

НАЦИОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
УКРАЇНИ «Київський політехнічний інститут»
Факультет інформатики та обчислювальної техніки
Кафедра автоматизованої обробки інформації та управління

КУРСОВА РОБОТА
по дисципліні
Об'єктно-орієнтоване програмування
(назва дисципліни)

Варіант № 6

Виконав: студент групи ІП-82
Волобуєв Н.О.
(Прізвище та ініціали)

Прийняла доцент Іванова Л.М.

Київ – 2020

ЗМІСТ

ЗАВДАННЯ	3
1 ОПИС ЗАСТОСУВАННЯ	4
1.1. Архітектура застосування	4
1.2. Діаграма варіантів використання	7
1.3. Діаграма класів	8
2 РЕКОМЕНДАЦІЇ ЩОДО ВИКОРИСТАННЯ ЗАСТОСУВАННЯ	14
2.1. Технічні характеристики	14
2.2. Рекомендації по встановленню та налаштуванню	14
2.3. Рекомендації користувачу	15
3 ВИХІДНИЙ КОД ЗАСТОСУВАННЯ	17
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	18

ЗАВДАННЯ

Написати консольне застосування за архітектурним шаблоном MVC з розподілом відповідальності між компонентами, яке виконує обробку набору даних згідно з варіантом завдання та забезпечує наступне:

1. Збереження даних у файлі (формат файлу будь-який);
2. Читання даних у пам'ять при запуску застосунку;
3. Збереження даних у той же файл при завершенні роботи застосунку, якщо дані були змінені;
4. Збереження проміжних даних у будь-який файл (користувач вводить ім'я файлу);
5. Інтерактивність з користувачем (мова інтерфейсу має обиратися при запуску застосунку на виконання);
6. Логування подій та помилок в роботі застосунку.

Завдання — 6 варіант:

1. Отримати список квартир, які мають задану кількість кімнат.
2. Отримати список квартир, які мають площу більшу заданої та розташовані вище заданого поверху.

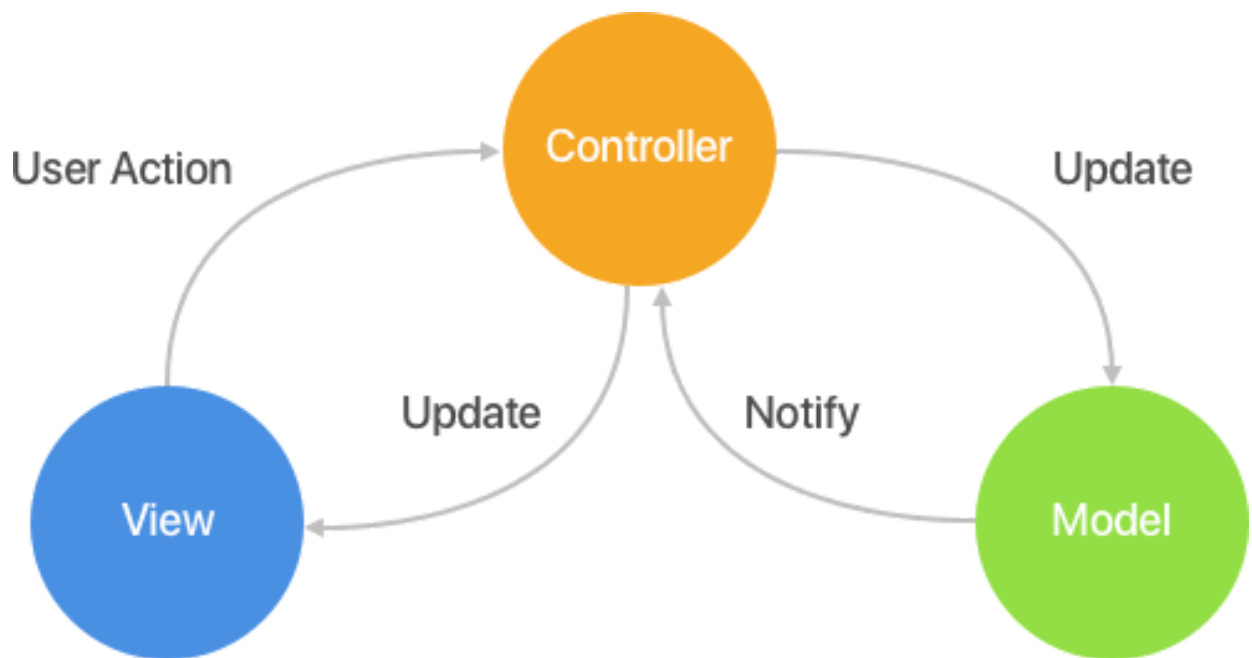
Квартира: Номер квартири, Площа, Поверх, Кількість кімнат, Тип будівлі, Строк експлуатації. Конструктор, Методи доступу, Метод toString()

