина пунктів меню для доступу до відповідної інформації. За замовчуванням першим відкривається інформація про тести. Інформація в основній частині вікна змінюється відповідно обраному пункту меню, обраний пункт меню визначається окремим кольором.

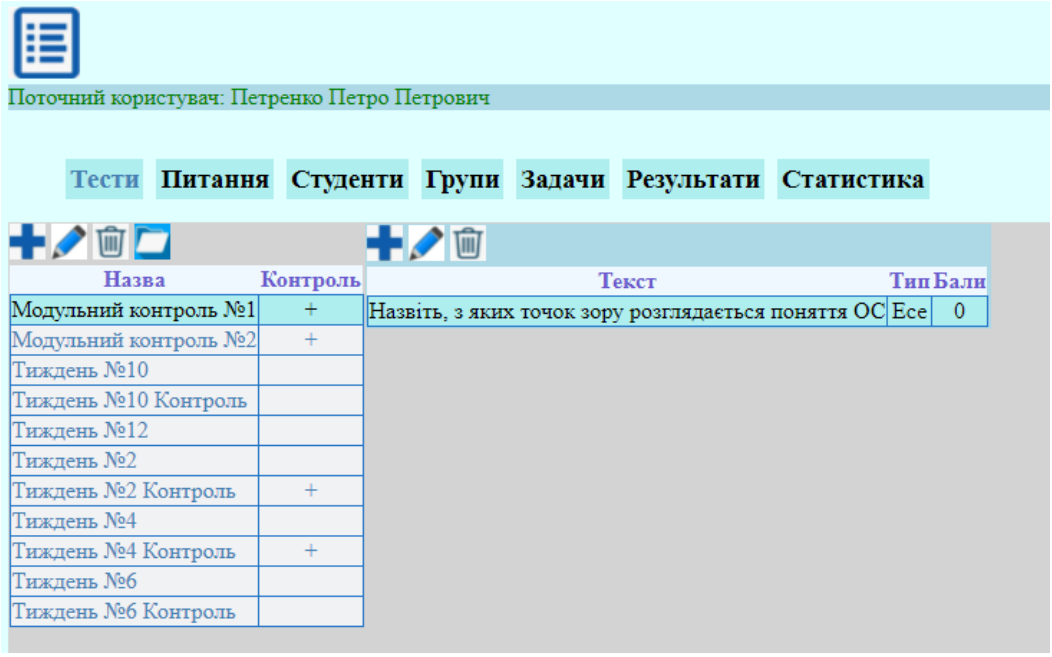

Рисунок 5.2 – Форма для роботи з тестами для викладача

В лівій частині вікна знаходиться список тестів, які створені поточним користувачем. При обранні тесту правіше відображається список питань, які входять до тесту (рис. 5.3), за замовчуванням відображається список питань першого у списку тесту. Обраний тест виділяється іншим кольором.

