НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (повна назва кафедри, циклової комісії)

**КУРСОВА РОБОТА**

з \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студента (ки) \_\_\_\_\_ курсу \_\_\_\_\_\_ групи

Спеціальності 121 «Інженерія програмного забезпечення»

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії

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

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

Київ- 2024рік

**ЗМІСТ**

ВСТУП.................................................................................................................................. 4

1 ПОРЯДОК ВИКОНАННЯ КУРСОВОЇ РОБОТИ......................................................... 5

1.1 Об'єктно-орієнтований аналіз предметної області ................................................. 5

1.2 Об'єктно-орієнтоване проектування ........................................................................ 6

1.3 Об'єктно-орієнтоване програмування ...................................................................... 6

1.4 Тестування програми ................................................................................................. 7

1.5 Оформлення пояснювальної записки....................................................................... 8

1.6 Захист курсової роботи.............................................................................................. 8

2 ЗМІСТ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ ....................................................................... 9

3 ПРАВИЛА ОФОРМЛЕННЯ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ ................................ 14

3.1 Загальні вимоги ........................................................................................................ 14

3.2 Нумерація.................................................................................................................. 15

3.3 Оформлення цитат і переліку посилань................................................................. 16

3.4 Оформлення додатків .............................................................................................. 17

4 ПОРЯДОК ВИКОНАННЯ ТА ЗАХИСТУ КУРСОВОЇ РОБОТИ............................. 18

4.1 Хід виконання та захисту курсової роботи ........................................................... 18

4.2 Критерії оцінювання курсової роботи ................................................................... 20

5 ТЕМИ ІНДИВІДУАЛЬНИХ ЗАВДАНЬ НА КУРСОВУ РОБОТУ........................... 25

5.1 Варіанти завдань ............................................Ошибка! Закладка не определена.

5.1.1 Тематика базових індивідуальних завданьОшибка! Закладка не

определена.

5.1.2 Тематика додаткових індивідуальних завданьОшибка! Закладка не

определена.

СПИСОК ЛІТЕРАТУРИ................................................................................................... 26

ДОДАТОК А ТЕХНІЧНЕ ЗАВДАННЯ.......................................................................... 27

ДОДАТОК Б ТИТУЛЬНА СТОРІНКА........................................................................... 29

ДОДАТОК В ЛИСТ ЗАВДАННЯ.................................................................................... 30

ДОДАТОК Г ТЕКСТИ ПРОГРАМНОГО КОДУ........................................................... 32

**Вступ**

Виконання курсової роботи з дисципліни "Основи програмування" є невід’ємною частиною навчального процесу студентів, які навчаються за спеціальністю 121 "Інженерія програмного забезпечення" на бакалаврському рівні. Робота має на меті не лише поглиблення теоретичних знань, отриманих під час вивчення курсу, але й їх застосування для розв’язання конкретних фахових задач, що сприяє формуванню висококваліфікованих фахівців у галузі програмування.

Тема моєї курсової роботи — "ОСББ. Управління будинком. Електронний довідник даних про мешканців та орендарів багатоквартирного будинку". Проект передбачає розробку програмного забезпечення, що дозволить автоматизувати процес збору, зберігання та обробки інформації про мешканців та орендарів: їхні особисті дані, номер квартири, параметри проживання, вартість оренди та комунальних послуг.

Основними завданнями, які стоять переді мною під час виконання курсової роботи, є:

* Розробка об'єктно-орієнтованої моделі даних;
* Програмування функціоналу для розрахунку квартплати з урахуванням різних факторів, таких як площа квартири, кількість мешканців, споживання води та електроенергії;
* Створення інтуїтивно зрозумілого графічного інтерфейсу користувача;
* Реалізація механізмів збору та аналізу даних для розрахунку доходів і витрат ОСББ.

Курсова робота вимагає застосування сучасних технологій розробки програмного забезпечення і використання навичок, набутих в процесі навчання, як у теоретичному, так і в практичному аспектах. Завершення проекту потребує від студента здатності до самостійної роботи, аналітичного мислення та вміння розв’язувати комплексні задачі. За результатами виконаної роботи будуть сформовані висновки, які покажуть ефективність запропонованого рішення.

Цей проект відображає практичне застосування набутих знань і спрямований на поліпшення управління житловими комплексами через використання інноваційних програмних рішень, що є актуальним і сучасним викликом в галузі інженерії програмного забезпечення.

**1 ПОРЯДОК ВИКОНАННЯ КУРСОВОЇ РОБОТИ**

Основні етапи виконання курсової роботи:

− об'єктно-орієнтований аналіз предметної області задачі;

− об'єктно-орієнтоване проектування архітектури програмної системи;

− об'єктно-орієнтоване програмування;

− тестування програми;

− оформлення пояснювальної записки;

− захист курсової роботи.

* 1. **Об'єктно-орієнтований аналіз предметної області**

А) Предметна область цього програмного забезпечення присвячена управлінню багатоквартирними будинками та обліку інформації про мешканців, орендарів, комунальні витрати та витрати на ремонт.

