МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки,
Факультет радіоелектроніки та телекомунікацій
(повне найменування інституту, факультету)

Кафедра інформаційних технологій електронних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)
бакалавра
(ступінь вищої освіти)

на тему «РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ РОБОТИ ДЕКАНАТУ «РОЗКЛАД» »

Виконав:студент 4 курсу, групи РТ-517
Спеціальності 172 Радіотехніка та Телекомунікації
(код і найменування спеціальності).
Освітня програма (спеціалізація)
Інтелектуальні технології мікросистемної радіоелектронної техніки
Живогляд В.А.
(прізвище та ініціали)
Керівник Куляба Т.І.
(прізвище та ініціали)
Рецензент Польська О.В
(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут інформатики та радіоелектроніки, факультет радіоелектроніки та телекомунікацій
Кафедра інформаційних технологій електронних засобів
Ступінь вищої освіти бакалавр
Спеціальність 172 «Телекомунікації та радіотехніка»
(код і найменування)
Освітня програма (спеціалізація) Інтелектуальні технології мікросистемної радіоелектронної техніки
(назва освітньої програми(спеціалізації)

ЗАТВЕРДЖУЮ
В.о. зав. каф. ІТЕЗ Огренич Є.В.,
канд. техн. наук
«27» 05 2021 року

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТА

Живогляд Віталій Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи: Розробка автоматизованої системи роботи деканату «Розклад»
Затверджена наказом ректора від « 22 « травня 2021 р. №.
2. Термін виконання роботи: з 16 квітня 2021 року до 31 травня 2021 року
3. Вихідні дані: праці вітчизняних та зарубіжних авторів та інша наукова література за темою дипломної роботи, автоматизована система, програмна система, критерій якості, алгоритми роботи та реалізація з системами складання автоматизованого розкладу занять.
4. Зміст пояснювальної записки (перелік питань, що підлягають розробці):
   4.1 Вступ
   4.2 Огляд та аналіз процесу складання розкладів в навчальних закладах
   4.3 Аналіз існуючих програмних продуктів, що автоматизують процес складання розкладу
   4.4 Проектування автоматизованої інформаційної системи складання розкладу
   4.5 Розробка програми
   4.6 Висновки
5. Перелік обов'язкового графічного матеріалу: Демонстраційний матеріал у форматі презентації PowerPoint (*.ppt) – 17 с. формату А4

Календарний план-графік

№ пор	Завдання	Термін виконання	Відмітка про виконання
1	Вибрати тему дипломної роботи	до 06.04.2021	
2	Затвердити тему і план роботи у наукового керівника	до 10.04.2021	
3	Визначити статистичну, інформаційну базу дослідження, скласти бібліографію	до 17.04.2021	
4	Оформити і обговорити з науковим керівником перший розділ роботи	до 27.04.2021	
5	Оформити і обговорити з науковим керівником другий розділ роботи	до 05.05.2021	
6	Доопрацювати роботу, оформити її кінцевий варіант	до 14.05.2021	
7	Отримати відгук керівника та рецензію	до 14.05.2021	
8	Підготувати доповідь на захист	до 15.05.2021	

1. Консультанти з окремих розділів

Розділ	Консультант (посада, П.І.Б)	Дата, підпис	
		Завдання видав	Завдання прийняв
Н/К	Гонєва У.Є.		

2. Дата видачі завдання: 06.05.2021 р.

Керівник дипломної роботи _____ Кулеба-Харитонова Т.І.
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Живожлуд В.І.
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

ПЗ: 55 с., 8 рис., 26 джерел.

РОЗКЛАД ЗАНЯТЬ, ФОРМУВАННЯ РОЗКЛАДУ, АВТОМАТИЗОВАНА СИСТЕМА, ЗАКЛАД ВИЩОЇ ОСВІТИ, ДОКУМЕНТООБІГ, НАВЧАЛЬНЕ ЗАНЯТТЯ, НАВЧАЛЬНИЙ ПРОЦЕС, НАВЧАЛЬНЕ НАВАНТАЖЕННЯ, КАФЕДРАЛЬНА ЗАЯВКА, ВИТЯГ ІЗ НАВЧАЛЬНИХ ПЛАНІВ, ОПТИМІЗАЦІЯ, РОЗПОВСЮДЖЕННЯ, ІНТЕРНЕТ-СЕРВІС, ПРОГРАМНА СИСТЕМА, ПІДСИСТЕМА, АУДИТОРНИЙ ФОНД, СПЕЦІАЛЬНІСТЬ, КРИТЕРІЙ ЯКОСТІ, ПРОГРАМНИЙ ПРОДУКТ

Об'єктом дослідження: діяльність методичного відділу деканату Національного університету «Запорізька Політехніка».

Предмет дослідження: створення автоматизованої системи розкладу занять.

Мета дослідження: розробка автоматизованої системи роботи деканату «Розклад» за допомогою сучасних технологій.

Методи дослідження: Для вирішення поставлених завдань використовувалися метод автоматизованого формування розкладу за допомогою використання генетичних алгоритмів.

Наукова новизна полягає в реалізації автоматизованої системи роботи деканату «Розклад».

Отримані результати можуть використовуватись у майбутніх дослідженнях за напрямком дослідження формування розкладу.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

Генетичний алгоритм – це еволюційний алгоритм пошуку, що використовується для вирішення задач оптимізації і моделювання, шляхом послідовного підбору, комбінування і варіації шуканих параметрів з використанням механізмів, що нагадують біологічну еволюцію. [1]

Ген – реально існуюча, незалежна одиниця спадковості, що комбінується й розщеплюється при схрещуваннях;

Жорсткі обмеження – це обмеження, які повинні неодмінно задовольнятися; такі які фізично не можуть бути порушені.

М’які обмеження – це обмеження, які можна порушувати, але це порушення повинно бути зведене до мінімуму. Їхнє виконання не є таким же обов’язковим, як жорстких.

Тех. підтримка – технічна підтримка, яка полягає у зворотному зв’язку між розробником та користувачем, при якому перший може допомагати іншому у вирішенні проблем зв’язаних з продуктом.

СУБД – Система управління базами даних

SQLite – це вбудована бібліотека, яка реалізує автономний, безсерверний, нульової конфігурації, транзакційний механізм СУБД SQL.

C++ – мова програмування високого рівня з підтримкою кількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної.

ВСТУП

Інформаційні технології стали невід'ємною частиною не тільки повсякденному житті людей, але і в діяльності інститутів. Величезна кількість процесів підлягає автоматизації в справжніх умовах інформатизації, в тому числі і процес складання розкладу, який є одним з основних організаційних етапів у навчальних закладах, який і буде досліджений в даній роботі.

Проблема складання розкладу занять у ВНЗ є предметом дослідження в роботах багатьох авторів. У сучасних умовах розвитку вищої освіти стає очевидною необхідність використання автоматизованих засобів планування і складання розкладу навчальних занять.

Процес складання розкладу заснований на аналізі значної кількості інформації і вимагає значних працевитрат. Ефективне управління ресурсами є складною і одним із першочергових завдань, що стоять перед ВНЗ в процесі надання освітніх послуг. Разом з тим, ефективне управління діяльністю організації в сучасному динамічному зовнішньому і внутрішньому середовищі неможливо без інформаційної підтримки.

Дослідження в області теорії розкладу проводились такими вченими М. Юнгінгер, Д. де Верра Лінг, Фаріон і Долланскі, Е. Бурке, А. Н. Безгін і С. Ю. Трегубов, Г. А. Попов, В. А Атрощенко., І. С. Семенюта, А. М. Донецьков, Е.Бурк і С. Петровік, але до сьогоднішнього дня ця проблема не вирішена повністю як в теорії, так і на практиці.

Метою дослідження існуючих теорій, комплексу моделей і методів оптимізації розкладу занять у вищому навчальному закладі є формування єдиної концепції складання розкладу в навчальних закладах з урахуванням виявлених їхніх переваг та недоліків.

Завдання складання розкладів є предметом наукових досліджень з середини минулого століття. Завдання планування розкладу навчальних

занять у ВНЗ є складання розкладу автоматизованої системи, характерною особливістю якої є дуже велика розмірність і наявність великого числа обмежень складної форми.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- розглянути і вивчити теоретичні аспекти, а також особливості процесу складання розкладів в навчальних закладах;

- проаналізувати існуючі програмні продукти, що автоматизують процес складання розкладів;

- дослідити існуючий процес складання розкладів;

- спроектувати автоматизовану інформаційну систему складання розкладу в навчальних закладах.

Об'єктом дослідження виступає діяльність методичного відділу деканату Національного університету «Запорізька Політехніка».

Актуальність полягає в необхідності автоматизації процесу складання розкладу для швидкого знаходження найоптимальнішого варіанту розкладу.

1 ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗОВАНОЇ СИСТЕМИ РОЗКЛАДУ ЗАНЯТЬ

1.1 Аналіз стану розробок в галузі створення розкладу

У сучасних умовах розвитку вищої освіти стає очевидною необхідність використання автоматизованих засобів планування та складання розкладу навчальних занять. Процес складання розкладу заснований на аналізі значної кількості інформації та вимагає значних трудовитрат. Ефективне управління ресурсами є складною і одним із першочергових завдань, що стоять перед вузами у процесі надання освітніх послуг. Разом з тим, ефективне управління діяльністю організації у сучасному динамічному зовнішньому та внутрішньому середовищі неможливо без інформаційної підтримки.

Дослідження в області теорії розкладу проводились такими вченими М. Юнгінгер, Д. де Верра Лінг, Фаріон і Долланскі, Е. Бурке, А. Н. Безгін і С. Ю. Трегубов, Г. А. Попов, В. А Атрощенко., І. С. Семенюта, А. М. Донецьков, Е.Бурк і С. Петровік, але до сьогоднішнього дня ця проблема не вирішена повністю як в теорії, так і на практиці. Метою дослідження існуючих теорій, комплексу моделей та методів оптимізації розкладу занять у вищому навчальному закладі є формування єдиної концепції складання розкладу в навчальних закладах з урахуванням виявлених їхніх переваг та недоліків. Завдання складання розкладів є предметом наукових досліджень з середини минулого століття. Завдання планування розкладу навчальних занять вузу є складання розкладу автоматизованої системи, характерною особливістю якого є дуже велика розмірність та наявність великого числа обмежень складної форми [1].

Розклад занять – документ, що регламентує організацію та проведення процесу навчання у ВНЗ. Удосконалення навчального процесу ВНЗ на практиці частіше за все здійснюється на основі планування занять.

Такі дослідники як А. А. Лазарєв, О. Р. Гафаров, А. Н. Безгін і С. Ю. Трегубов, Бурк Е. і Петровік С., у своїх роботах доводять, що принципи, комплекс моделей та методів, а також логіка оптимізації розкладу занять у вищому навчальному закладі, можуть претендувати на науковий базис теорії розкладу.

Перші дослідження даної проблеми були представлені в 60-х роках двадцятого століття [16, 17, 22]. З 1963 р. починаючи з робіт К. Готлієба проводилися спроби визначити та систематизувати завдання складання розкладу, а також визначити підходи до їх автоматизації [21]. Так, дослідниками Шмідтом та Строхлейном в 1979 було зібрано близько 200 праць з проблеми автоматизації складання розкладів [21]. У 1986 р доктор М. Юнгінгер провів дослідження проблеми складання розкладу в університетах Німеччини. Було описано програмне забезпечення, що реалізує функції складання розкладу, а також запропонований підхід до вирішення завдання, ґрунтуючись на методі прямої евристики [17].