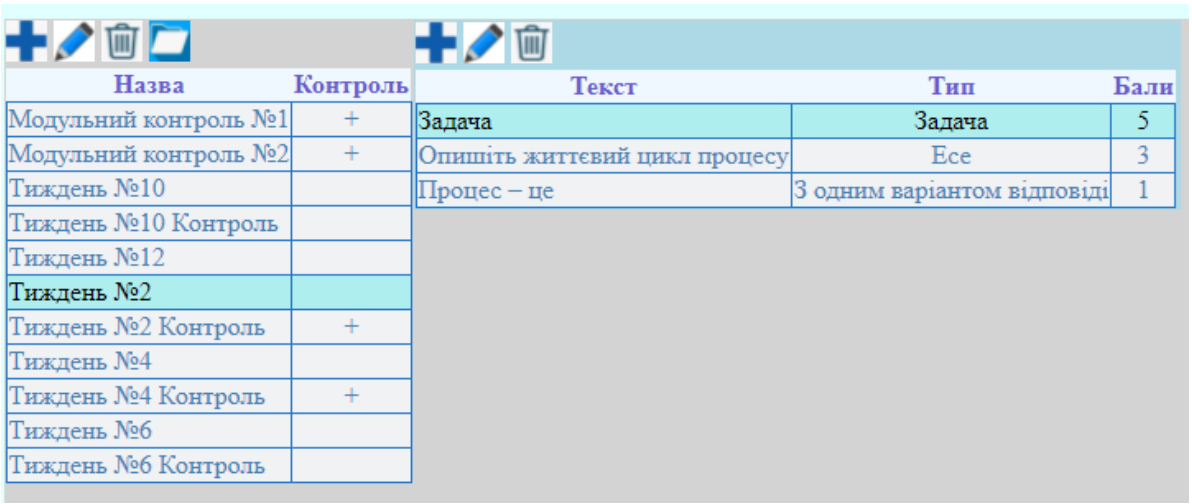

Рисунок 5.3 – Відображення питань обраного тесту

При натисканні на кнопку  над списком тестів в правій частині вікна формуються поля для створення нового тесту (рис.5.4).

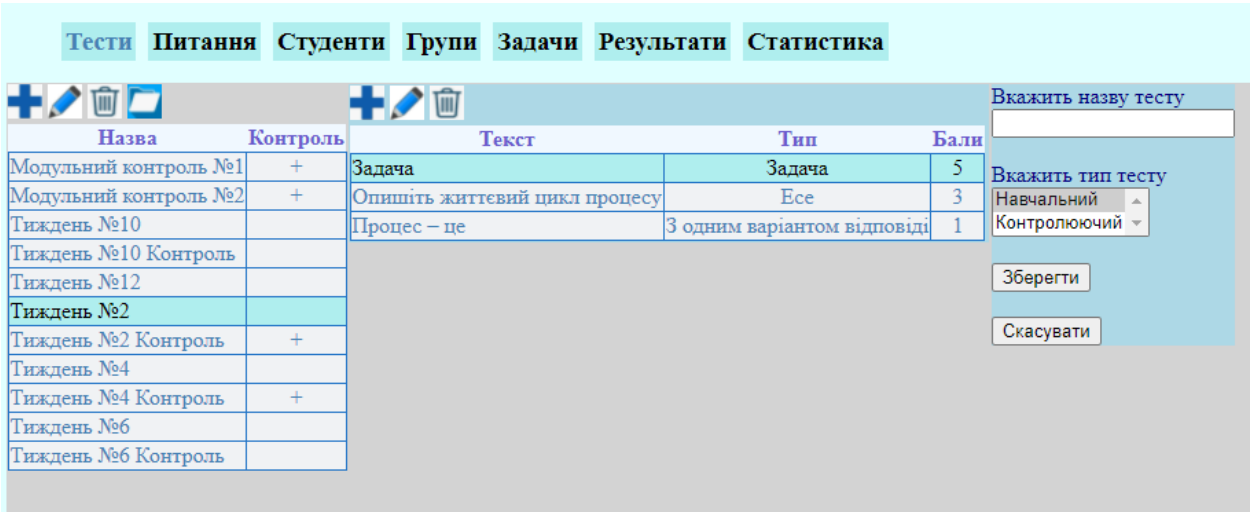

Рисунок 5.4 – Створення нового тесту

У разі відсутності доданих питань до тесту поряд зі списком тестів відображається лише кнопка для додавання нового питання в тест (рис.5.5).

При натисканні на кнопку  в правій частині вікна формуються поля для редагування обраного тесту (виділений кольором) (рис.5.5).