Терміни та поняття:

* Багатоквартирний будинок
* Квартира
* Кімната
* Орендар (Tenant)
* Власник квартири (LandLord)
* Витрати на комунальні послуги (UtilityExpenses)
* Витрати на ремонт (RepairExpense)
* Оренда
* Дохід від оренди

Суб'єкти:

* Власники квартир (LandLord)
* Орендарі (Tenant)
* Об'єднання співвласників багатоквартирного будинку (ОСББ)

Об'єкти:

* Багатоквартирний будинок (Building)
* Квартира (Apartment)
* Кімната (Room)

Способи взаємодії суб'єктів:

* Власники квартир здають їх в оренду орендарям
* Орендарі сплачують орендну плату власникам квартир
* ОСББ збирає інформацію про мешканців, орендарів, комунальні витрати та витрати на ремонт

Способи використання об'єктів:

* Багатоквартирний будинок є сукупністю квартир
* Квартира складається з кімнат та використовується для проживання
* Кімната є частиною квартири та використовується для різних цілей (спальня, вітальня тощо)

Закономірності:

* Вартість оренди квартири залежить від її площі, кількості кімнат та місцезнаходження
* Комунальні витрати залежать від кількості мешканців, споживання ресурсів (електрика, вода, опалення тощо)
* Витрати на ремонт виникають після закінчення терміну оренди та залежать від ступеня зносу квартири
* Дохід власника квартири формується від орендної плати за вирахуванням витрат на комунальні послуги та ремонт

Отже, ця предметна область охоплює управління багатоквартирними будинками, облік інформації про мешканців, орендарів, комунальні витрати та витрати на ремонт, а також взаємодію між власниками квартир, орендарями та об'єднанням співвласників будинку.

Б) Словник предметної області включатиме такі елементи:

Іменники:

1. Будинок (Building)
2. Квартира (Apartment)
3. Кімната (Room)
4. Орендар (Tenant)
5. Власник (LandLord)
6. Оренда (Rent)
7. Витрати (Expenses)
8. Комунальні послуги (Utilities)
9. Ремонт (Repair)
10. Дохід (Income)
11. Об'єднання співвласників (Owners' Association)
12. Платіж (Payment)
13. Площа (Area)
14. Поверх (Floor)
15. Контракт (Contract)
16. Термін (Term)

Дієслова, пов'язані з іменниками:

1. Будинок: будувати, володіти, управляти, обслуговувати.
2. Квартира: орендувати, винаймати, проживати, ремонтувати, обладнати.
3. Кімната: використовувати, меблювати, прибирати.
4. Орендар: орендувати, винаймати, сплачувати, проживати, звітувати.
5. Власник: володіти, здавати в оренду, отримувати дохід, ремонтувати.
6. Оренда: укладати, сплачувати, поновлювати, припиняти.
7. Витрати: нараховувати, сплачувати, відшкодовувати.
8. Комунальні послуги: споживати, оплачувати, розраховувати.
9. Ремонт: проводити, оцінювати, фінансувати.
10. Дохід: обчислювати, отримувати, розподіляти.
11. Об'єднання співвласників: управляти, координувати, представляти.
12. Платіж: здійснювати, обробляти, відстежувати.
13. Площа: обчислювати, вимірювати, оцінювати.
14. Поверх: розташовувати, будувати.
15. Контракт: укладати, переглядати, поновлювати.
16. Термін: встановлювати, продовжувати, закінчувати.

Цей словник охоплює основні терміни та поняття, пов'язані з управлінням багатоквартирними будинками, орендою квартир, комунальними витратами, ремонтами та взаємодією між учасниками процесу.

В) Функціональні вимоги до розроблюваної системи управління багатоквартирними будинками:

1. Облік інформації про будинки:

- Можливість додавання, видалення та редагування інформації про будинки (назва, кількість поверхів, адреса тощо).

- Перегляд списку всіх будинків у системі.

2. Облік інформації про квартири:

- Можливість додавання, видалення та редагування інформації про квартири (номер, кількість кімнат, площа, поверх, вартість за квадратний метр).

- Прив'язка квартири до певного будинку.

- Перегляд списку всіх квартир у певному будинку.

3. Облік інформації про власників квартир (домовласників):

- Можливість додавання, видалення та редагування інформації про власників (ім'я, контактна інформація, квартири у власності).

- Перегляд списку всіх власників квартир у системі.

- Перегляд інформації про квартири, якими володіє певний власник.

4. Облік інформації про орендарів:

- Можливість додавання, видалення та редагування інформації про орендарів (ім'я, контактна інформація, орендована квартира).

- Перегляд списку всіх орендарів у системі.

- Перегляд інформації про квартиру, яку орендує певний орендар.

5. Управління орендою квартир:

- Можливість встановлення терміну оренди для квартири.

- Розрахунок вартості оренди квартири на основі площі, кількості кімнат та вартості за квадратний метр.

- Облік доходу від оренди для власника квартири.

6. Облік комунальних витрат:

- Можливість додавання та редагування інформації про комунальні витрати для певної квартири (вартість електроенергії, води, опалення тощо).

- Перегляд історії комунальних витрат для квартири.

7. Облік витрат на ремонт:

- Можливість додавання та редагування інформації про витрати на ремонт для певної квартири (опис ремонту, вартість).

- Перегляд історії витрат на ремонт для квартири.

8. Звітність та аналітика:

- Генерація звітів про дохід від оренди для власників квартир.

- Генерація звітів про комунальні витрати та витрати на ремонт для квартир.

- Аналіз та візуалізація даних (наприклад, графіки доходів або витрат за певний період).

9. Обробка виключних ситуацій:

- Належна обробка виключних ситуацій, таких як некоректний введений користувачем вхідний дані, порушення обмежень даних тощо.