Для формулювання задачі складання розкладу застосовувалися методи лінійного програмування (1970 – 1980 рр.) [16, 18], мережеві моделі (1980 р.) [22], логічне програмування в обмеженнях (1990-ті) [18, 20]. В останні десятиліття набуло розвитку напрямок пошуку ефективних евристичних методів рішення задачі складання розкладу занять вузу. Представлені праці зарубіжних авторів (Лінг (1992), Фаріон і Долланскі (1994 г.), Бурке (2004)). Проблеми автоматизації складання розкладу розглядалися в роботах багатьох дослідників: М. М. Клеванського, Р. В. Конвея, Є. Г. Коффмана, В. Л. Максвелла, М. Г. Маслова, Л. В. Міллера, В. Г. Пайкерса та ін.

Кількісний та якісний ріст вищої освіти вимагає нового підходу до вирішення проблем управління навчальної, наукової та економічної діяльності університетів. В останні роки цей підхід привів до використання сучасних комп'ютерних технологій та математичних методів в управлінні вищими навчальними закладами. У сучасному світі все частіше використовуються різні типи систем автоматизації технологічних процесів,

які завжди виконувалися вручну. Інтерес до питань автоматизації складання розкладу виник та став рости з розвитком інформаційних та комп'ютерних технологій.

1.2 Переваги використання системи автоматизованого розкладу в університеті

Завдяки процесу інформатизації за останні 15-20 років, комп'ютерні технології стали важливою частиною життя людини в якості загального інформативного ресурсу та знаряддя соціалізації. За допомогою цієї обставини вирішальним чином змінюються цінності системи управління освітою.

У нинішньому інформативному світі пізнання стрімко втрачають свою актуальність. Одночасно з цим процес підготовки кваліфікованих кадрів залишився важким та досить тривалим. Можна сказати, що на даний момент основне завдання вищої освіти в тому, щоб навчити учнів самостійно знаходити та застосовувати знання. На практиці це можна впровадити лише з використанням в освітньому процесі новітніх технологій, які спрямовані на досягнення позначених нами цілей навчання.

Актуальність автоматизації у навчальному закладі пояснюється новими вимогами до оптимізації навчального процесу та зменшенню трудовитрат з боку керівництва. Зменшуючи витрати на повсякденні рутинні роботи, у методистів залишається більше часу на роботу з педагогічним складом, учнями та робочими програмами. Однак, автоматизація – це трудомісткий процес, який неодмінно супроводжується низкою проблем.

На сьогоднішній день використання інформаційних технологій в освітніх установах не рідкість. Інформатизацію застосовують не тільки в освітніх технологіях, але і безпосередньо в управлінні освітнім процесом, це:

бухгалтерія, кадровий склад, компонування навчальних планів та річних навантажень, навчального розкладу.

Розклад для вузів вельми відрізняється від розкладу для шкіл: так як в школах навчаються неповнолітні та під час навчання за них відповідають вчителі, то по шкільному розкладом учні повинні бути повністю зайняті (без «вікон»), в той час як для університетського це зобов'язання відсутнє [1].

Частіше стали набувати популярності програми для складання розкладу, адже в традиційній формі це досить трудомісткий процес. Розклад становили вручну, на папері, що не виключало накладок з розкладом викладачів та аудиторіями.

Автоматизований розклад в свою чергу дозволяє майже повністю уникнути цих проблем. Крім цього, незаперечною перевагою такого програмного забезпечення є виведення необхідних звітів, довідок та виписок окремо для кожного викладача, які можна відправляти по електронній пошті прямо з програми.

Програма призначена для вирішення завдань автоматизованого складання навчальних розкладів та оперативного управління приміщеннями у ВУЗах. З її допомогою складати розклад можна в автоматичному, ручному та змішаному режимах з урахуванням багатьох обмежень та умов. При цьому можна побудувати як допустимий розклад, так і оптимізоване, в якому скорочено кількість вікон або кількість використовуваних приміщень [2, с. 9].

Якщо навчальний заклад має у своєму розпорядженні декілька корпусів, то програма враховує час переміщення між ними. При традиційній системі формування розкладу, на кожному факультеті були задіяні мінімум два методиста – на очне та заочне відділення, а при інноваційному підході один фахівець автоматизованого розкладу може керувати розкладами кількох навчальних підрозділів без зайвих трудовитрат.

Зручність використання таких програм так само полягає в тому, що можна заздалегідь виставляти пріоритети викладачів, переваги, звичайно,

віддаються загально університетським викладачам: фізична культура, іноземна мова, філософія та інші, так як у цих співробітників найбільше навантаження.

Не дивлячись на те, що вигоди від застосування автоматизованих програм очевидні, залишається ряд проблем, які належить вирішити з часом. Основна проблема – це фінансування. Якісні та ліцензійні програми коштують істотних грошових коштів, а також припускають регулярне обслуговування та оновлення. Безсумнівно, постає гостре питання – як виділити бюджет на основні витрати при впровадженні автоматизованих програм.

Виникають також питання організаційного та технологічного характеру. Складнощі виникають із синхронізацією між структурними підрозділами. Існує необхідність створення єдиної бази даних для всіх відділів, щоб комфортніше було використовувати інформацію про навчальний процес. Оптимальним вирішенням цієї проблеми стане створення «Єдиного інформаційного вікна». Таким чином, потреба у вирішенні завдань автоматизації побудови розкладу занять буде зростати, що обумовлено розвитком освітньої системи та переходом до більш широкого використання індивідуальних траєкторій навчання.

Автоматизація науково-технічних дій інститутів зверне нагляд та регулювання в вищих навчальних закладах з буденного та напружуючого навчання у дуже добре систематизовану активність, що приносить задоволення фахівця. Разом з правильно вибудуваної, технологічною системою організації буде так само рости і загальна ефективність управління всього інституту. Автоматизація навчального процесу вузу допоможе організувати операції у відповідність до запитів та потреб організації. Зручна система пошуку полегшить знаходження необхідних відомостей. Правильно та ефективно вибудовані фільтри допоможуть швидко знайти необхідну інформацію у базі даних.

Якісне зростання вищих навчальних закладів запитує новий підхід для вирішення завдань управління навчальною та господарською діяльністю ВНЗ. Цей підхід в останні роки знаходить своє втілення в застосуванні сучасних засобів обчислювальної техніки та математичних методів в управлінні вищими навчальними закладами. У сучасному світі все більшого поширення набувають різного роду системи автоматизації технічних процесів, які завжди виконувалися вручну. Наприклад, системи прийняття рішення в маркетингу, експертні системи, що замінюють досвідчених фахівців, прогнозують системи в самих різних областях науки та техніки. До таких же процесів відноситься і складання розкладу, яке до сих пір у багатьох навчальних закладах створюється вручну на основі багаторічного досвіду.

В даний час неможливо уявити роботу практично будь-яких видів діяльності без автоматизації. Вона необхідна для прискорення складних операцій, в тому числі розрахунків. Автоматизація практично виключає людський фактор, дозволяє контролювати виконання трудових функцій всієї установи.

Проводити автоматизацію процесів слід для того, щоб уберегти установу від зайвих витрат. Деякі керівники вважають за краще справлятися силами і вміннями своїх співробітників. Але придбати певне програмне забезпечення доступно багатьом, витрати на нього швидко окупаються.

Автоматизація особливо необхідна в тих установах, де чисельність співробітників перевищує кілька десятків людей. Керівникам неможливо контролювати велику кількість працівників без застосування сучасних інформаційних систем.

Всі автоматизовані процеси мають високу швидкість. Генерація найскладніших звітів йде лічені хвилини. Працівник з відповідною кваліфікацією виконає їх не менше ніж за тиждень, навіть якщо буде залишатися після зміни.

Правильно вибране програмне забезпечення буде простим та доступним для користувачів. Освоїти автоматизований процес зможе навіть студент, який не має досвіду роботи та необхідної кваліфікації. Персональні комп'ютери і встановлені на них спеціальні програми не допускають помилок. Якщо враховувати всі обмеження, що пред'являються до розкладу, то завдання ручного складання розкладу помітно ускладняється. Виходом зі сформованої ситуації є автоматизація процесу створення навчального розкладу. Автоматизація процесів дозволяє мінімізувати людський фактор, адже машини не втомлюються, не лінуються. Програми забезпечують безпеку збережених даних на персональному комп'ютері.

Як відомо, навчальний план є документом, що лежить в основі організації процесу навчання з конкретної спеціальності навчання в будь-якій освітній системі. Від того, як він складений, багато в чому залежить як якість підготовки фахівців, так і витрати освітнього закладу на реалізацію освітнього процесу. Питання підвищення якості освіти особливо гостро постають у зв'язку з появою великої кількості різних освітніх установ, що використовують нетрадиційні технології. Хороша фундаментальна підготовка, а також базові теоретичні загально професійні знання є основною відмінною рисою університетської освіти. Вони забезпечують випускникові успіх як в чисто професійної області, так і в соціальній сфері, підвищуючи його соціальну захищеність завдяки можливості зміни спрямованості своєї роботи [18].

Істотне оновлення навчальної бази, а також використання інформаційних технологій в плануванні та організації навчального процесу дозволяє говорити про можливість підвищення якості підготовки фахівців. Неодноразово підкреслювалося, що найважливішим завданням найближчого часу стане розробка та впровадження механізмів забезпечення якості освітнього процесу.

Крім того, існуючі методи формування навчальних планів орієнтовані на отримання одного навчального плану для кожної спеціальності навчання,

що ускладнює оптимальне стикування різних навчальних планів в рамках одного навчального закладу, а також в тому, щоб отримати оцінку ресурсних витрат при відкритті нової спеціальності. Висока трудомісткість складання навчальних планів внаслідок відсутності теоретичного інструментарію, що лежить в основі автоматизації процесу складання навчальних планів.

Навчальний план є основним документом планування процесу підготовки фахівця з конкретної спеціальності. Сформовані навчальні плани повинні забезпечувати якісну підготовку фахівців. З іншого боку вони повинні бути реальними в плані витрат освітнього закладу на реалізацію освітнього процесу, який проводиться відповідно до цих навчальними планами.

Отже, процес автоматизації ВНЗ є необхідним процесом для переходу на управління нового рівня: формування певних керуючих та організаційних структур вузу, можливості взаємодії між структурами. Правильне функціонування дозволить уникнути появи дублів документів та непотрібних дій, що суттєво прискорить прийняття своєчасних та грамотних рішень.

1.3 Основні проблеми автоматизації складання розкладу

Розклад занять у вищому навчальному закладі служить для зведення в єдину взаємопов'язану систему викладачів, навчальних предметів та призначених для проведення занять місць – аудиторій. Оптимізація розкладу занять є одним з основних чинників, здатних істотно оптимізувати навчальний процес [1].

Одна з проблем, яка виникає на етапі впровадження автоматизованого розкладу, це питання фінансування. Тут керівництву необхідно вирішити – чи буде купуватися спеціальне програмне забезпечення або такий ресурс буде розроблений безпосередньо кваліфікованими програмістами на базі університету. І той, і інший варіант мають свої переваги. Перший варіант

передбачає вибір серед безлічі продуктів, які вже були апробовані на базах інших навчальних закладів, але це вимагає покупки дуже дорогого ліцензійного продукту з подальшим технічним та гарантійним обслуговуванням.