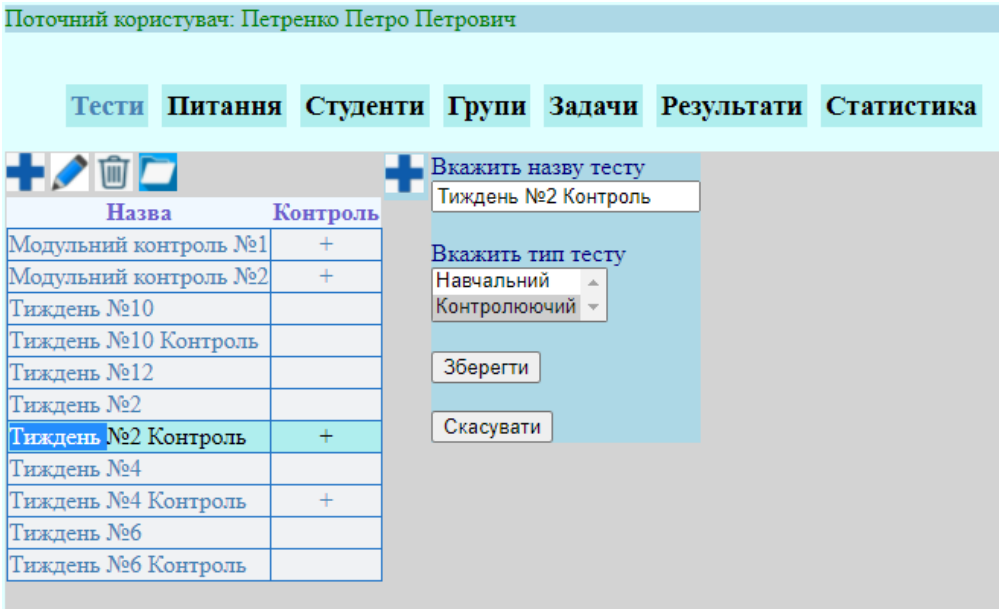

Рисунок 5.5 – Редагування тесту

Видалення тесту виконується при натисканні на кнопку  . Після підтвердження видалення запис видаляється з бази даних.

При натисканні пункту меню Питання основна частина вікна відображає список питань, створених поточним користувачем, та відповіді на питання (рис.5.6).

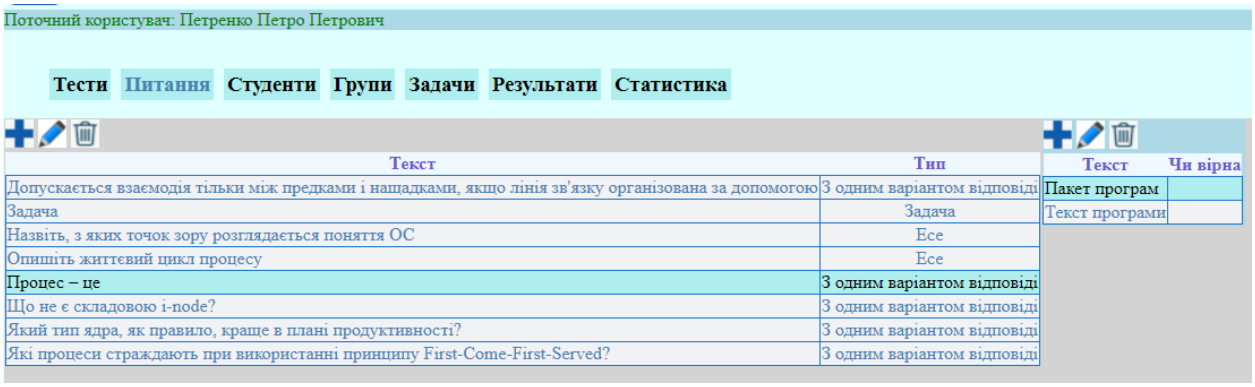

Рисунок 5.6 – Перегляд питань

Для питань, які передбачають варіанти відповідей, в правій частині відображається список доданих варіантів.

Редагування та створення питання виконується аналогічно тесту (рис. 5.7).