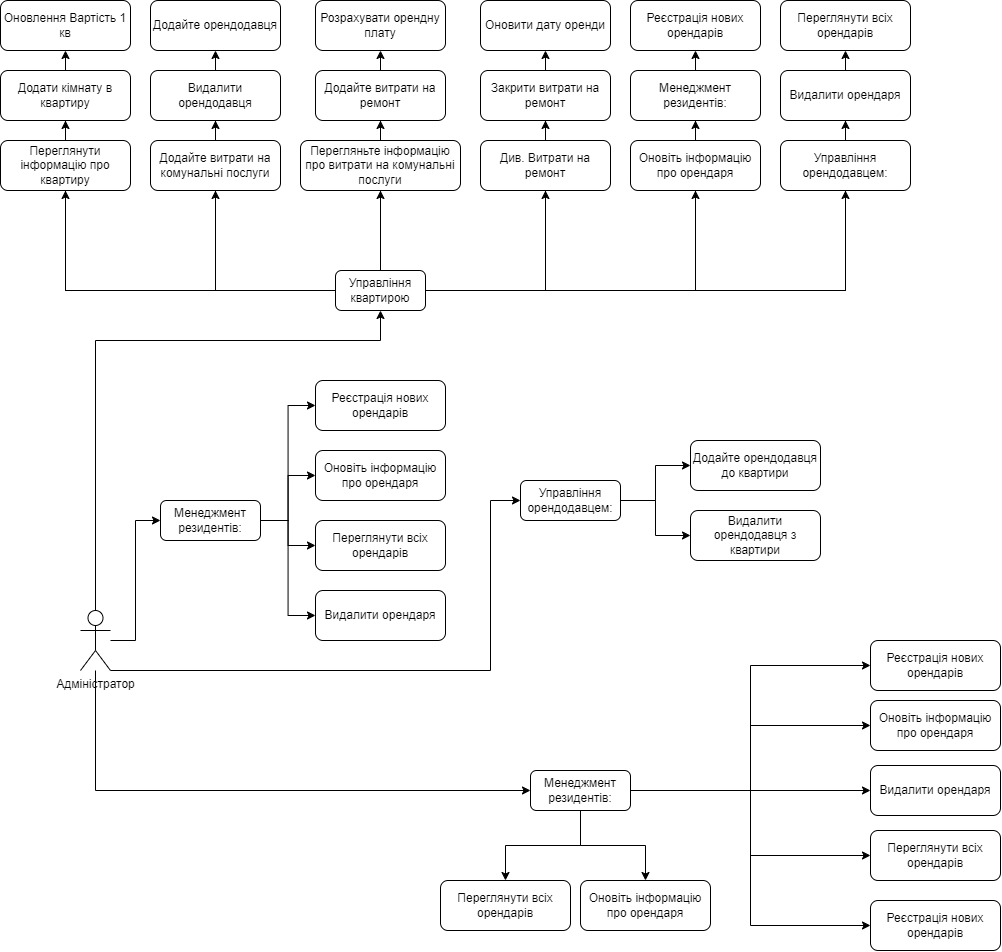
Ці функціональні вимоги охоплюють основні аспекти управління багатоквартирними будинками, обліку інформації про власників, орендарів, квартири, оренду, комунальні витрати та ремонти, а також генерацію звітності та аналітики.

Приблизний сценарій поведінки об'єктів в системі управління багатоквартирними будинками:

1. Створення об'єктів класу Building (будинки):
   * Адміністратор системи створює нові об'єкти класу Building, вказуючи назву будинку та кількість поверхів.
2. Додавання квартир до будинку:
   * Для кожного будинку адміністратор створює об'єкти класу Apartment, вказуючи номер квартири, кількість кімнат, поверх, вартість за квадратний метр та прив'язуючи її до відповідного об'єкту Building.
3. Створення об'єктів класу LandLord (власників квартир):
   * Адміністратор створює об'єкти класу LandLord для кожного власника квартири, вказуючи їх контактні дані.
   * Для кожного власника квартири встановлюється зв'язок з відповідними об'єктами класу Apartment, які він володіє.
4. Здача квартир в оренду:
   * Власник квартири (об'єкт LandLord) встановлює терміни оренди для своїх квартир (об'єктів Apartment) та визначає вартість оренди.
   * Система розраховує дохід власника квартири від оренди на основі вартості оренди та терміну.
5. Створення об'єктів класу Tenant (орендарів):
   * Адміністратор або власник квартири створює об'єкти класу Tenant для кожного орендаря, вказуючи їх контактні дані.
   * Для кожного орендаря встановлюється зв'язок з об'єктом класу Apartment, який він орендує.
6. Облік комунальних витрат:
   * Власник квартири або адміністратор створює об'єкти класу UtilityExpenses для кожної орендованої квартири, вказуючи витрати на електроенергію, воду, опалення тощо за певний період (місяць).
   * Об'єкти UtilityExpenses зберігаються у відповідних об'єктах класу Apartment.
7. Облік витрат на ремонт:
   * Після закінчення терміну оренди власник квартири створює об'єкти класу RepairExpense для своїх квартир, вказуючи опис ремонту та вартість.
   * Об'єкти RepairExpense зберігаються у відповідних об'єктах класу Apartment.
8. Генерація звітів та аналітики:
   * Адміністратор або власник квартири може запросити звіти про дохід від оренди для певного власника квартири або для всіх власників.
   * Адміністратор або власник квартири може запросити звіти про комунальні витрати та витрати на ремонт для певної квартири або для всіх квартир.
   * Система може генерувати графіки та візуалізації на основі даних про доходи, витрати тощо.
9. Обробка виключних ситуацій:
   * У випадку некоректного введення даних користувачем або порушення обмежень даних, система повинна належним чином обробляти ці виключні ситуації та повідомляти користувача про помилки.

Цей сценарій демонструє, як об'єкти різних класів (Building, Apartment, LandLord, Tenant, UtilityExpenses, RepairExpense) взаємодіють між собою для виконання різних функцій системи управління багатоквартирними будинками, таких як облік інформації про будинки, квартири, власників та орендарів, управління орендою, облік комунальних витрат та ремонтів, а також генерація звітності та аналітики.