Важливою проблемою впровадження автоматизованого розкладу є неготовність викладачів включитися в нову систему. Педагоги «старої школи» звикли звертатися на кафедру для зміни розкладу з особистих причин, обмінюватися між собою навчальними групами, скорочувати навантаження, об'єднувати підгрупи в групи та групи в потік. При використанні автоматизованої системи це є неможливим з огляду на повної відповідності робочим навчальним планам та загальної погодинної навантаження.

Оптимізація навчального процесу передбачає контроль над виконанням навчального навантаження, що, на жаль, не завжди подобається викладачам та сприймається досить гостро. При складанні розкладу в програмі враховується тільки навантаження, а не конкретні побажання викладача, як це було раніше. Абсолютно нормально розуміти процес оптимізації навчального навантаження. Рекомендується уникати збігів з аудиторією. При складанні розкладу в цьому випадку рекомендується розставляти такі заняття в першу чергу, щоб уникнути накладок з аудиторіями.

Наступна проблема з впровадженням автоматизованого графіка – знайти кваліфікований персонал, який буде обробляти та компілювати всю інформацію. Такі фахівці повинні не тільки мати можливість використовувати стандартні комп'ютерні програми, а й навчатися роботі зі спеціальним програмним забезпеченням, вони проходять навчання безпосередньо в процесі роботи під час апробації автоматизованої системи, що може значно уповільнити процес планування. Вони повинні знаходитися під наглядом наставника, який вже має досвід. Крім того, фахівці відділу автоматизованого розкладу відповідають за організацію навчального процесу, що означає, що проблеми можуть виникнути через людський

фактор через відпустки через хворобу, навчання співробітників, відрядження та відпустки.

Таким чином, автоматизована система розкладу в даний час має ряд проблем, але її інтерфейс має велику область застосування на практиці та ефективну реалізацію в навчальному процесі.

У багатьох університетах розклад складено в ручному режимі, воно складено на картонних табличках, а сітка розкладу застосовується до них невеликим почерком з олівцем. У тій же формі вони поширюються на факультети та кафедри. Знайти правильну інформацію або зробити конкретний зразок у такій ситуації вкрай складно.

Тому, як мінімум, необхідно автоматизувати запис та зберігання інформації про розклад класів. Програмні продукти дозволяють вам успішно створювати розклад занять в автоматичному режимі, але в той же час губляться досвід та стабільна звична структура розкладу, отримана за допомогою ручної компіляції. Крім того, пропонована програма використовує локальний підхід, тобто автоматизацію тільки одного відділу, відповідального за планування. Співробітники цього відділу зобов'язані проводити трудомісткий процес введення вихідної інформації в єдину базу даних.

Вивчення досвіду створення таких систем показує, що в останні роки було зроблено багато спроб поліпшити планування навчального процесу, побудувавши алгоритми для оптимізації задач планування навчальної роботи установи. Вища освіта та її подальша реалізація з комп'ютерних технологій. Ці дослідження в різний час були проведені та тривають в деяких університетах [4,5].

Проте, практична реалізація освітнього планування з використанням комп'ютерних технологій відбувається тільки в декількох університетах. Аналіз стану цих подій призводить до наступних висновків:

- розробка та впровадження університетами завдань АСУ з ініціативи, ця робота зазвичай спрямована на вирішення індивідуальних проблем.

Різноманітність дослідницьких груп та розробників призвело до створення цілого ряду систем для розробки алгоритмів та програм, призначених для обслуговування університету;

- багато систем дають програмісту повну відповідальність за облік реальних потреб. Зокрема, з урахуванням потреб вчителів, обмеженість кількості уроків в день і на тиждень – всі ці рутинні завдання, як і багато інших, повинні вирішуватися людиною навмання, найчастіше за допомогою методів в силі;

- доступні програми не включають багато користувачів режим роботи та не підтримують всі необхідні потоки електронних документів;

- розробка єдиних елементів, типових для створення єдиної автоматизованої системи управління вищими навчальними закладами – інтеграційний портал майже не існує;

- доступна програма має дуже зручний інтерфейс для введення вихідних даних та коригування програми.

Паралельно з розширенням роботи по вдосконаленню системи управління вищою освітою шляхом створення та впровадження різних автоматизованих систем управління в університетах виникла необхідність уніфікувати засоби створити програму комп'ютерного навчання. Для цього необхідно чітко оформити вимоги до календаря та розробити відповідну алгоритмічну підтримку.

При розробці алгоритмів автоматичного планування класів існує гостра проблема створення універсальних алгоритмів, що враховують конкретні умови кожного конкретного завдання. Ці алгоритми повинні бути досить «гнучкими», тобто без істотних змін можна було б включати та виключати вимоги з системи вимог до календаря. Однак спроба вирішити проблему за допомогою єдиного універсального алгоритму в даний час неможлива. Алгоритми рішення великого числа завдань не забезпечують ефективності, що забезпечується більш конкретними алгоритмами, адаптованими до конкретних умов.

Для систем планування сильна залежність від специфіки конкретних навчальних закладів вже знаходиться на рівні математичних моделей та представлення даних, що ускладнює використання звичайних систем. Система, створена в університеті, як правило, не може бути ефективно використана в іншому університеті без змін.

Отже, для вирішення існуючих проблем необхідно створити гнучку та легко адаптивну систему, засновану на нових принципах, використовуючи сучасні комп'ютерні технології. Ці функції також повинні бути реалізовані без зміни вихідного коду системи. Щоб охопити найбільш типові випадки, необхідно створити кілька типових алгоритмів, що реалізують планування. Ця система повинна мати можливість додавати та змінювати існуючу базу даних та призначений для користувача інтерфейс. Все це дозволило б встановити вимоги в кожному вищому навчальному закладі, що відповідають його критеріям, вибираючи і впроваджуючи відповідний алгоритм, щоб отримати бажаний графік.

1.4 Умови та завдання організації автоматизованої системи розкладу занять

Планування є однією з найбільш поширених завдань оптимізації освітнього процесу у навчальних закладах. Ефективність роботи вчителів, оволодіння навчальними матеріалами учнями та раціональне використання матеріальних ресурсів залежать від якості планування.

Автоматизація розкладу – традиційне завдання у навчальних системах управління, але в даний час немає єдиного загальноприйнятого способу його вирішення.

Всі підходи до планування засновані на евристичних методах, спрямованих на кого-небудь з професійним досвідом. Важко формалізувати ці методи, оскільки вони пов'язані з прийняттям рішення оператором, який

встановлює графік, який керується досвідом та інтуїцією. Часто сам працівник, який встановлює графік, не може відповісти на питання, чому він вибрав конкретний варіант інвестицій, а не інший прийнятний варіант. Однак, незважаючи на складність формалізації алгоритмів, можна розрізняти характеристики таких евристичних підходів в залежності від вимог до планування.

Звичайно, для кожної школи ці вимоги різні, тому що вони історично пов'язані з особливістю організації навчального процесу. Однак, навіть при всіх деталях, можна виділити загальні вимоги, що пред'являються до розкладу:

- мінімальна кількість занять у того, хто навчається в день;
- максимальна кількість занять у того, хто навчається в день;
- мінімізація вікон в учнів;
- максимальна кількість годин навчального навантаження на тиждень на кожного учня;
- облік тимчасових відстаней між корпусами при зміні корпусу навчаються;
- облік побажань викладачів;
- цикл занять з дисципліни не повинен закінчуватися лекційним заняттям, якщо є семінарські (практичні) заняття;
- цикл занять з дисципліни не повинен починатися з семінарського (практичного) заняття, якщо є лекційні заняття;
- до кожного лекційного заняття усі групи потоку повинні підходити, отримавши однакову кількість годин семінарських (практичних) занять;
- не проводити більше двох лекцій по одній і тій же дисципліні в день та не більше одного/двох семінарських занять по одній і тій же дисципліні в день;
- мінімальна кількість занять у викладача в день;
- максимальна кількість занять у викладача в день;
- мінімізація вікон у професорсько-викладацького складу;

- мінімізація переробок викладачів згідно штатного розкладу;

- мінімізація кількості одночасно проводяться однакових дисциплін у навчальному закладі в один і той же час;

- максимальне використання навчального фонду. Це може включати вимоги до максимального щільному розміщенню навчаються в класі в залежності від здатності класу, а також мінімізації простоїв;

- облік тимчасових відстаней між корпусами при зміні корпусу викладачем.

Завдання планування безпосередньо залежить від початкових умов. Ви можете групувати завдання планування відповідно до цих умов в певні групи:

- складання розкладу з апріорно відомої інформацією про розподіл груп між ППС;

- складання розкладу без урахування ППС, використовуючи лише навантаження кафедр;

- складання розкладу без урахування навантаження кафедр.

Давайте докладніше розглянемо характеристики кожної з перерахованих вище груп завдань. У завданнях з відомою інформацією про розподіл груп серед викладачів проблема полягає в тому, щоб враховувати побажання вчителя, контролювати руху під час зміни тіла, узгоджувати з програмою вчителя (маючи кілька класів одночасно). Для планування працівника потрібно дві основних години для складання: груповий розклад та розклад викладачів. Завдання стає особливо важким, якщо вчителі розподіляють навантаження між собою з точністю до групи і оператор розкладу не може змінити цей розподіл.

Тому одним з підходів до скорочення жорстких обмежень оператора є використання розподілу вчителів без груп: вчителі вказують тільки на викладацький склад, курс та кількість груп, які вони будуть проходити протягом семестру. Певна група призначається оператором, який планує. Це

дозволяє вам отримати ще одну додаткову ступінь свободи, яка зменшить кількість блокувань.

У завданнях, в яких використовується тільки завантаження відділів, оператор більше не повинен брати до уваги побажання викладача або перетин його курсів, і оператору не потрібно зберігати два графіка одночасно: група та вчитель. Однак опосередковано оператор зобов'язаний враховувати той факт, що відділ повинен мінімізувати кількість професорів, необхідних для забезпечення академічної навантаження.

Тому дуже важливо задовольнити вимогу максимально скоротити кількість однакових предметів, що знаходяться одночасно в навчальному закладі. Облік цієї вимоги призводить до появи такої влади, що вказує на кількість класів, які відділ може проводити в одній і тій же дисципліні в один і той же час.

Після планування департаменти повинні самі організувати вчителів. У завданнях планування без урахування відділів ступенів свободи набагато більше, ніж в інших; тому дуже логічний підхід полягає в тому, щоб залишити групу або потік. Однак в такому випадку при складанні розкладу не можна враховувати при складанні вимоги, пов'язані з викладачами. Облік цієї вимоги призводить до появи такої величини, як потужність кафедри, яка показує, скільки занять кафедра може вести по одній і тій же дисципліні одночасно. Після складання розкладу кафедрам доводиться розставляти викладачів самим. Це саме можна сказати і до проблеми між викладачами та лабораторними по підгрупах – на лабораторні заняття кафедра повинна надавати двох викладачів і дві навчальні аудиторії з необхідним обладнанням, однак, часто виходить так, що один і той же викладач веде одночасно у обох підгруп. Через це порушуються людино-години в табелі, а так само форма навчання, через що деякі навчаються можуть недоотримати інформацію [16].