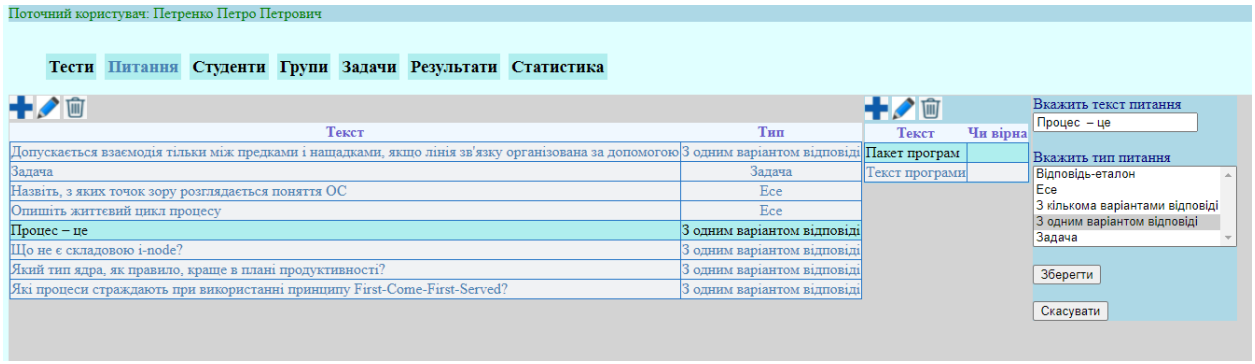

Рисунок 5.7 – Редагування питання

Для питань типу Задача існує окремий пункт меню (рис. 5.8)

Рисунок 5.8 – Перегляд задач

Редагування процесу в задачі виконується аналогічно іншим пунктам меню (рис.5.9).

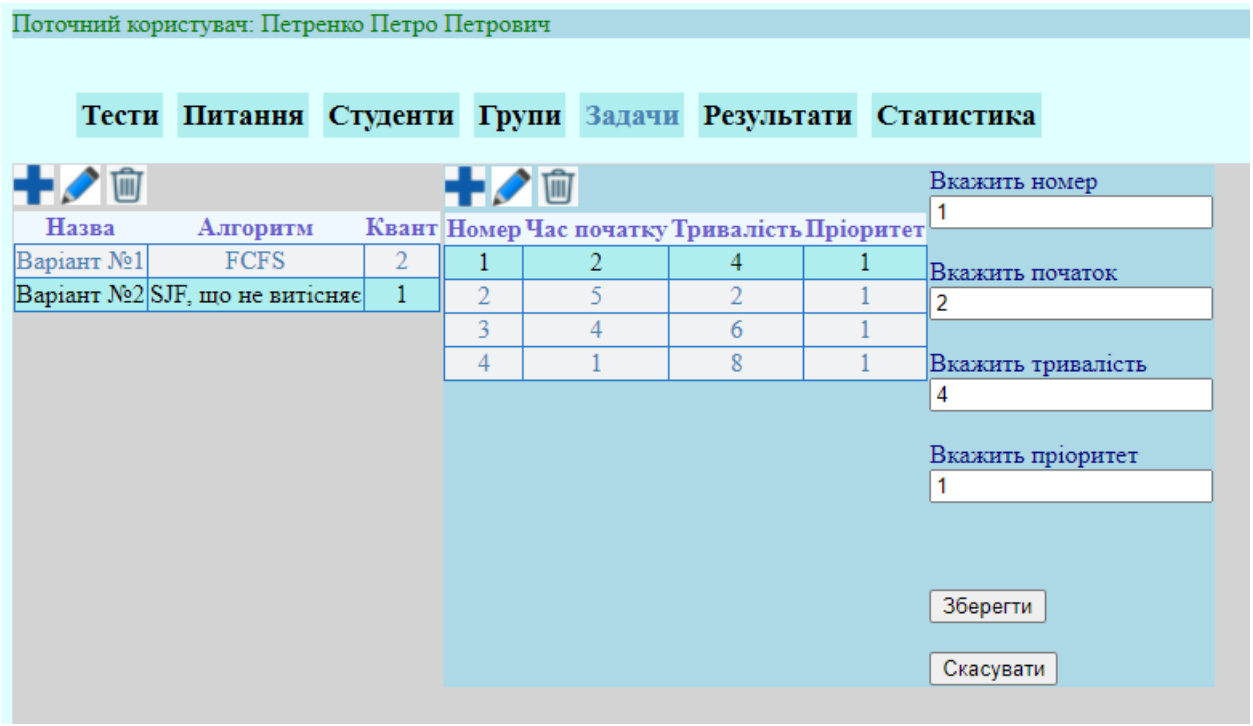

Рисунок 5.9 – Редагування процесу в задачі

При додаванні або редагуванні процесів виконується наступний контроль:

– нумерація процесів виконується з 1;

– нумерація наскрізна, не повинно бути пропущених номерів процесів;

– для безпріоритетних алгоритмів у всіх процесів вказується однаковий пріоритет.