Діаграмма прецедентів можна побачити на - **рис 1.1**



**Рис 1.1**

* 1. **Об'єктно-орієнтоване проектування**

1.2 Об'єктно-орієнтоване проектування

На цьому етапі розробки курсового проекту описується архітектура системи, яка базується на принципах об'єктно-орієнтованого програмування (ООП).

1.2.1 Обробка словника предметної області

Першим кроком в ООП-проектуванні є аналіз предметної області, що включає виявлення ключових понять та їх характеристик. Це робиться шляхом обробки словника предметної області, який містить перелік іменників та дієслів.

Іменники слугують основою для виділення класів системи. Класи представляють собою сутності предметної області, які мають власні властивості та методи.

Дієслова використовуються для визначення методів класів. Методи описують дії, які може виконувати об'єкт класу.

На основі аналізу словника предметної області формується діаграма класів, яка відображає зв'язки між класами та їхні характеристики.

1.2.2 Визначення характеристик класів

Наступним кроком є визначення характеристик (атрибутів та методів) виділених класів.

Атрибути описують властивості об'єкта класу. Атрибути можуть бути простими (наприклад, тип даних int, string) або складними (масиви, об'єкти інших класів).

Методи описують дії, які може виконувати об'єкт класу. Методи можуть мати параметри та повертати значення.

Важливо зазначити, що набори атрибутів і методів у різних класах можуть перетинатися. Це означає, що один і той самий атрибут або метод може використовуватися в декількох класах.

1.2.3 Виявлення відносин між класами

Одним з ключових принципів ООП є інкапсуляція, яка передбачає приховування внутрішньої реалізації класу та надання доступу до нього лише через його інтерфейс. Цей принцип реалізується за допомогою відносин між класами.

Існує два основних типи відносин між класами:

Успадкування - це відношення, яке дозволяє одному класу (потомку) успадковувати властивості та методи іншого класу (батька). Це означає, що об'єкти класу-потомка мають доступ до всіх атрибутів і методів класу-батька.

Агрегація - це відношення, яке описує зв'язок частин з цілим. Це означає, що один об'єкт класу може містити один або декілька об'єктів іншого класу.

1.2.4 Застосування принципів ООП

На етапі ООП-проектування важливо враховувати не тільки особливості предметної області, але й такі принципи ООП як абстракція, інкапсуляція, поліморфізм.

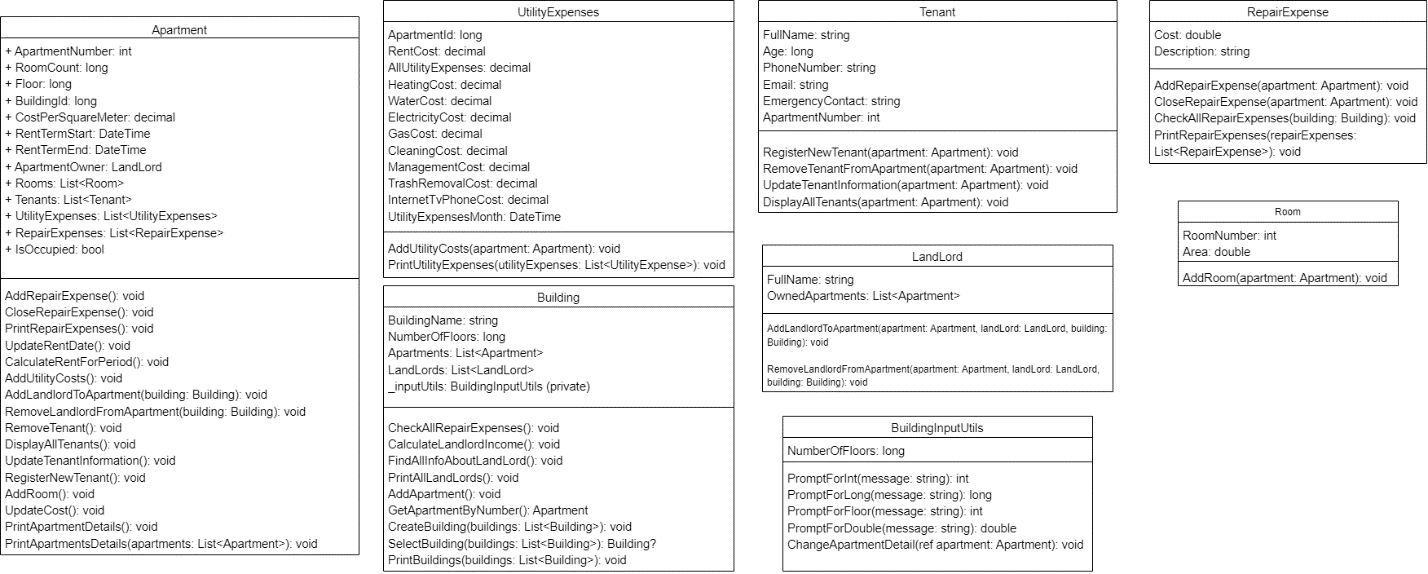
Абстракція - це принцип, який дозволяє приховувати деталі реалізації об'єкта та надавати користувачеві доступ лише до його суттєвих характеристик.

Інкапсуляція - це принцип, який описує приховування внутрішньої реалізації об'єкта та надання доступу до нього лише через його інтерфейс.

Поліморфізм - це принцип, який дозволяє об'єктам різних класів реалізовувати одну і ту ж операцію по-різному.

1.2.5 Результати проектування

Результатом ООП-проектування є діаграма класів, яка відображає зв'язки між класами та їхні характеристики. Діаграма класів є важливим інструментом для розуміння архітектури системи та її подальшої реалізації.