1 ОПИС ЗАСТОСУВАННЯ

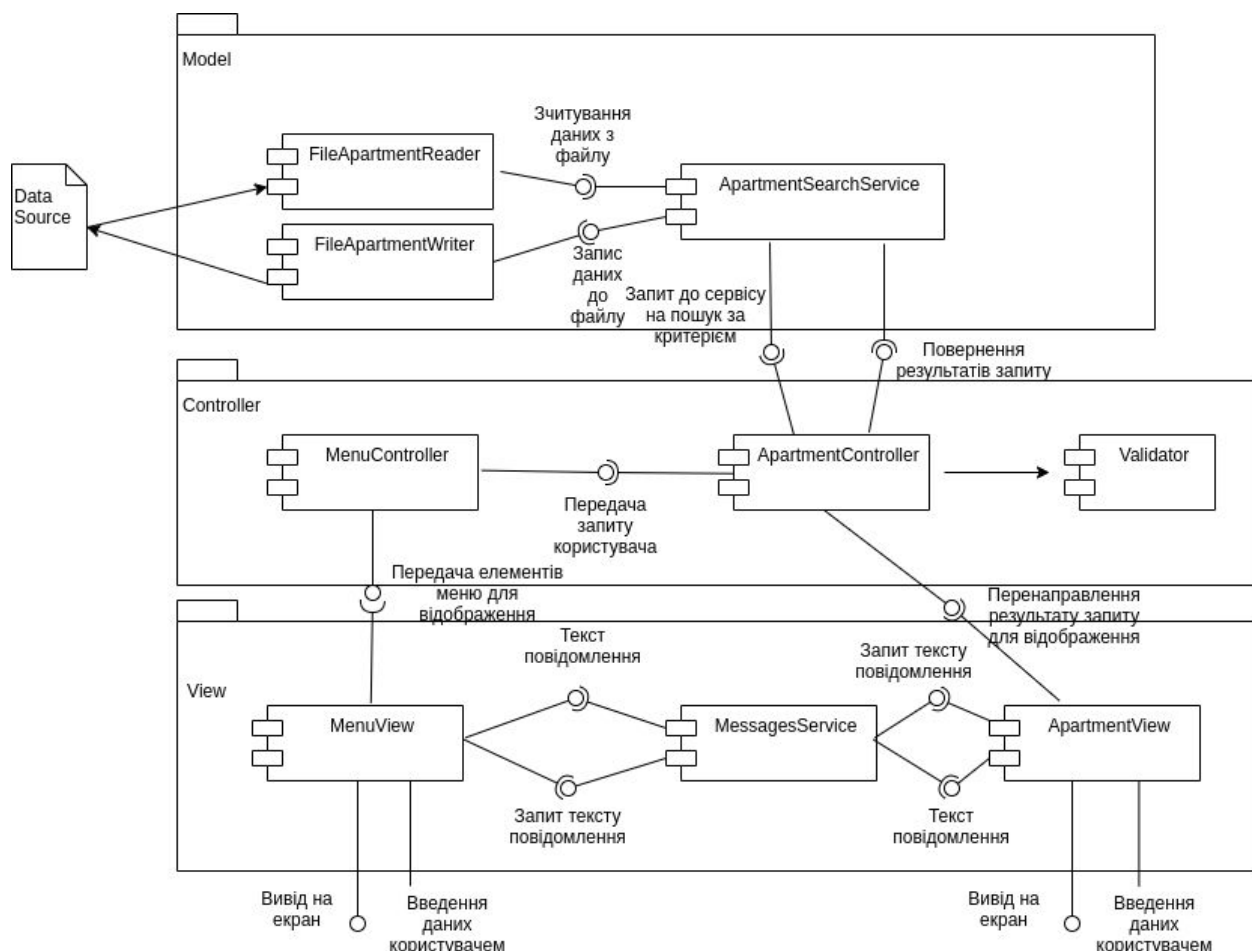
1.1. Архітектура застосування

Архітектурним шаблоном даної роботи було обрано MVC. Причиною використання цієї структури була організація структури системи, що дозволяє відокремити логіку від користувацького інтерфейсу. За MVC це досягається розділенням додатку на три частини: Model (модель), View (вид), Controller (контролер).