Користувач зі статусом Студент може побачити відкриті для нього тести, раніше або в поточний час.

Для виконаних раніше може переглянути спроби та побачити результати.

Для відкритих в поточний час, якщо використані не всі спроби, користувач може виконати тест.

При виконанні тесту користувачу виводяться питання тесту у довільному порядку.

Якщо питання має тип Задача, то вікно для введення відповіді містить таблицю для формування схеми взаємодії процесів (рис. 5.10).

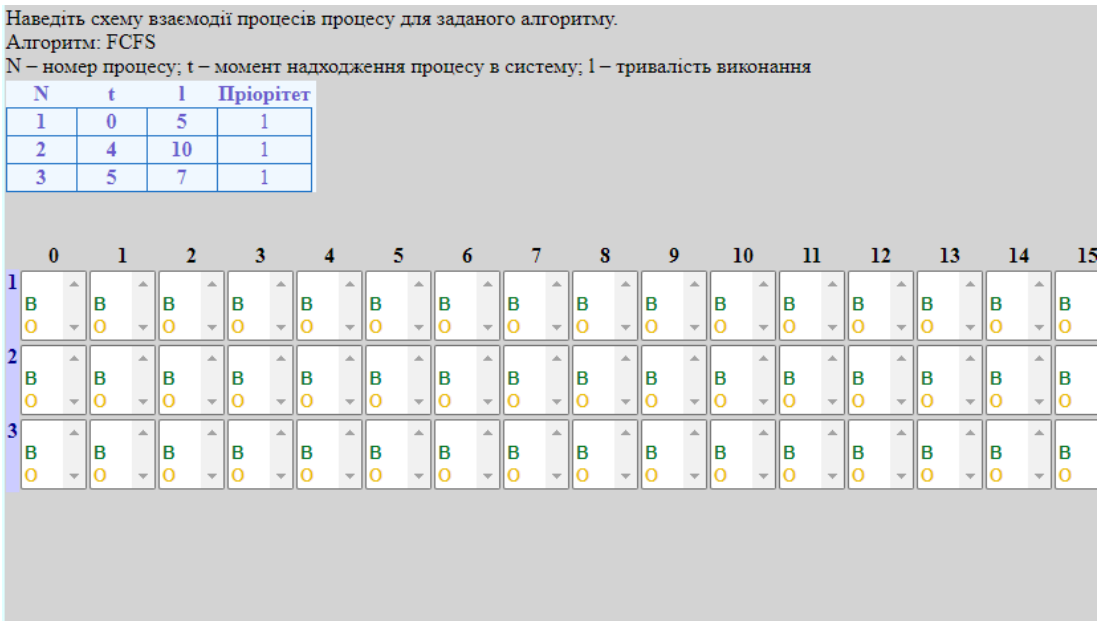

Рисунок 5.10 – Таблиця для введення схеми взаємодії процесів

Таблиця містить стільки рядків, скільки процесів включено в задачу. Стовпці таблиці відповідають моментам часу, починаючи з 0го.

Для кожного моменту часу та для кожного процесу необхідно обрати стан – процес очікує виконання. процес виконується, процес не задіяний (ще не народився. або закінчив роботу).

Щоб змінити стан комірки таблиці, необхідно обрати в списку відповідну літер (В – виконується, О – очікує). При цьому буде змінений колір комірки (рис. 5.11).

Рисунок 5.11 – Зміна станів комірок таблиці

5.3 Висновки до розділу

В розділі описані методи окремих класів для обробки даних, які описують тести та їх складові. Також надані приклади віконних форм програми та описані їх основні компоненти.

6 ТЕСТУВАННЯ ПРОГРАМИ ДЛЯ ОТРИМАННЯ ЗНАНЬ ПРО ПЛАНУВАННЯ РОБОТИ ПРОЦЕСІВ

6.1 Опис тестових випадків

Для тестування системи обрано метод «чорної скриньки», за допомогою тест-кейсів. Це дозволяє протестувати роботу окремих функцій програми та реакцію програми на дії користувача при роботі з програмний продуктом.

Нижче описані тести для окремих варіантів використання.