Діаграмма классів **– рис 1.2**

**Рис 1.2**

* 1. **Об'єктно-орієнтоване програмування**
  2. **Тестування програми**
  3. **Оформлення пояснювальної записки**

**ДОДАТОК Б ТИТУЛЬНА СТОРІНКА**

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(повна назва кафедри, циклової комісії)

**КУРСОВА РОБОТА**

з \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студента (ки) \_\_\_\_\_ курсу \_\_\_\_\_\_ групи

Спеціальності 121 «Інженерія програмного забезпечення»

Керівник \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Кількість балів: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії

\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

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

Київ- 2024 рік

ДОДАТОК В ЛИСТ ЗАВДАННЯ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені ІГОРЯ СІКОРСЬКОГО»

Кафедра інформатики та програмної інженерії

Дисципліна Основи програмування

Спеціальність «Інженерія програмного забезпечення»

Курс \_\_ Група ІП-\_\_\_ Семестр \_\_\_

**ЗАВДАННЯ**

**на курсову роботу студента**

Ткаченко Костянтин Олександрович

(прізвище, ім’я, по батькові)

1. Тема роботи

2. Строк здачі студентом закінченої роботи

3. Вихідні дані до роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці)

5. Перелік графічного матеріалу ( з точним зазначенням обов’язкових креслень)

6. Дата видачі завдання

**КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Назва етапів курсової роботи | Термін вико нання етапів  роботи | Підписи ке рівника, сту дента |
| 1. | Отримання теми курсової роботи |  |  |
| 2. | Підготовка ТЗ |  |  |
| 3. | Аналіз предметної області |  |  |
| 4. | Проектування архітектури програмної системи |  |  |
| 5. | Розробка сценарію роботи програми |  |  |
| 6. | Узгодження з керівником інтерфейсу користувача |  |  |
| 7. | Розробка програмного забезпечення |  |  |
| 8. | Узгодження з керівником плану тестування |  |  |
| 9. | Тестування програми |  |  |
| 10. | Підготовка пояснювальної записки |  |  |
| 11. | Здача курсової роботи на перевірку |  |  |
| 12. | Захист курсової роботи |  |  |

Студент

(підпис) (прізвище, ім’я, по батькові)

Керівник

(підпис) (прізвище, ім’я, по батькові)

"\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_ р.

**Анотація:**

Курсова робота реалізує консольний додаток для управління житловими будинками, апартаментами, мешканцями та власниками будинків. Додаток надає можливість створення та вибору будинків, додавання апартаментів до будинків, а також управління апартаментами, мешканцями та власниками. Користувач може здійснювати фінансові транзакції, такі як додавання витрат на комунальні послуги та розрахунок орендної плати. Також реалізовані функції для управління власниками будинків, включаючи перегляд інформації про власників та розрахунок загального доходу від оренди. Використовується об'єктно-орієнтований підхід для створення та управління об'єктами будинків, апартаментів та мешканців.

Вступ ................................................................................................................. 24

Опис основного функціоналу .............................................................................. 4

Реалізація консольного додатку .......................................................................... 5

2.1. Основна частина;................................................................................... 5

2.2.1 Постановка задачі.....................................................

2.2.2 Аналіз предметної області.......................................... 8

2.3.3 Опис архітектури програмної системи ........................................ 10

2.4.4 Опис програмного забезпечення ................................................. 12

2.5.5 Результати тестування .................................................................. 14

2.6.6 Інструкція користувача ................................................................ 15

Висновки ........................................................................................................... 16

Список використаних джерел ............................................................................. 17

Додатки .............................................................................................................. 18

**Вступ**

Курсова робота має на меті розробку консольного додатку для керування житловими будинками, апартаментами, мешканцями та власниками. Головною метою цього додатку є створення зручного та ефективного інструменту для автоматизації процесів управління нерухомістю. У вступі обґрунтовується актуальність теми дослідження, а також вказується на потребу у розробці програмного забезпечення, спрямованого на оптимізацію управління житловими приміщеннями та пов'язаними з ними процесами.

Для досягнення цієї мети у розділі "Опис основного функціоналу" надається огляд функцій, які має виконувати розроблений додаток. Детально розглядається функціонал управління будинками, апартаментами, мешканцями та власниками, а також фінансові транзакції, пов'язані з управлінням нерухомістю.

У наступному розділі "Реалізація консольного додатку" надається опис структури проекту та основного алгоритму роботи програми. Детально розглядаються основні класи та їх взаємодія.

В розділі "Управління будинками" розглядаються процеси створення та вибору будинків у додатку. Описується функціонал для додавання та вибору будівельних об'єктів.

Послідовно, у розділі "Управління апартаментами" надається огляд функціоналу для управління апартаментами, включаючи їх створення, перегляд інформації та роботу з мешканцями та власниками.

У "Фінансові транзакції" описуються процеси фінансового обліку в додатку, включаючи роботу з витратами та розрахунок арендної плати.

В розділі "Управління власниками будинків" аналізується функціонал, пов'язаний з управлінням власниками нерухомості, включаючи перегляд інформації про власників та розрахунок доходу від оренди.

* 1. **Опис основного функціоналу**

Цей розділ надає огляд основного функціоналу розробленого консольного додатку для керування житловими будинками, апартаментами, мешканцями та власниками. Описуються основні можливості програми, які включають в себе:

**Створення та управління будинками:**

* + Можливість створення нових будинків з вказанням основних характеристик.
  + Перегляд списку існуючих будинків.
  + Вибір конкретного будинку для подальшої роботи.

**Управління апартаментами:**

* + Додавання нових апартаментів до вже існуючих будинків.
  + Перегляд інформації про апартаменти, включаючи їх розмір, кількість кімнат тощо.
  + Обробка дій з мешканцями та власниками апартаментів, такі як реєстрація нових мешканців, оновлення інформації про них тощо.