У завданнях складання розкладу без урахування кафедр ступенів свободи набагато більше, ніж в інших, тому досить логічним підходом є рух

від групи або потоку. Однак в такому випадку при складанні розкладу не можна враховувати при складанні вимоги, пов'язані з викладачами. Саме тому важливо заздалегідь поставити в програмі пріоритети – від загально університетських викладачів до вузькопрофільних, які ведуть дисципліни у малої кількості груп.

Окремим завданням є розміщення класів в основних дисциплінах (якщо групи потоків не заповнені профілем), виборні дисципліни, іноземна мова, фізичне виховання та виборні курси. Проблема полягає головним чином в тому, що ми не завжди знаємо, до яких академічних груп вдаються одна або інша група для виконання вищевказаних дисциплін, що може привести до дублювання діяльності для студентів, які слідують одночасно курси з предметів обов'язкового навчання і вищезазначених предметів. Рішення полягає в тому, щоб призначати окремі дні для цих дисциплін або розміщувати ці класи першої або останньої парою.

Саме через те, що формалізувати всі вимоги, що пред'являються до розкладу, досить складно, а також складно врахувати унікальність бізнес-процесів в кожного навчального закладу, використання «жадібних» алгоритмів оптимізації без використання евристичних підходів призводить до результату, що не задовольняє кожен конкретний навчальний заклад. Тобто, не вийде розробити алгоритм, який би враховував всі особливості навчального закладу.

2 РОЗРОБКА І РЕАЛІЗАЦІЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ РОЗКЛАДУ ЗАНЯТЬ НАВЧАЛЬНОГО ЗАКЛАДУ

2.1 Вимоги до розроблюваної системи

Створення єдиної інформаційної системи є однією з цілей при проведенні автоматизації ВНЗ. Використання єдиного центру зберігання інформації (бази даних) з метою мінімізувати функції окремих користувачів – не менш важливе завдання при створення автоматизованих систем. Одна з головних вимог до системи є можливість зберігати інформації про студентів, кафедр та навчальних планів з подальшим їх використанням.

На рис. 2.1 представлено структуру програмного продукту.

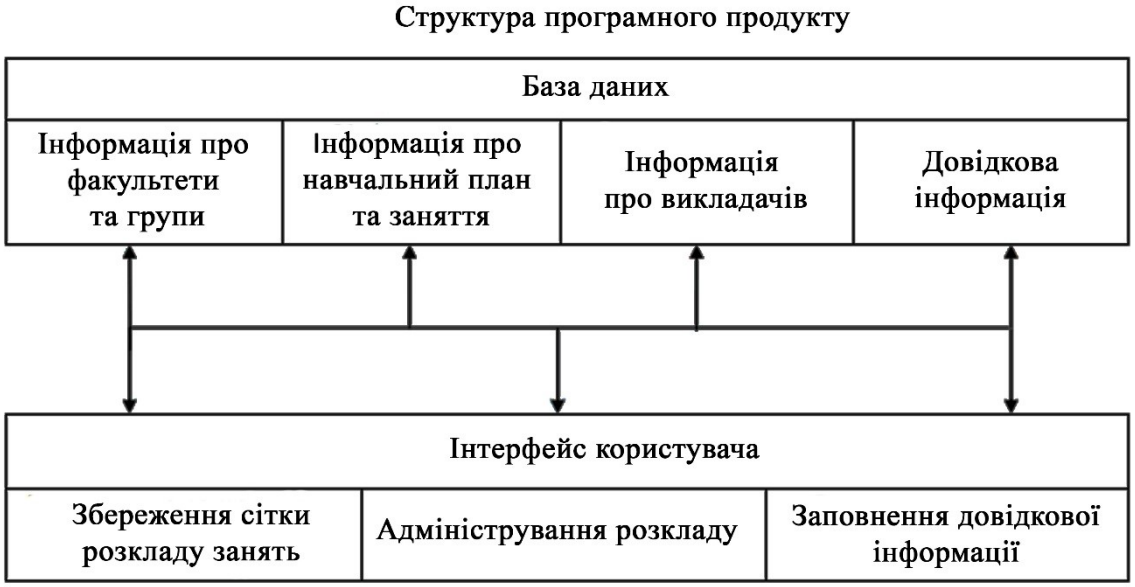

Рисунок 2.1 – Структура програмного продукту

Короткий опис призначень основних об'єктів:

- «База даних» – об'єкт, що містить інформацію про спеціальності, викладачів, навчальних планах, групах додаткову довідкову інформацію.

- «Інтерфейс» – об'єкт, що забезпечує високу інформативність виведеної на екран інформації, що організує зручність її виведення і обробки користувачем автоматизованої системи.

Також однією із зобов'язань, що подаються до розроблюваної системи, є створення та налаштування практичного для користувача інтерфейсу, який забезпечує легке сприйняття і опрацювання інформації, а також мінімізує власні операції. Створюється система при розрахунку навантаження і штатної чисельності кафедр університетів враховує прийняті норми часу в вузах України.

Звітні форми надають адміністратором детальну інформацію:

- навантаження спеціальностей і кафедр за рік;
- нормативному кількості штатних одиниць по кафедрам за певний проміжок часу.

Отже, методами досягнення поставлених завдань можна вважати:

- створення єдиної БД автоматизованої системи вищого навчального закладу;
- створення та налаштування призначеного для користувача інтерфейсу;
- генерація звітних форм.

2.2 Розгляд системи управління базами даних та інтерфейсу

Реалізація БД є критично важливим аспектом розробки програмного забезпечення, що вимагає зберігання і обробки інформації.

SQLite – це компактна вбудована реляційна база даних. Вихідний код бібліотеки переданий в суспільне надбання. Є чисто реляційної базою даних.

Слово «вбудовуваний» означає, що SQLite не використовує парадигму клієнт-сервер. Тобто движок SQLite не є окремо працюючим процесом, з яким взаємодіє програма, а надає бібліотеку, з якої програма компонується і движок стає складовою частиною програми [5].

Таким чином, в якості протоколу обміну використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому комп'ютері, на якому виповнюється програма. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції запису весь файла, який зберігає базу даних, блокується; ACID [1] – функції досягаються в тому числі за рахунок створення файлу журналу.

Додаток, що включає в себе SQLite, використовує її функціональність за допомогою простих викликів функцій. Оскільки функції викликаються в тому ж процесі, що працює додаток, виклики працюють швидше, ніж це було б у разі взаємодії між процесами.

Відхід від клієнт-серверної моделі зовсім не означає, що SQLite – це навчальна або урізана СУБД. Це означає лише специфіку її застосування в ролі вбудованого компонента. Існує безліч типів додатків, від «записників» до браузерів і операційних систем, які потребують невеликих локальних базах даних.

Оскільки SQLite працює в рамках іншої програми, під час запису файл бази даних блокується. Таким чином, записувати дані можна тільки послідовно. У той же час читати базу можуть відразу кілька процесів. Крім NULL, SQLite підтримує всього чотири типи даних – INTEGER, REAL, TEXT і BLOB. Останній тип – це двійкові дані. При цьому в стовпчик, оголошений одним типом, можуть записуватися дані будь-якого іншого. Якщо SQLite не може перетворити передані дані в заявлений для стовпця тип, то залишає їх як є.

Для забезпечення доступу до необхідної інформації була розроблена централізована база даних (БД) у середовищі SQLiteStudio.

БД є реляційною та приведена до третьої нормальної форми [2]. Умовно всю БД можна розділити на 4 логічні частини, відповідно до вхідної інформації, збереження якої необхідно забезпечити.

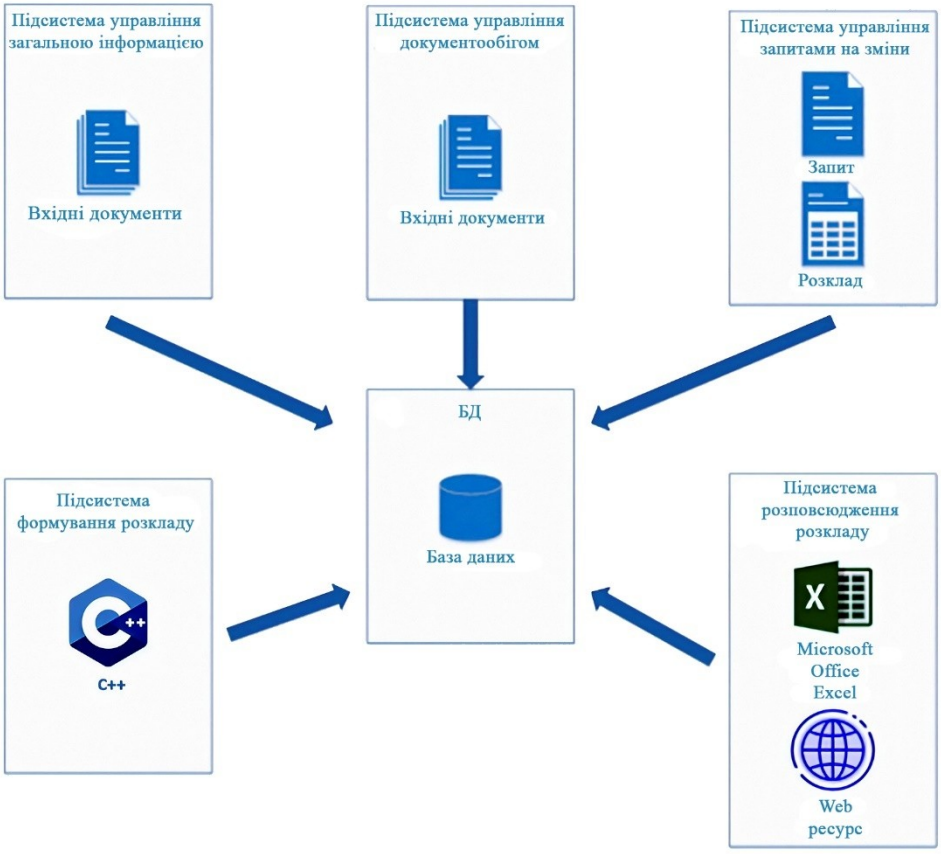

Рисунок 2.2 – Структура програмної системи

- Аудиторний ресурс містить дані про:
  - корпус;
  - аудиторію;
  - тип заняття, для якого призначена аудиторія;
  - спеціалізоване апаратне та програмне забезпечення аудиторії;
  - кафедру, факультет, яким аудиторія належить.
- Академічні групи та спеціальності охоплюють дані про:
  - групи;
  - курси;
  - спеціальності;
  - освітньо-професійні програми (ОПП);

- варіанти скорочень, якими можуть позначати студентів певної спеціальності, терміни навчання, курсу, форми навчання (ці дані у подальшому будуть використані для редагування вхідних файлів).

- Педагогічний склад включає дані про:

- розподіл навчальних занять (бажані аудиторії для викладання певної дисципліни);

- побажання щодо графіка роботи (особливо важливо для викладачів, що працюють за сумісництвом).

- Розклад.

- Програмні модулі.

На рис. 2.3 представлений склад програмної системи в частині забезпечення автоматизованого управління документами, необхідними для формування розкладу занять.