В табл. 6.1 наведений тест-кейс для варіанту використання «Додавання тесту», передумова – авторизований користувач з правами доступу викладача.

Таблиця 6.1 - Тест-кейс варіанту використання «Додавання тесту»

Крок	Опис	Очікуваний результат	Виконано
1	Обрати пункт меню «Тести»	На екрані з’явився список тестів	Успішно
2	Натиснути кнопку «Додати»	На екрані з’явилися поле для внесення назви тесту та прапорець для вказівки, чи є тест навчальним або контролюючим	Успішно
3	Ввести назву «Новий тест»	Назва введена	Успішно
4	Змінити значення прапорця	Значення змінено	Успішно
5	Натиснути кнопку «Зберегти»	Поля для введення зникли, новий тест з’явився у списку тестів	Успішно

Продовження табл. 6.1

Крок	Опис	Очікуваний результат	Виконано
6	Натиснути кнопку «Додати»	На екрані з’явилися поле для внесення назви тесту та прапорець для вказівки, чи є тест навчальним або контролюючим	Успішно
7	Натиснути кнопку «Скасування»	Поля для введення зникли	Успішно

В табл. 6.2 наведений тест-кейс для варіанту використання «Редагування запитання», передумова – авторизований користувач з правами доступу викладача.

Таблиця 6.2 - Тест-кейс варіанту використання «Редагування запитання»

Крок	Опис	Очікуваний результат	Виконано
1	Обрати пункт меню «Питання»	На екрані з’явився список питань	Успішно
2	Двічі клікнути по одному з питань	На екрані з’явився список відповідей для питання	Успішно
3	Натиснути кнопку «Редагувати» над списком питань	На екрані з’явилися поле з текстом питання та випадаючий список з типами питань, в якому виділений тип поточного питання	Успішно

Продовження табл. 6.2

Крок	Опис	Очікуваний результат	Виконано
4	Змінити текст питання	Текст змінено	Успішно
5	Змінити тип питання	Тип змінено	Успішно
6	Натиснути кнопку «Зберегти»	Поля опису питання зникли, в списку питань змінено дані	Успішно

В табл. 6.3 наведений тест-кейс для варіанту використання «Додавання відповіді», передумова – авторизований користувач з правами доступу викладача.

Таблиця 6.3 - Тест-кейс варіанту використання «Редагування запитання»

Крок	Опис	Очікуваний результат	Виконано
1	Обрати пункт меню «Питання»	На екрані з’явився список питань	Успішно
2	Двічі клікнути по одному з питань	На екрані з’явився список відповідей для питання	Успішно
3	Натиснути кнопку «Додати» над списком відповідей	На екрані з’явилися поле для внесення тексту відповіді та прапорець для вказівки. чи є відповідь вірною	Успішно
4	Внести текст відповіді	Текст внесено	Успішно
5	Змінити значення прапорця	Значення змінено	Успішно

Продовження табл. 6.3

Крок	Опис	Очікуваний результат	Виконано
6	Натиснути кнопку «Зберегти»	Поля опису відповіді зникли, в списку відповідей з’явилася нова відповідь	Успішно

Таким чином, розроблено та виконано тест-кейси для тестування функцій програми. Усі тести пройдено успішно, що підтверджує працездатність застосунку.

6.2 Експериментальне дослідження результатів роботи програми

Для перевірки досягнення мети роботи – скорочення часу на навчання теми планування роботи процесів в операційній системі – проведено експериментальне дослідження.

В експерименті приймали участь 18 студентів.

Всі студенти прослухали лекцію з теми «Планування роботи процесів в операційній системі». Далі всі студенти отримали по 2 завдання навести схему роботи процесів для двох випадково обраних алгоритмів з множини можливих

Перша група з 6 студентів отримала завдання у вигляді тестів у розробленій програмі та можливість пропрацювати навчаючі тести з задачами. Друга група з 6 студентів отримала завдання у вигляді тестів у розробленій програмі без можливості пропрацювати навчаючі тести з задачами. Третя група з 6 студентів отримала завдання у паперовому вигляді.

Всі групи мали однаковий час на виконання завдання.