**Фінансові транзакції:**

* + Додавання витрат на комунальні послуги та інші розходи.
  + Розрахунок орендної плати за проживання в апартаментах.

**Управління власниками будинків:**

* + Перегляд інформації про власників нерухомості.
  + Розрахунок доходу від оренди за апартаменти.

Цей розділ робить акцент на функціях програмного забезпечення, які дозволяють зручно та ефективно керувати житловою нерухомістю та пов'язаними з нею процесами.

**2. Реалізація консольного додатку**

**2.1**. **Структура проекту**

Структура проекту консольного додатку включає наступні основні класи:

* **Клас Program**: Це вхідна точка програми, де знаходиться метод Main(). В цьому класі відбувається ініціалізація додатку та його головний цикл роботи.
* **Клас Building**: Відповідає за моделювання будинків, зберігає інформацію про апартаменти та власників.
* **Клас Apartment**: Представляє модель апартаментів у будинках. Містить методи для управління мешканцями, власниками та фінансовими операціями.
* **Клас Landlord**: Відповідає за моделювання власників нерухомості та їхню інформацію.
* **Клас UtilityExpense**: Моделює витрати на комунальні послуги та інші витрати.
* **Клас RepairExpense**: Моделює витрати на ремонтні роботи в апартаментах.
* **Клас Room**: Відповідає за моделювання кімнат у апартаментах.
* **Клас Tenant**: Представляє модель мешканців апартаментів.
* **Клас BuildingInputUtils**: Надає допоміжні методи для обробки введення даних про будинки.

**2.2**. **Основний алгоритм роботи**

1. **Ініціалізація та завантаження даних:**
   * При запуску програми відбувається ініціалізація необхідних змінних та структур даних.
   * Завантажуються дані про наявні будівлі та апартаменти з файлу (якщо такий існує).
2. **Відображення головного меню:**
   * Користувачеві пропонується меню з трьома пунктами:
     + Створення нового будинку
     + Вибір існуючого будинку
     + Перегляд списку всіх будинків
3. **Обробка вибору користувача:**
   * Залежно від обраного пункту меню виконується відповідна дія:
     + **Створення нового будинку:**
       - Запит у користувача інформації про новий будинок (назва, адреса, кількість поверхів тощо).
       - Створення нового об'єкта "Будинок" з отриманою інформацією.
       - Збереження нового будинку у списку будівель.
     + **Вибір існуючого будинку:**
       - Відображення списку доступних будівель.
       - Запит у користувача номера вибраного будинку.
       - Завантаження інформації про обраний будинок з списку або з файлу.
       - Перехід до меню роботи з обраним будинком.
     + **Перегляд списку всіх будинків:**
       - Відображення списку всіх доступних будівель з базовою інформацією (назва, адреса).
       - Повернення до головного меню.
4. **Робота з обраним будинком:**
   * Після вибору будинку користувачеві пропонується меню з такими пунктами:
     + Додати апартамент
     + Переглянути всі апартаменти
     + Вибрати апартамент
     + Операції з власниками
     + Вихід
   * Обробка вибору користувача аналогічно до головного меню.
5. **Робота з апартаментом:**
   * Після вибору апартаменту користувачеві пропонується меню з такими пунктами:
     + Керування апартаментом
     + Керування мешканцями
     + Керування власниками
     + Фінансові операції
     + Вихід
   * Обробка вибору користувача аналогічно до меню роботи з будинком.
6. **Повторення циклу:**
   * Пункти 2-5 повторюються до тих пір, поки користувач не обере пункт "Вихід" на будь-якому рівні меню.
7. **Завершення роботи:**
   * При виборі пункту "Вихід" програма завершує роботу.

**Основна частина**

**2.2.1 Постановка задачі**

У цьому розділі буде представлена інформаційна сутність задачі, яка є ключовою для розробки програмної системи. Основною метою цього розділу є аналіз специфікації вимог технічного завдання на курсовий проект та формулювання додаткових вимог, що випливають із нього та мети курсової роботи.

Результатом аналізу буде конкретизація постановки завдання, структури вхідних та вихідних даних, основних операцій та методів, які використовуються для їх обробки. Також будуть визначені основні параметри та функціональні можливості програмної системи, що становлять загальну концепцію розроблення.

Інформаційна сутність задачі включає:

* Опис предметної області: ОСББ (Об’єднання співвласників багатоквартирного будинку) та управління багатоквартирним будинком.
* Опис функціоналу електронного довідника даних про мешканців та орендарів, включаючи відомості про ПІБ, номер квартири, термін та вартість оренди, розмір квартири, кількість мешканців, вартість квартирної плати та інші параметри.
* Вимоги до розробки програмної системи: використання об’єктно-орієнтованого підходу, дотримання принципів ООП, впровадження обробки виключень, застосування єдиної конвенції найменувань та принципів написання "чистого" коду.
* Перелік основних операцій та методів, які необхідно реалізувати у програмній системі.
* Опис вхідних та вихідних даних, включаючи формати та способи представлення інформації.

Цей розділ визначає загальний контекст та обґрунтовує необхідність розробки програмної системи, що забезпечить ефективне управління багатоквартирним будинком та покращить взаємодію між його мешканцями та орендарями.

**2.2.2 Аналіз предметної області**

У цьому розділі буде проведено аналіз предметної області проекту, що полягатиме у реферативному описі основних аспектів ОСББ (Об’єднання співвласників багатоквартирного будинку) та управління багатоквартирним будинком.