Controller отримує введені користувачем дані від View, виконує виклики до Model, після чого передає результат виконання до View. View надає користувацький інтерфейс для взаємодії з користувачем. Він обробляє введення даних користувачем та відображення даних у зрозумілій формі. Model зберігає дані застосування та реагує на їх зміну.



Мал. 1 - Архітектурний шаблон MVC



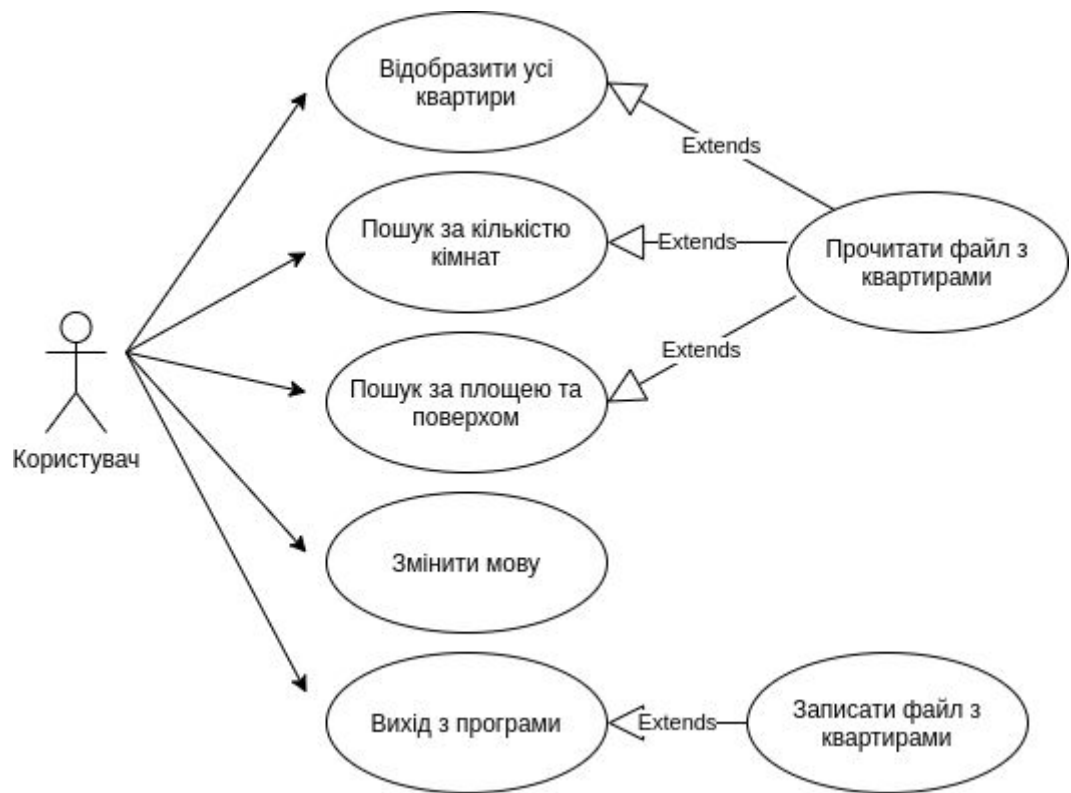
Мал. 2 - Діаграма компонентів програми

На діаграмі зображено основні компоненти програми, розробленої у даній курсовій роботі, та зв'язки між ними. На діаграмі усі компоненти розділені на три частини: Model, View та Controller.