Результати тестування наведені у табл. 6.4, визначався час на отримання першого результату (Т), кількість помилок у відповіді на завдання (Р), кількість спроб для отримання вірного результату (С) (для третьої групи додатково проведена консультація перед наступними спробами).

Таблиця 6.4 – Результати експериментального дослідження

Студент	Група 1			Група 2			Група 3		
	Т, хв	Р	С	Т, хв	Р	С	Т, хв	Р	С
1	15	2	3	12	1	2	21	2	2
2	13	0	1	15	2	2	17	0	1
3	16	2	2	20	2	3	33	5	3
4	15	0	1	17	1	2	15	7	5
5	14	1	2	22	0	1	24	3	2
6	10	0	1	15	1	2	30	0	1
Середнє значення	13,83	0,83	1,67	16,83	1,17	2,00	23,33	2,83	2,33

6.3 Висновки до розділу

В розділі надано опис тест-кейсів для окремих варіантів використання, спроектовані тести дозволили перевірити відповідність роботи програми заданим функціональним вимогам за допомогою методу «чорної скриньки». За допомогою експерименту показано, що використання автоматизованої системи тестування з можливістю пройти навчаючі тести, в які входять специфічні задачі для візуалізації поведінки процесів в операційній системі, значно скоротило час на отримання позитивних результатів при вирішенні завдань.

ВИСНОВКИ

В роботі спроектовано та реалізовано програму, за допомогою якої можна вирішувати як задачі універсального тестування, так і включення специфічних задач на поведінку операційної системи при планування роботи процесів з використанням різних алгоритмів.

Проведено аналіз існуючих програмних продуктів для тестування знань, універсальних та спеціалізованих. Показано, що для вирішення задачі тестування знань з роботи операційних систем автоматизованих засобів не існує.

Проаналізовано особливості предметної області та сформульовано вимоги до програмного продукту, функціональні та нефункціональні. Функціональні вимоги описано у вигляді варіантів використання, з детальними сценаріями.

Розроблено алгоритми побудови схеми поведінки процесів для довільної кількості взаємодіючих процесів для різних алгоритмів планування.

Створено базу даних для зберігання інформації про тести, питання та умови задач, а також результати виконання тестів студентами.

Реалізовано інтерфейс у вигляді веб застосування.

Підтверджено працездатність програми за допомогою тест-кейсів при тестування методом «чорної скриньки» та за допомогою експерименту показано досягнення мети розробки – скорочено час на отримання вірного рішення задач на планування роботи процесів в 1,6 разів.

СПИСОК ЛІТЕРАТУРИ

1. Алыкова А.Л. Особенности автоматизированного тестирования знаний студентов в области программирования. - Вестник ИГЭУ. - Вып. 4, 2005, с. 1-4.

2. 2Обучающие программы. - URL: https://sites.google.com/site/elobrasres/programmnye-sredstva-ucebnogo-naznacenia/obucausie-programmy

3. Класична теорія тестування. - URL: https://sites.google.com/site/klasicnateoriatestuvanna/video

4. Малыгин А. Система автоматизированного тестирования знаний. URL: https://cadregion.ru/produkty/sistema-avtomatizirovannogo-testirovaniya-znanij.html

5. Фетисов В. С. Основные требования к компьютерным системам тестирования знаний (КСТЗ). – Педагогические измерения, 2015, № 3, с. 39-48.

6. Гладких И. Ю. Якушин А.В. Системы автоматизированного тестирования по программированию в образовательном пространстве. Современные проблемы науки и образования. – 2016. – № 3.