Реферативний опис предметної області:

* ОСББ: це об’єднання власників квартир у багатоквартирному будинку, які спільно приймають рішення щодо управління та експлуатації спільних частин будинку.
* Управління багатоквартирним будинком: включає в себе організацію обслуговування будинку, забезпечення ремонтних робіт, контроль за платежами та інші аспекти повсякденного життя мешканців та орендарів.
* Електронний довідник даних: програмна система, яка забезпечує зберігання та обробку інформації про мешканців та орендарів багатоквартирного будинку, включаючи їх особисті дані, інформацію про квартири, умови оренди, платежі та інше.

Діаграма прецедентів системи:

1. Реєстрація нових мешканців та орендарів.
2. Оновлення інформації про мешканців та орендарів.
3. Перегляд інформації про всіх мешканців та орендарів.
4. Видалення мешканців та орендарів з бази даних.
5. Перегляд інформації про конкретного мешканця або орендаря.

Сценарій роботи програми:

1. Користувач запускає програму.
2. Вибирає необхідну опцію з меню.
3. Здійснює відповідні дії, які включають реєстрацію, оновлення, перегляд та видалення інформації про мешканців та орендарів.
4. Завершує роботу програми.

Аналіз предметної області дозволяє краще зрозуміти потреби користувачів та визначити функціональні можливості програмної системи.