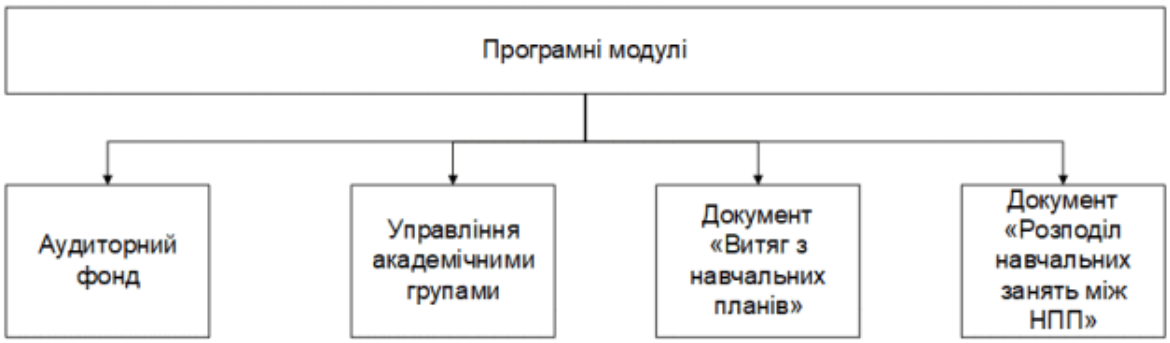

Рисунок 2.3 – Структура модулів підсистеми

Отже, реалізація БД є критично важливим аспектом розробки програмного забезпечення, що вимагає зберігання і обробки інформації.

Інтерфейс програми «Розклад» представлений з основного вікна, яке зображено на рисунку 2.4. Це є реалізація основних інструментів для складання розкладу.

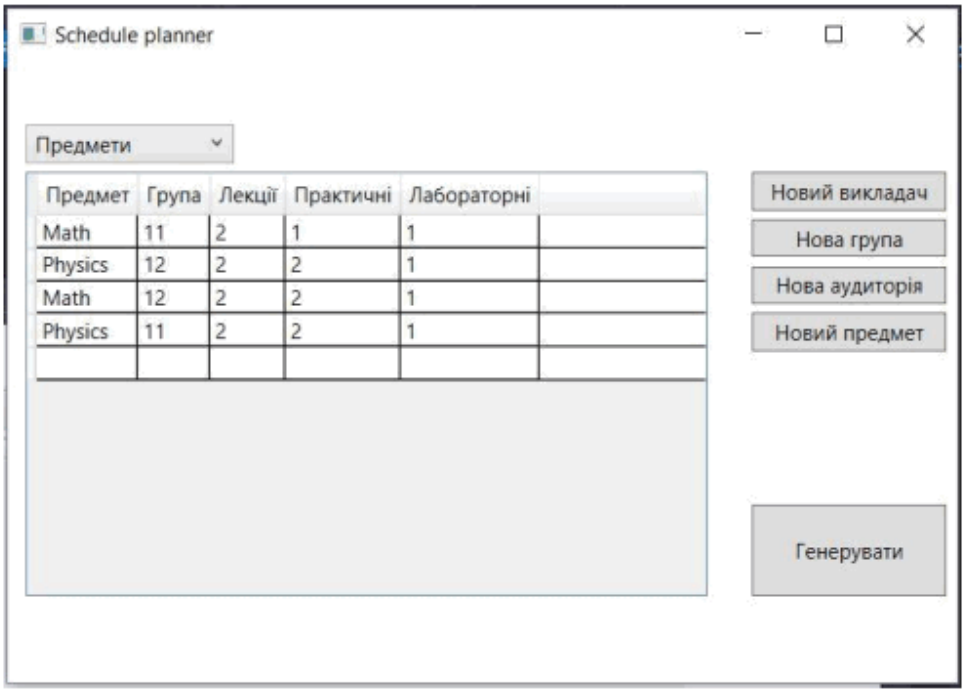

Рисунок 2.4 – Головне вікно

Ми можемо спостерігати кнопки адміністрування базою даних, де ми будемо додавати нових викладачів, студентів, аудиторії, дисципліни, а також є таблиця зі всіма даними, яку ми можемо переглянути перед формуванням розкладу.

Якщо ми перейдемо по кнопці, то з’являться нові вікна, де зрозуміло пояснено як користуватися ними. Дані діалогові вікна зображено на рис. 2.5.

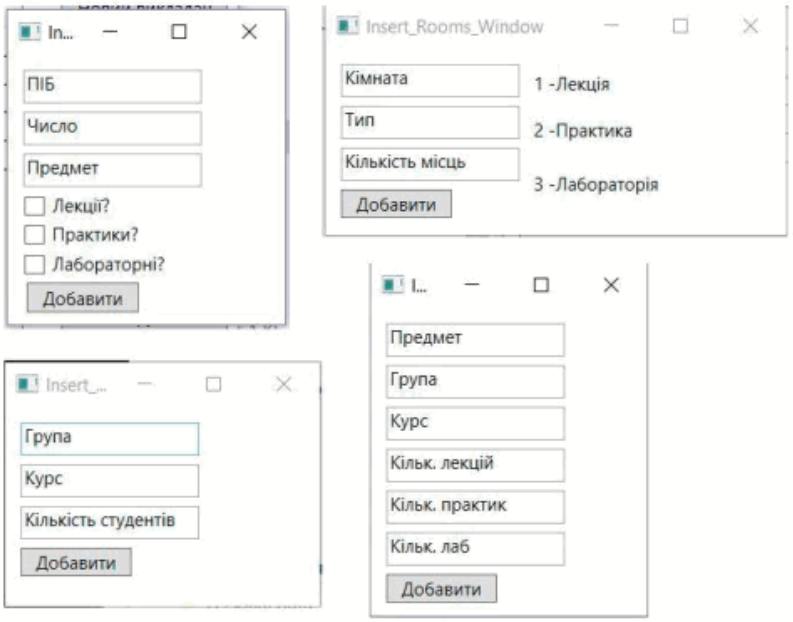

Рисунок 2.5 – Діалогові вікна введення даних

Після натиснення на кнопку «Генерувати» будуть запущені всі необхідні для обчислення алгоритми та створені генерації, які будуть відібрані як найкращі.

Наступний крок після вдалої генерації розкладу є збереження у XLSX файл на диску в папку з виконуваним файлом. На рисунку 2.6 наведено розклад в файлі Excel.

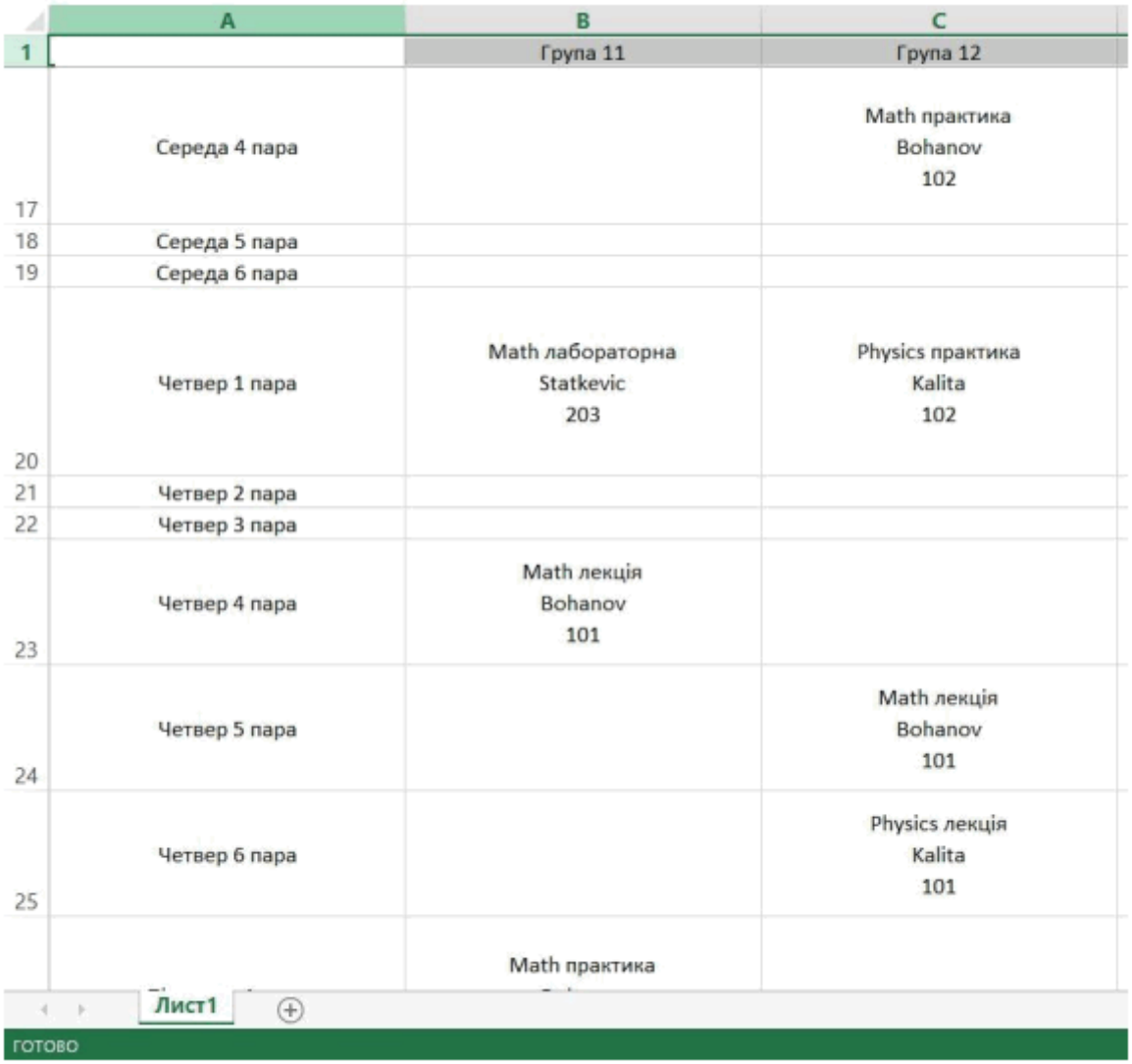

Рисунок 2.6 – Excel файл з розкладом

Отже, за допомогою формування розкладу виконуються всі жорсткі вимоги: відсутні дублікати, вибрані відповідні аудиторії, викладач та група студентів не знаходяться у двох місцях одночасно. З погляду нежорстких

вимог, розклад містить побажання викладачів та студентів та відсутність вікон у викладачів.

2.3 Алгоритми роботи з системами складання автоматизованого розкладу занять у ВНЗ

Генетичні алгоритми розглядаються як один з вдалих підходів до вирішення оптимізаційних задач. Спостерігається зацікавленість вчених у використанні принципу генетичного алгоритму, запропонованого Дж. Холландом (Мічиганський університет) в 1975 р [7]. Генетичні алгоритми мають велике число реалізацій з дослідженими методами оптимізації, поліпшення і налаштування, об'єднаними єдиною ідеєю вдосконалення алгоритму.

ГА є методом машинного навчання, які використовують механізми відбору в природі [8], які здійснюють випадковий і паралельний пошук рішень, які оптимізують заздалегідь визначену фітнес функцію [3].

У природі генетична інформація визначається четверичним кодом, заснованому на чотирьох нуклеотидах: аденін, цитозин, гуанін і тимін, з'єднаних разом в послідовність ДНК, яка лежить в основі генетичного коду [10]. При перенесенні цієї структури в інформатику, здається природним базувати всі кодування на двійковому коді, загальноприйнятому в комп'ютерних науках. Тобто використовуються хромосоми, які є двійковими рядками, що кодують вирішення поставлених завдань, і саме це є відмінною рисою генетичних алгоритмів.