7. 7Обучающие компьютерные программы. URL: https://sites.google.com/ site/obuchkomprog/home/informacionnye-razdely/tipy-komputernyh-obucausih-programm/klassifikacia

8. Компьютерная обучающая программа: определение и основные особенности. URL: https://sites.google.com/site/obuchkomprog/home/informacionnye-razdely/komputernaa-obucausaa-programma-opredelenie-i-osnovnye-osobennosti

9. Електронні засоби навчання Педагогічне програмне забезпечення. Електронний підручник. URL: https://pedagogy.lnu.edu.ua/wp-content/uploads/2016/11/%D0%95%D0%9B%D0%95%D0%9A%D0%A2%D0%A0%D0%9E%D0%9D%D0%9D%D0%86-%D0%97%D0%90%D0%A1%D0%9E%D0%91%D0%98-%D0%9D%D0%90%D0%92%D0%A7%D0%90%D0%9D%D0%9D%D0%AF.pdf

10. Обучающие программы. URL: https://sites.google.com/site/infortechvobrazovanii/komputerizacia-skolnogo-obrazovania/obucausie-programmy

11. Изучай программирование на ITVDN. URL: https://itvdn.com/ru

12. Тыщенко О.Б. Диалог компьютера и студента. Высшее образование в России, 2000, №6, с. 120-123. (

13. Г.В. Лаврентьев, Н.Б. Лаврентьева. Инновационные обучающие технологии в профессиональной подготовке специалистов. - Алтайский гос. ун-т, Барнаул, 2009. - 166 с.

14. Васильков Ю. В., Заботина Н. Н., Гущина Л. С. Роль и опыт применения информационных технологий в формировании компетентностного подхода в образовании. Научно-методический электронный журнал «Концепт». – 2016. – Т. 37. – С. 30–37.

15. Воронин Ю.А. Компьютеризированные технологии в процессе подготовки учителя. - Педагогика. - 2003. - № 8. – С. 53-59.

16. Тема 15 Моделирование как метод познания. Классификация и формы представления моделей. URL: https://www.altstu.ru/media/f/Tema-15-Modelirovanie.pdf

17. Velykodniy, S. S. Analysis and Synthesis of the Results of Complex Experimental Research on Reengineering of Open CAD Systems. *Applied Aspects of Information Technology. Publ. Science i Technical*. Odessa: Ukraine. 2019; Vol. 2 No. 3: 186–205. DOI: https://doi.org/10.15276/aait.03.2019.2

18. Алгоритми планування процесів у сучасних ОС. URL: https://wiki.cuspu.edu.ua/index.php/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B8_%D0%BF%D0%BB%D0%B0%D0%BD%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%96%D0%B2_%D1%83_%D1%81%D1%83%D1%87%D0%B0%D1%81%D0%BD%D0%B8%D1%85_%D0%9E%D0%A1.

19. Планирование процессов. URL: http://komputercnulja.ru/operacionnye-sistemy/planirovanie-processov.

20. Технології у вашому класі. URL: https://www.microsoft.com/uk-ua/education/educators/classroom-technology

21. Sway для образовательных учреждений. URL: https://sway.office.com/education/?WT.mc_id=WEDUNAVSEDU_WEB_Persistent

22. Організація роботи в класі з OneNote. URL: https://www.microsoft.com/uk-ua/education/products/onenote

23. Grissom T. 10 best uses for OneNote in your teaching and learning. URL: https://educationblog.microsoft.com/en-us/2017/03/10-best-uses-for-onenote-in-your-teaching-and-learning/

24. Microsoft Teams для освіти. URL: https://www.microsoft.com/uk-ua/education/products/teams

25. Сведения о Google Классе. URL: https://support.google.com/edu/classroom/answer/6020279?hl=ru

26. Use Cases. URL: https://www.usability.gov/how-to-and-tools/methods/use-cases.html

27. Браш К. Вариант использования. URL: https://searchsoftwarequality.techtarget.com/definition/use-case

28. Use Case Examples. https://www.gatherspace.com/use-case-examples/

29. Non-functional Requirements: Examples, Types, How to Approach. URL:https://www.altexsoft.com/blog/non-functional-requirements/

30. Архитектура MVC. URL:https://github.com/codedokode/pasta/blob/master/arch/mvc.md

31. Архитектура приложения. URL: https://php.zone/oop-v-php-prodvinutyj-kurs/ arhitektura-prilozheniya-i-pattern-mvc

32. Kungurtsev, O. B., Novikova, N. O. Identification of Class Models Imperfection. Herald of Advanced Information Technology. Publ. Science i Technical. 2020; Vol. 3 No. 2: p. 13–22. Odesa: Ukraine. DOI: https://doi.org/10.15276/hait.02.2020.1