**2.2.3 Опис архітектури програмної системи**

**2.2.4 Опис програмного забезпечення**

**2.3.3 Опис архітектури програмної системи**

**2.4.4 Опис програмного забезпечення**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Назва классу | Назва методу | Призначення  методу | Опис вхідних  Парамет ів | Опис вихідних парамет рів | Заголовний  файл |
| 1 | Apartment | AddRepairExpense() | Додавання нового запису про витрати на ремонт апартаментів. | - | - | Apartment.cs |
| 2 | Apartment | CloseRepairExpense() | Закрити запис про витрати на ремонт апартаментів. | - | - | Apartment.cs |
| 3 | Apartment | PrintRepairExpenses() | Вивести на друк інформацію про витрати на ремонт апартаментів. | - | - | Apartment.cs |
| 3 | Apartment | UpdateRentDate() | Оновити дату початку та кінця орендного терміну для апартаментів. | - | - | Apartment.cs |
| 4 | Apartment | CalculateRentForPeriod() | Розрахувати та вивести загальну суму орендної плати за певний період. | - | - | Apartment.cs |
| 5 | Apartment | AddUtilityCosts() | Додати новий запис про витрати на комунальні послуги для апартаментів. | - | - | Apartment.cs |
| 6 | Apartment | AddLandlordToApartment(Building building) | Додати орендодавця до апартаментів. | building (Building): Об'єкт класу Building, що представляє будівлю, до якої належать апартаменти | - | Apartment.cs |
| 7 | Apartment | RemoveLandlordFromApartment(Building building) | RemoveLandlordFromApartment(Building building) | building (Building): Об'єкт класу Building, що представляє будівлю, до якої належать апартаменти. | - | Apartment.cs |
| 8 | Apartment | RemoveTenant() | Видалити орендаря з апартаментів. | - | - | Apartment.cs |
| 9 | Apartment | DisplayAllTenants() | Вивести інформацію про всіх орендарів апартаментів. | - | - | Apartment.cs |
|  | Apartment | UpdateTenantInformation() | Оновити інформацію про орендаря апартаментів. | - | - | Apartment.cs |
|  | Apartment | RegisterNewTenant() | Зареєструвати нового орендаря для апартаментів. | - | - | Apartment.cs |
|  | Apartment | AddRoom() | Додати кімнату до апартаментів. | - | - | Apartment.cs |
|  | Apartment | UpdateCost() | Цей метод дозволяє оновити вартість оренди за квадратний метр для апартаментів. | - | - | Apartment.cs |
|  | Apartment | PrintApartmentDetails() | Друк детальної інформації про апартаменти. | - | - | Apartment.cs |
|  | Apartment | PrintApartmentsDetails(List<Apartment> apartments) (статичний): | Друк інформації про декілька апартаментів. | apartments (List<Apartment>): Список апартаментів, для яких потрібно вивести інформацію. | - | Apartment.cs |
|  | Apartment | CalculateRentForPeriod(DateTime startPeriod, DateTime zendPeriod) | Ця приватна функція розраховує суму оренди за заданий період. | startPeriod: Дата початку періоду оренди.  endPeriod: Дата закінчення періоду оренди. | Повертає double значення, що являє собою загальну суму оренди за заданий період. | Apartment.cs |
|  | Apartment | CalculateTotalArea() | Ця приватна функція розраховує загальну площу квартири. | - | Повертає double значення, що являє собою загальну площу квартири. | Apartment.cs |
|  | Building | CheckAllRepairExpenses | Ця функція викликає метод CheckAllRepairExpenses класу RepairExpense, ймовірно, передаючи поточний об'єкт Building як аргумент. | this (об'єкт Building) | - | Building.cs |
|  | Building | CalculateLandlordIncome | Ця функція розраховує дохід власника нерухомості на основі оренди його квартир протягом певного періоду. | - | - | Building.cs |
|  | Building | FindAllInfoAboutLandLord() | Знайти всю інформацію про конкретного власника нерухомості. | - | - | Building.cs |
|  | Building | PrintAllLandLords() | **:** Вивести список усіх власників нерухомості. | - | - | Building.cs |
|  | Building | AddApartment() | Додати нову квартиру до списку квартир будів | - | - | Building.cs |
|  | Building | GetApartmentByNumber() | Отримати об'єкт квартири за її номером. | - | Об'єкт класу Apartment (якщо знайдено квартиру за номером) або null (якщо квартира не знайдена) | Building.cs |
|  | Building | CreateBuilding() | Створити новий об'єкт класу Building та додати його до списку будівель. | List<Building> buildings: Список будівель, до якого буде додано новий об'єкт. | - | Building.cs |
|  | Building | SelectBuilding() | Обрати будівлю з наданого списку. | List<Building> buildings: Список будівель, з якого користувач буде обирати. | Building?: Об'єкт класу Building (якщо вибрано будівлю) або null (якщо вибір недійствний або немає будівель) | Building.cs |
|  | Building | PrintBuilding() | Вивести список усіх будівель. | List<Building> buildings: Список будівель для виводу. | - | Building.cs |
|  | RepairExpense | AddRepairExpense | Додати нові витрати на ремонт до квартири. | Apartment apartment: Об'єкт класу Apartment, до якого додаються витрати на ремонт. | - | RepairExpense.cs |
|  | RepairExpense | CloseRepairExpense | Закрити витрати на ремонт (видалити їх зі списку витрат квартири). | Apartment apartment: Об'єкт класу Apartment, з якого видаляються витрати на ремонт. | - | RepairExpense.cs |
|  | RepairExpense | CheckAllRepairExpenses | Перевірити всі витрати на ремонт для певного власника нерухомості. | Building building: Об'єкт класу Building, який містить інформацію про власників та їх квартири. | - | RepairExpense.cs |
|  | RepairExpense | PrintRepairExpenses | Вивести список витрат на ремонт. | List<RepairExpense> repairExpenses: Список об'єктів класу RepairExpense, що містить дані про витрати на ремонт. | - | RepairExpense.cs |
|  | Room | AddRoom | Додати нову кімнату до квартири. | Apartment apartment: Об'єкт класу Apartment, до якого додається кімната. | - | Room.cs |
|  | UtilityExpense | AddUtilityCosts | Додати витрати на комунальні послуги для певного місяця до квартири. | Apartment apartment: Об'єкт класу Apartment, до якого додаються витрати на комунальні послуги. | - | UtilityExpense.cs |
|  | UtilityExpense | PrintUtilityExpenses | Вивести список витрат на комунальні послуги. | List<UtilityExpense> utilityExpenses: Список об'єктів класу UtilityExpense, що містить дані про витрати на комунальні послуги. | - | UtilityExpense.cs |
|  | LandLord | AddLandlordToApartment | Додати власника нерухомості (LandLord) до квартири. | Apartment apartment: Об'єкт класу Apartment, до якого додається власник.  Building building: Об'єкт класу Building, до якого належить квартира. | - | LandLord.cs |
|  | LandLord | RemoveLandlordFromApartment | Видалити власника нерухомості з квартири. | Apartment apartment: Об'єкт класу Apartment, до якого додається власник.  Building building: Об'єкт класу Building, до якого належить квартира. | - | LandLord.cs |
|  | Tenant | RegisterNewTenant | Зареєструвати нового орендаря в квартирі. | Apartment apartment: Об'єкт класу Apartment, в якому реєструється орендар. | - | Tenant.cs |
|  | Tenant | RemoveTenantFromApartment | Видалити орендаря з квартири. | Apartment apartment: Об'єкт класу Apartment, з якого видаляється орендар. | - | Tenant.cs |
|  | Tenant | UpdateTenantInformation | Оновити інформацію про орендаря. | Apartment apartment: Об'єкт класу Apartment, в якому проживає орендар. | - | Tenant.cs |
|  | Tenant | DisplayAllTenants | Вивести список усіх орендарів квартири. | Apartment apartment: Об'єкт класу Apartment, для якого виводяться орендарі. | - | Tenant.cs |
|  | BuildingInputUtils | PromptForInt | Запросити в користувача ціле число. | message (string): Строкове повідомлення, що виводиться перед запитом вводу. | int: Ціле число, введене користувачем. | BuildingInputUtils.cs |
|  | BuildingInputUtils | PromptForLong | Запросити в користувача довге ціле число. | message (string): Строкове повідомлення, що виводиться перед запитом вводу. | long: Довге ціле число, введене користувачем. | BuildingInputUtils |
|  | BuildingInputUtils | PromptForFloor | Запросити в користувача номер поверху, враховуючи загальну кількість поверхів у будівлі. | message (string): Строкове повідомлення, що виводиться перед запитом вводу. | int: Номер поверху, введений користувачем. | BuildingInputUtils |
|  | BuildingInputUtils | PromptForDouble | Запросити в користувача число з рухомою комою. | message (string): Строкове повідомлення, що виводиться перед запитом вводу. | double: Число з рухомою комою, введене користувачем. | BuildingInputUtils |
|  | BuildingInputUtils | ChangeApartmentDetail | Дозволяє змінити деталі квартири. | apartment (ref Apartment): Посилання на об'єкт класу Apartment, деталі якого потрібно змінити. | - | BuildingInputUtils |

**Висновок**

У даній курсовій роботі було розроблено консольний додаток для управління житловою нерухомістю, який надає зручний і ефективний інтерфейс користувачу для керування будинками, апартаментами та їхніми власниками. Додаток включає розділи для створення, вибору та управління будинками, керування апартаментами, роботи з жителями та операцій з власниками апартаментів.

Кожен розділ додатку має свій функціонал, який дозволяє користувачеві виконувати різноманітні дії, такі як створення нових будівель, додавання апартаментів, реєстрація та оновлення інформації про жителів, а також управління власниками апартаментів.

Додаток реалізовано з використанням мови програмування C#, що дозволило створити ефективну та зручну програму з легкою навігацією та виконанням необхідних завдань з керування нерухомістю через консольний інтерфейс.

В цілому, розроблений додаток надає зручні та потужні інструменти для керування житловою нерухомістю, що робить його корисним інструментом для управлінців нерухомістю та агентств з нерухомості.

**2.1 Реалізація консольного додатку**

* 1. **Захист курсової роботи**