ГА виконує глобальний пошук в просторі рішень, яке складається з векторів даних. Першим кроком алгоритму є ініціалізація простору рішень випадково згенерованими значеннями. На кожній ітерації, доступні рішення зазнають мутації або схрещуються з іншими рішеннями для того, щоб

створювати нові рішення. В кінці кожної ітерації, кожен індивідуум (рішення-кандидат) оцінюється з використанням заданої фітнес-функції. Таким чином, фітнес-функція – ключова частина кожного еволюційного алгоритму, і призначена для пошуку того рішення, яке є найкращим для даного завдання.

Після того, як кожна особина була оцінена, найменш придатні кандидати відхиляються, при цьому залишаються тільки кращі з доступних рішень в популяції. Цикл ітерацій повторюється до тих пір, поки зумовлений критерій зупинки не буде досягнутий. Як критерій зупинки може використовуватися або число ітерацій, або умова проходження певного порогу значення фітнес-функції, яке повинно бути досягнуто в просторі рішень [6].

При застосуванні генетичних алгоритмів до виконання реального завдання складення розкладу необхідно визначити насамперед внутрішній формат даних, який, з одного боку, повинен бути зручним для роботи алгоритму, а з іншого, враховувати всі особливості зовнішніх даних.

Вхідні дані доцільно зберігати у зовнішній базі даних в зручному форматі для наповнення та подання. Це дозволить підключити модуль складання розкладу до більш складної системи управління навчальним процесом університету. Однак в роботі алгоритму використання такого формату є неприйнятним з огляду на низькій швидкості доступу до таких даних і їх структурі, яка не відповідає внутрішнім структурам алгоритму. Тому перед початком роботи необхідно трансформувати зовнішні дані у внутрішній формат, а після закінчення – виконати зворотну трансформацію.

Ген – це неподільна одиниця, яка входить до складу хромосоми і кодує одну амінокислоту. Аналогічно в нашій презентації хромосоми ген кодує один кортеж розкладу – трійку (група, аудиторія, час). Група визначає сукупність деяких інших даних: навчальний курс, для якого сформована група (наприклад, Математика – l, Англійська мова – 2); тип заняття (лекція, лабораторна робота, практичне заняття). Кожен тип заняття визначає список

можливих викладачів і аудиторій для проведення заняття (ці дані надходять зовні): список студентів, що належать групі; викладач, який читає в даній групі.

Ці дані можна розділити на дві категорії:

- статичні дані, які не змінюються в процесі роботи алгоритму, такими є навчальний курс і тип заняття для кожної групи;

- динамічні дані, фактично входять в саме розклад, а саме список студентів в кожній групі, викладач цієї групи.

Список студентів може змінюватися (якщо це не обмежена ззовні). Наприклад, для чотирьох практичних груп з Математики – 1 неважливо, в якій групі буде той чи інший студент, тому в складання розкладу також входить генерація списків цих груп. Аналогічно не має значення, який викладач з деякого набору читатиме у тій чи іншій групи. Статичні дані будемо ідентифікувати кодом групи, а динамічні – номером її модифікації.

Отже, кожна група представлена в гені статичної та динамічної частинами. Статичну частину складають предмет, тип заняття, номер групи, а динамічну – поточна модифікація групи (лектор, список студентів). В межах одного розкладу для груп з одним кодом (сформованих для одного предмета і типу заняття), але з різними номерами модифікація групи буде однакова.

Генетичні алгоритми не вимагають виконання будь-яких правил при побудові початкового покоління рішень. Розклади спочатку можуть бути конфліктними, з порушеннями жорстких обмежень. Природно, це спрощує процес генерації початкового покоління, визначаючи для нього тільки один простий в реалізації метод – генерацію повністю випадкового розкладу. Але в подальшому, на другій фазі алгоритму, такий спосіб створення початкової популяції викликає ланцюжок проблем.

По-перше, необхідно визначати вагові коефіцієнти і для жорстких обмежень, причому ці коефіцієнти повинні бути набагато більшими, ніж ті, які визначаються для нежорстких обмежень. Але вибрати правильне співвідношення складно, до того ж це вимагає тривалих експериментів.

По-друге, значний ресурс буде витрачатися на розрахунки, зміну і підтримку заздалегідь неприпустимих розкладів. Існує ймовірність того, що ці неприпустимі рішення стануть безконфліктними протягом роботи алгоритму, але вона мізерно мала для повністю випадкових початкових рішень. Тому будемо дотримуватися безконфліктності розкладів на всіх етапах роботи алгоритму [18]. Це, природно, ускладнить алгоритм, але виграш буде незрівнянно більшим – не витрачатиметься зайве процесорний час, а також не потрібно вводити вагові коефіцієнти для жорстких обмежень, що істотно спростить налаштування алгоритму. На етапі початкової генерації безконфліктний, відносно жорстких обмежень, алгоритм побудови розкладу прийме наступний вигляд.

1) Визначення складу груп:

- з вхідних даних формується список студентів для кожного навчального курсу;
- вміст списків перемішується;
- з вхідних даних визначаються набори груп (лекційні, практичні і т.д.);
- для кожного набору обчислюється кількість груп і їх розмір;
- списки студентів заповнюються в групи послідовно.

2) Призначення педагогічного складу, який відповідає за групи:

- за вхідними даними визначається, які курси і види занять може вести викладач;
- випадковим чином формуються відповідності викладач та група.

3) Визначення часу проходження заняття. Групи розставляються випадковим чином в тимчасові слоти з урахуванням наступних вимог:

- немає жорстких конфліктів по викладачах;
- немає жорстких конфліктів по студентах.

4) Визначення аудиторії:

- на кожне отримане заняття визначається випадкова аудиторія зі списку доступних так, щоб не було конфлікту з зайнятості аудиторій.

Оскільки розкладів, згенерованих таким чином, має бути багато і всі вони різні, в алгоритмі присутні випадкові функції (див. пункт 1.2, 2.2, 3.1, 4.1). На підставі цього можна отримати велику кількість неоптимізованих, але безконфліктних (щодо жорстких обмежень) розкладів.

Далі зі згенерованих розкладів формуються хромосоми, що становлять початкову популяцію, яка в подальшому буде видозмінюватися. Зауважимо, що згідно з запропонованою структурою даних інформація про списки груп і викладачів задається в бітових полях. На практиці вони є двійковим поданням чисел, що означають номер модифікації групи. Цими номерами ставиться в однозначну відповідність певний список всіх підгруп, а також набір відповідностей викладач група.

Отже, після ініціалізації маємо структуру даних, що представляє набір хромосом і зручну для проведення над нею дій основної фази роботи алгоритму – емуляції еволюційних процесів. У цій фазі алгоритм передбачає ітеративне виконання наступних дій над хромосомами-розкладами:

- відбір хромосом для репродукції;
- застосування операторів зміни до хромосом-кандидатам;
- репродукція нового покоління;
- перевірка умов закінчення роботи алгоритму.

Ці кроки застосовуються до популяції рішень до тих пір, поки перевірка умов закінчення роботи алгоритму не приведе до позитивного результату.

2.4 Реалізація генетичного алгоритму

Розглянемо детально описані модифікації елементів генетичного алгоритму. Для всіх розроблених алгоритмів наводяться вихідні коди на мові C ++

Оператор селекції. При розробці програми складання розклад занять були протестовані різні оператори селекції. Найбільш ефективним виявився гібридний підхід, який поєднує в собі випадкову і рангову селекцію. Даний оператор працює наступним чином: всі особини популяції упорядковано (ранжуються) у порядку убування значення їх фітнес-функцій. Максимальним значенням фітнес-функції для кожної особини є 1, а мінімальним – 0. Потім датчиком випадкових чисел генерується певна кількість P, що лежить в інтервалі від 0 до N/10 (де N – розмір популяції). Після цього в циклі підсумовуються значення фітнес-функцій особин до тих пір, поки дана сума не перевищує P. Потім цикл зупиняється і передостання особина, яка брала участь у формуванні загальної суми, вибирається в якості одного з батьків. Такий підхід дозволяє забезпечити випадковий вибір батьків з 10% найбільш пристосованих особин. Привидемо приклад роботи реалізації на мові C ++:

```
int AlterSet(int index)
{
   int i=current_bestsize;
   if(bestFlags[index] || pool[index]-
>GetFitness()<=(pool[best_pool[current_bestsize-1]]->GetFitness()))
      return 0;
   while(true)
   {
      if(i<best_pool.size())
      {
         if(i==0){
```

```
            break;
        }
        if(pool[index]->GetFitness()<pool[best_pool[i-1]]->GetFitness())
            break;
        best_pool[i]=best_pool[i-1];
      }
    else bestFlags[best_pool[best_pool.size()-1]]=false;
    i--;
  }
  if(current_bestsize<best_pool.size())
    current_bestsize++;
  best_pool[i]=index;
  bestFlags[index]=true;
  return 0;
}
```

Оператор схрещування. Основним оператором, що впливає на якість рішень, отриманих за допомогою ГА, є оператор схрещування [14]. Здатність генетичних алгоритмів поєднувати різні рішення відрізняє їх від інших евристичних методів. Тому оператори кросовера повинні бути реалізовані таким чином, щоб добре пристосовані батьки могли бути інтегровані в одне більш пристосоване потомство. Найбільш часто в ГА застосовується простий одноточковий оператор кросовера. Він обмінює частини батьківських хромосом з кожного боку випадково обраної точки перетину. Наприклад, в двох 10-бітових хромосомах, якщо точка перетину знаходиться між п'ятим і шостим бітами, перше потомство успадковує від першого до п'ятого біта першого батька і від шостого до десятого біта другого з батьків. Друге потомство отримує те, що залишилося від половинки батьків. Цей процес

дуже схожий на генетичну рекомбінацію в реальному ДНК. Одноточковий кросовер ефективний в деяких простих випадках, але в реальних задачах він зазвичай генерує непридатні хромосоми [15]. Двоточковий ж кросовер обмінює гени навколо двох точок перетину, що дозволяє складати більшу кількість можливих комбінацій. Саме такий оператор і був використаний при вирішенні задачі складання розкладу. Нижче оператор схрещування в реалізації на мові C ++:

```
Solution* Crossover(Solution* parent1,Solution* parent2)
{
   vector<bool> c_point(all_classes.size());
   for( int i = 5; i > 0; i-- )
   {
      while( 1 )
      {
         int p = rand() % all_classes.size();
         if( !c_point[ p ] )
         {
            c_point[ p ] = true;
            break;
         }
      }
   }
   Solution* offspring=new Solution(no_of_rooms,no_of_classes);
   map<Class*, int>::const_iterator p1list_iter = (parent1->GetClasses()).begin();
   map<Class*, int>::const_iterator p2list_iter = (parent2->GetClasses()).begin();
   int first=rand()%2;
   for(int i=0;i<all_classes.size();i++)
   {
      if(c_point[i])
         first=1-first;
      if(first)
      {
         (offspring->GetClasses()).insert(pair<Class*, int>((*p1list_iter).first,
(*p1list_iter).second));
         for(int i=0;i<(*p1list_iter).first->GetDuration();i++)
         {
offspring.GetSlots().at((*p1list_iter).second+i).push_back((*p1list_iter).first);
         }
```

```
        }
        else
        {
           offspring->GetClasses().insert(pair<Class*, int>((*p2list_iter).first,
(*p2list_iter).second));
           for(int i=0;i<(*p2list_iter).first->GetDuration();i++)
           {
                        offspring-
>GetSlots().at((*p2list_iter).second+i).push_back((*p2list_iter).first);
           }
        }
        // iterating the parent lists
        p1list_iter++;
        p2list_iter++;
    }
    CalculateFitness(offspring);
    return offspring;
}
```

Оператор мутації. У той час як оператор кросовера намагається об'єднати вже розглянуті хороші рішення, оператор мутації вводить різноманіття в популяцію і дозволяє алгоритму генерувати нові рішення, які неможливо отримати лише з використанням одного тільки оператора кросовера. Кросовер може привести тільки до створення обмеженого набору хромосом на основі батьків. Застосовуючи невелика кількість мутацій до хромосомі, можна досліджувати нові області пошукового простору. Найбільш поширеним типом мутації з кодуванням бітових рядків є проста бітова мутація, в якій значення випадкового біта в хромосомі батьківської особини інвертується. При використанні ж цілочислового кодування, мутація може бути реалізована кількома способами.

Один метод, відомий як позиційна мутація, полягає у видаленні елемента з однієї позиції і вставці його в хромосому в іншу позицію, відповідно змінюючи і інші гени.

Інший підхід, заснований на порядок проходження генів, здійснює перестановку двох елементів в хромосомі. Альтернативний підхід, який вважається досить ефективним на практиці, заснований на скремблювання

генів в хромосомі. Цей метод створює хромосому нащадка, переставляючи гени батьківської хромосоми. Для кожного з операторів мутації необхідно визначати параметр, який регулює ступінь мутації. Для перших двох методів даний параметр визначає кількість замінних або переставляючих генів, а для третього підходу параметр мутації може визначати розмір області хромосоми-батька, в якій будуть здійснені перестановки. Зразок оператора з програми:

```
Solution* Mutation(Solution* ini)
{
   if( rand() % 100 > mutationProbability )
      return ini;
   int size = DAYS_NUM*DAY_HOURS*no_of_rooms;
   for( int i = mutationSize; i > 0; i-- )
   {
      int mutation_position = rand() % no_of_classes;
      int pos1 = 0;
      map<Class*, int>::iterator it = ini->GetClasses().begin();
      for( ; mutation_position > 0; it++, mutation_position-- );
      pos1 = ( *it ).second;
      Class* mutation_class = ( *it ).first;
      int rooms = config.GetNumberOfClassRooms();
      int duration_class = mutation_class->GetDuration();
      int day = rand() % DAYS_NUM;
      int room = rand() % rooms;
      int time = rand() % ( DAY_HOURS + 1 - duration_class );
      int pos2 = day * rooms * DAY_HOURS + room * DAY_HOURS + time;
```

```
      for( int i = duration_class - 1; i >= 0; i-- )
      {
         list<Class*>& slot_iterator = ini->GetSlots()[ pos1 + i ];
         for( list<Class*>::iterator it =slot_iterator.begin(); it != slot_iterator.end();
it++ )
         {
            if( *it == mutation_class )
            {
               slot_iterator.erase( it );
               break;
            }
         }
         ini->GetSlots().at( pos2 + i ).push_back( mutation_class );
      }
      ini->GetClasses()[ mutation_class ] = pos2;
   }
   CalculateFitness(ini);
   return ini;
}
```

Фітнес-функції. Побудова ефективної фітнес-функції є одним з самих складних і відповідальних етапів. Фітнес-функція повинна оцінювати задоволення м'яких і жорстких обмежень, а також штрафувати за будь-яке їх порушення. У розробленій програмі, фітнес-функція використовує хромосому в якості вхідних даних і повертає значення придатності в діапазоні від нуля (найменш пристосована особина) до одиниці (найбільш пристосована особина). Привидемо приклад фітнес-функція з програми:

```
void CalculateFitness(Solution* ch)
```

```
{
   int criteria_number=0;
   int daySize=DAY_HOURS*no_of_rooms;
   vector<bool>  parameter=ch->GetCriteria();
   int score=0;
   for(map<Class*, int>::const_iterator it = ch->GetClasses().begin(); it != ch->GetClasses().end(); ++it, criteria_number += 5 )
   {
      int position=(*it).second;
      int day = position / daySize;
      int time = position % daySize;
      int room = time / DAY_HOURS;
      time = time % DAY_HOURS;
      int duration = ( *it ).first->GetDuration();
      bool room_overlap=false;
      for(int i=0;i<duration;i++)
      {
         if(ch->GetSlots()[position+i].size()>1)
            room_overlap=true;
      }
      if(!room_overlap)
         score++;
      parameter[criteria_number+0]=!room_overlap;
      Class* curr=(*it).first;
      ClassRoom* r=config.GetClassRoomById(room+1);
```

```
parameter[criteria_number+1]=r->GetStrength()>=curr->GetStrength();
if(parameter[criteria_number+1])
   score++;
parameter[criteria_number+2]=!curr->LabClass() || ( curr->LabClass() && r->IsLab());
if(parameter[criteria_number+2])
   score++;
bool teacher_clash=false,student_clash=false;
int break_check=0;
for(int i=0,p=day*daySize+time;i<no_of_rooms;i++,p+=DAY_HOURS)
{
   for(int i=0;i<duration;i++)
   {
      list<Class*>& cl = ch->GetSlots()[ p + i ];
      for( list<Class*>::const_iterator it = cl.begin(); it != cl.end(); it++ )
      {
         if( curr != *it )
         {
            if( !teacher_clash && curr->TeacherClash( **it ) )
               teacher_clash = true;
            if( !student_clash && curr->StudentClash( **it ) )
               student_clash = true;
            if( teacher_clash && student_clash )
               break_check=1;
         }
```

```
            }
            if(break_check)
                break;
        }
        if(break_check)
                break;
    }
    if(!teacher_clash)
        score++;
    parameter[criteria_number+3]=!teacher_clash;
    if(!student_clash)
        score++;
    parameter[criteria_number+4]=!student_clash;
  }
  ch->GetFitness()=(float)score/(config.GetNumberOfSubjectClasses()*5);
}
```

Отже, ми розглянули на зразках з програми розроблений генетичний алгоритм оптимізації розкладу занять, визначено основні його етапи (селекція,фітнес-функція, схрещування, мутація і т.д.).

2.5 Виконання алгоритму автоматичної генерації розкладу

Алгоритм функції програми автоматичної генерації розкладу виконаний у вигляді декількох окремих класів, що навчаються імпортуванням даних з бази даних, розстановкою предметів і забезпечення виконання мінімальних умов оптимальності (відсутність накладень занять, як

у студентів, так і викладачів, забезпечення використання аудиторного фонду за цільовим призначенням і т.д.)

Алгоритм реалізований у вигляді 5 наступних етапів:

- загрузка з бази даних усіх наявних груп та інформації про їх навчальний план. Складання на їх основі двовимірного масиву «комірок» (що представляють собою специфічний тип даних, який має наступні параметри необхідні для складання розкладу: id групи, id предмета, id викладача, id аудиторії, тип необхідної аудиторії для заняття, назва предмета і тип проведеного заняття (теоретичне або практичне)), де кожен стовпець відповідає за групу, а строчка за відповідне одне заняття;

- розстановка на основі наявного двовимірного масиву аудиторій з урахуванням параметра комірок «тип необхідної аудиторії для заняття»;

- нормалізація масиву – додавання порожніх «комірок» або видалення наявних (з виведенням попередження про це користувачеві) для доведення кількості рядків в кожному стовпці до 60 (максимальна кількість занять проводяться за два тижні);

- розстановка занять;

- збереження згенерованого, у вигляді двовимірного масиву «комірок», розкладу в базу даних з можливістю подальшої демонстрації і редагування.

Даний алгоритм в своєму схематичному вигляді представлений на рис. 2.7.

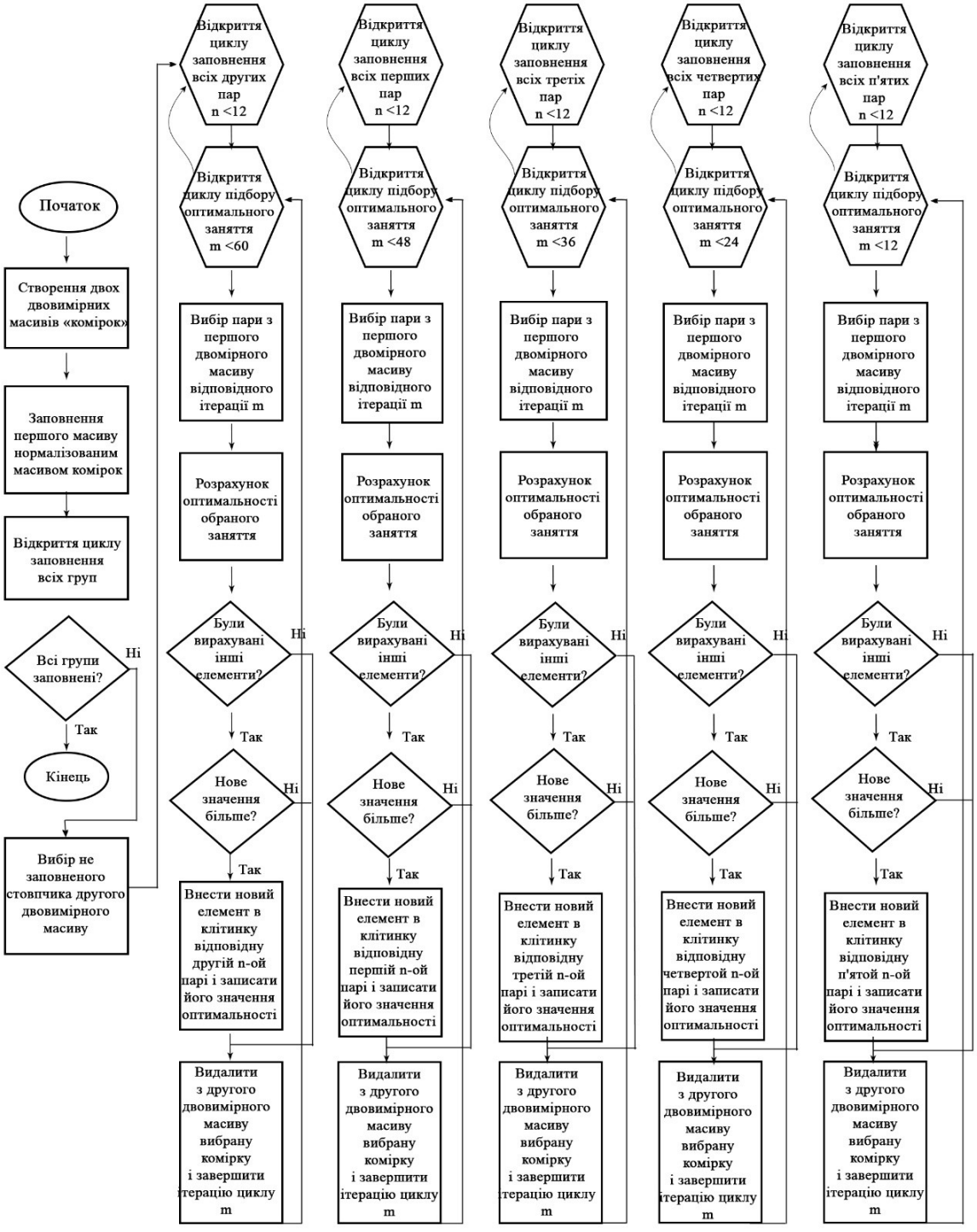

Рисунок 2.7 – Алгоритм розстановки занять

У зазначеному вище алгоритмі, для визначення оптимальності вибору конкретного заняття під різні пари (першої, другої, третьої, четвертої і п'ятої) використовуються несхожі один від одного критерії.

Важливо відзначити, що розстановка занять починається не з першої пари, а з другої, що дозволяє алгоритму органічно вписувати можливі переходи між корпусами (здійснювані під час великої перерви між другою і третьою парою) і уникнути виникнення «вікон» в розкладі, що виникають, пріоритетно на цих парах.

Розглянемо критерії оптимальності вибору для кожної окремо взятої пари.

Критерії оптимальності других пар :

- відповідно даного заняття аудиторія повинна бути вільна;
- відповідно даного заняття викладач повинен бути вільний;
- відповідно особистих вподобань викладачів.

Критерії оптимальності перших пар:

- наявність занять на другій парі;
- аудиторії другої пари повинені відповідати корпусу аудиторії першої пари;
- відповідно даному заняттю аудиторія повинна бути вільна;
- відповідний даному заняттю викладач повинен бути вільний;
- відповідно особистих вподобань викладачів.

Критерії оптимальності третьої пари:

- корпус аудиторії третьої пари переважно, але необов'язково повинен відповідати корпусу аудиторії другої пари;
- відповідно даного заняття аудиторія повинна бути вільна;
- відповідно даного заняття викладач повинен бути вільний;
- відповідно особистих вподобань викладачів.

Критерії оптимальності четвертої пари:

- наявність занять на третій парі;
- корпус аудиторії четвертої пари повинен відповідати корпусу аудиторії третьої пари;
- відповідно даного заняття аудиторія повинна бути вільна ;
- відповідно даного заняття викладач повинен бути вільний;

- відповідно особистих вподобань викладачів.

Критерії оптимальності п'ятої пари:

- корпус аудиторії п'ятої пари повинен відповідати корпусу аудиторії третьої пари;

- відповідно даного заняття аудиторія повинна бути вільна;

- відповідно даного заняття викладач повинен бути вільний;

- відповідно особистих вподобань викладачів.

Для п'ятої пари переважно виставлення порожнього вікна заняття, якщо таке неможливо відповідно до існуючого навчального плана.

Перехід між корпусами передбачається виконувати виключно в проміжку між другою і третьою парою, проте кількість переходів, відповідно до критеріїв оптимальності третьої, четвертої та п'ятої пари має мінімізуватися.

ВИСНОВКИ

Звернення до сучасної автоматизованої системи – це логічний та важливий етап у формуванні вищої освіти в Україні. Університет повинен прагнути формувати такі умови, в яких інноваційні підходи будуть застосовуватися і студентами, і викладачами, і керівництвом. Автоматизація процесу ведення розкладу занять і багатофунціональність діючої системи, безсумнівно, мають переваги при її використанні в університеті, покращуючи при цьому діяльність фахівців і викладачів і отже підвищуючи якість освітніх послуг в Національном університеті «Запорізька Політехніка».

В результаті проведеної роботи була розроблена автоматизована система розкладу занять навчального закладу з метою її впровадження в навчальний процес університету. Дана система має зручний призначений для користувача інтерфейс, що дозволяє легко освоїти роботу в програмі, гнучкість ж програмного коду в разі необхідності дозволить задовольнити зростаючі вимоги до системи. Розроблену автоматизовану систему можна інтегрувати в єдину інформаційну систему ВНЗ, що, безсумнівно, позитивно позначиться на роботі користувачів при аналізі структури навантаження, при плануванні структурної доопрацювання і уніфікації наявних навчальних планів.

Розроблена система дозволить підвищити швидкість обробки інформації, скоротить терміни формування звітів і заощадить час роботи користувачів. Автоматизація процесу адміністрування розкладу занять і гнучкість розробленої автоматизованої системи дають переваги при її використанні в системі освіти, покращуючи при цьому діяльність персоналу, а разом з тим і підвищуючи якість наданого ВНЗ освіти.

Модульна реалізація розробленої системи автоматизованого складання розкладу в структурі загальної автоматизованої системи ведення документообігу забезпечує можливість спільного використання баз даних і

впровадження загальної політики захисту інформаційного забезпечення системи.

На основі сформульованого генетичного алгоритму було реалізовано програмний продукт, що формує розклад занять з допомогою даного алгоритму, на мові програмування C++.

Було сформульовано критерій якості розкладу (цільова функція, що залежить обернено пропорційно від об’єму порушень м’яких обмежень, що накладаються на розклад, значення якої потрібно максимізувати для отримання оптимального розкладу).

На закінчення слід сказати, що рішення, запропоновані для усунення проблем автоматизації навчального розкладу, дозволять розробити систему, спрямовану на специфіку роботи ВНЗ, залучити співробітників до участі в розробці і в подальшій роботі з системою. Відтак буде зменшуватися навантаження на співробітників шляхом зменшення обсягу створення і реєстрації документів, з'явиться можливість відстежувати весь життєвий цикл документів, і тим самим підвищиться ефективність і продуктивність праці при роботі з документами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Астахова І. Ф. Створення розкладу навчальних занять на основі генетичного алгоритму / Астахова І. Ф., Фірас А. М. // Вісник воронежского державного університету, серія: «Системний аналіз и інформаційні технології». – 2013. – № 2. – С 93-99.

2. Бабкіна Т. С. Задача складання розкладу: рішення на основі багатоагентного підходу / Бабкіна Т. С. // Бізнес-інформатика. – 2008. – №1. – С.23-28.

3. Батищев П. С. Опыт использования информационных технологий при составлении расписания учебных занятий. Текст. // Среднее профессиональное образование. – 2003. – №11. С. – 38.

4. Бевз С. В. Розробка автоматизованої системи формування розкладу
магістратури / Бевз С. В., Войтко В. В., Бурбело С. М., Шоботенко А. М. // Інформаційні технології та комп'ютерна техніка – 2009 - №4 – С. 30-65.

5. Бевз С. В. Автоматизація процесу формування розкладу сесії. / Бевз С. В., Войтко В. В., Бурбело С. М., Куба Т. О., Сухоносов О. О.// Принципові концепції та структурування різних рівнів освіти з оптикоелектронних інформаційно-енергетичних технологій – 2009 - №4 – С. 25-36.

6. Безгинов А. Н. Обзор существующих методов составления расписания [Текст] / А. Н. Безгинов, С. Ю. Трегубов // Информационные технологии и программирование: межвузовский сборник статей. Выпуск 2(14). – М.: МГИУ, 2005.

7. Верьовкін В. И. Автоматизоване створення розкладів навчальних занять вишу с урахуванням складності дисциплін і втомленості студентів / Верьовкін В. И., Ісмагілова О. М., Атавін Т. А. // Доповіді ТУСУР. – 2009. – №1 (19), частина 1. – С. 221-225.

8. Воробович О. Н. Алгоритм формирования расписания занятий студенческих групп в высшем учебном заведении [Текст]. // Материалы межвуз. науч. конф. / под ред. Е. А. Вейсова, Ю. А. Шитова, КГТУ. – Красноярск, 2013. – С. 29 – 35.

9. Галузин К. С. Методика составления оптимального учебного расписания с учетом предпочтений Текст. / К. С. Галузин, В. Ю. Столбов // Теоретические и прикладные аспекты информационных технологий: Сб.науч.тр. /ГосНИИУМС. Вып. 53. – Пермь, 2014. – С. 43-50.

10. Голуб Б. Л., Ящук Д. Ю. Навчальний посібник до вивчення дисципліни «Основи організації баз даних» для студентів, що навчаються за спеціальностями галузі 12 «Інформаційні технології». К: ТОВ «ЦП КОМПРИНТ», 2017. 151 с.

11. Деканова М. В. Математична модель и алгоритм побудови розкладу
навчальних занятий університету / Деканова М. В. // Вісник Полоцкого державного університету. Серія С. – 2013. – №12. – С. 24-33.

12. Діаграма Ганта [Електронний ресурс]. – Режим доступу: http://uk.wikipedia.org/wiki/Діаграма_Ганта. – Назва з екрану.

13. Конвей, Р. В. Теория расписаний. Текст. / Р. В. Конвей, В. Л. Максвелл, Л. В.Миллер М.: Наука, 2015. – 360 с.

14. Логоша, Б. А. Комплекс моделей и методов оптимизации расписания занятий в вузе / Б. А. Логоша, А. В. Петропавловская // Экономика и математические методы. – 2013. – Т. 29., №4.

15. Мейер Д. Теория реляционных баз данных [Текст]. М.: Мир, 2014.

16. Мокін В. Б., Бевз С. В., Бурбело С. М. Розробка та впровадження систем документообігу і менеджменту навчального процесу магістерської підготовки // Оптико-електронні інформаційно-енергетичні технології. — 2006.— № 2. — С. 5–12.

17. Мулява І. Я. Вирішення задачі автоматизованого формування розкладу навчального закладу за допомогою генетичних алгоритмів // Міжнародний науковий журнал "Інтернаука". — 2018. — №9.

18. Мулява І. Я. Програмна модель формування розкладу навчальних занять // Наука онлайн: Міжнародний електронний науковий журнал — 2018. — №5.

19. Рубальская, О. Н. Автоматизированные системы составления учебных расписаний / О. Н. Рубальская, Г. Б. Рубальский // Новые информационные технологии в образовании: аналитические обзоры по основным направлениям развития высшего образования. – М., 2011.

20. Симоненко В. П., Симоненко С. И. Метод пошагового конструирования для составления расписания занятий в учебных заведениях. Системні дослідження та інформаційні технології, 2008, № 4. С. 77–85.

21. Снитюк В. Є. Аспекти формування цільової функції в задачі складання розкладу занять у вищих навчальних закладах на основі суб'єктивних переваг / Снитюк В. Є., Сіпко Є. Н. // Автоматика. Автоматизація. Електротехнічні комплекси і системи - 2013 – №2 – С.98-104.

22. Снитюк В. Є. Про особливості формування цільової функції та обмежень в задачі складання розкладу занять / Снитюк В. Є., Сіпко Є. Н. // Математичні машини і системи – 2014 - №3 – С. 67-76.

23. Управління проектами [Електронний ресурс]. – Режим доступу: http://uk.wikipedia.org/wiki/Управління_проектами. – Назва з екрану.

24. Юхимчук С. В., Бевз С. В., Бурбело С. М., Дмитришин О. В., Чернова І. О. Розробка локальної автоматизованої системи розподілу навантаження в процесі ведення документообігу // Вісник Вінницького політехнічного інституту. – Вінниця. – № 1. – 2009. – С. 107-111.

25. Юхимчук С. В., Бевз С. В., Бурбело С. М., Крещенецька М. В., Богатчук С. М. Розробка автоматизованої системи формування розподілу навантаження дисциплін магістерської підготовки // Наукові праці ВНТУ. – 2008. – № 4. – Режим доступу до журн.:

http://www.nbuv.gov.ua/e-journals/VNTU/2008-4/2008- 4.files/uk/ 08svymts_uk.pdf.

26. Fonseca, C.M.; Fleming, P.J. Genetic algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In Proceedings of the Fifth International Conference on Genetic Algorithms, UrbanaChampaign, IL, USA, 17–22 July 1993; Volume 93, pp. 416–423.