При виконанні програми першим кроком MenuController за допомогою MenuView отримує вибір деякої опції меню користувачем. Після цього, в залежності від обраної функції, керування передається до ApartmentController. Цей контролер, в свою чергу, за допомогою ApartmentView (ApartmentSearchUserPromptView, ApartmentSearchUserInput) отримує від користувача запит на пошук, а саме значення критеріїв цього пошуку. Дані, введені користувачем передаються у Validator на перевірку. У разі виникнення помилки, ApartmentController викликає потрібний метод у ApartmentView для повідомлення користувача про це. У разі успішної

валідації, запит на пошук передається до Моделі — у ApartmentSearchService, який за заданими критеріями виконує пошук та повертає результат до ApartmentController, який передає їх для виведення у ApartmentView. Слід зазначити, що на рівні моделі присутні компоненти FileApartmentReader та FileApartmentWriter, які виконують читання та запис у файл відповідно. Для локалізації на рівні View існує компонент MessagesService, який повертає локалізовані повідомлення в залежності від обраної користувачем мови.

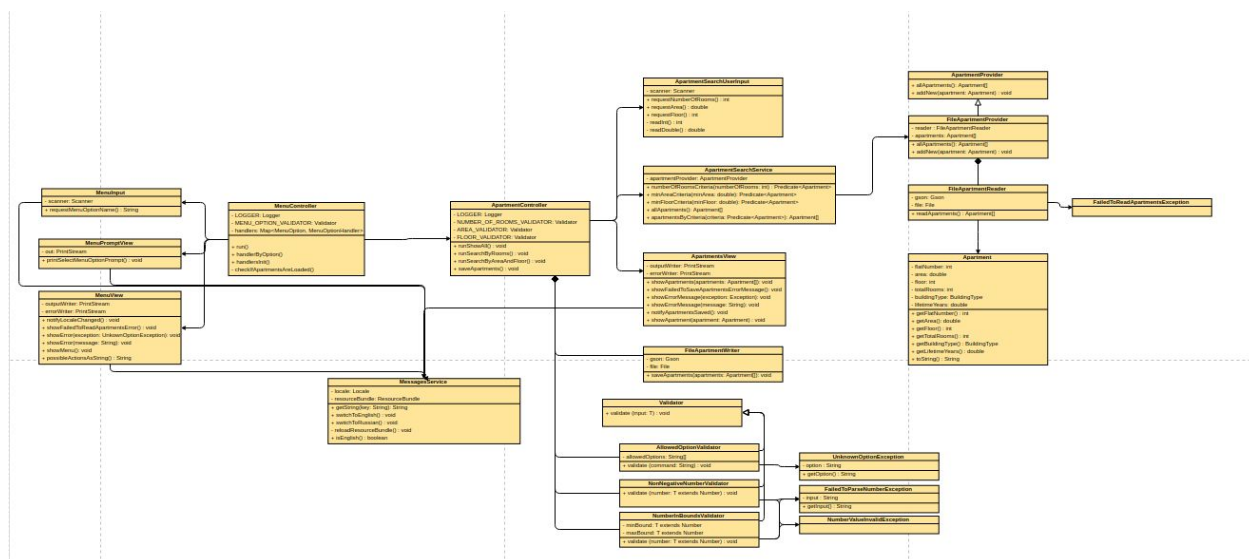
1.2. Діаграма варіантів використання



Мал. 3 - Діаграма використання застосування (Use Case Diagram)

На Use Case діаграмі, що наведена вище, показано основні способи використання програми. До них входять: відображення списку усіх квартир, пошук за критеріями (кількість кімнат, площа, поверх), можливість зміни мови та вихід з програми.

1.3. Діаграма класів



Мал. 3 - Діаграма класів застосування (Class Diagram)

На цій діаграмі можна побачити структуру курсового проекту: класи та взаємозв'язки між ними.

В пакеті controller знаходяться: MenuController, ApartmentController.

В пакеті input знаходяться класи, які належать до View та відповідають за отримання даних введених користувачем: ApartmentSearchUserInput, MenuInput - використовуються з контролерів.

До пакету models належать: Apartment — сутність квартири. Також в цьому пакеті знаходиться BuildingType - який є перерахуванням усіх можливих типів будівель. Також присутній клас MenuOption, який містить перелік існуючих опцій меню. Також у цьому пакеті є пакет persistence, де знаходяться FileApartmentReader, FileApartmentWriter, які відповідають за читання/запис файлів з даними. У пакеті providers знаходяться класи, які є джерелом даних: ApartmentProvider, та FileApartmentProvider, що реалізують його. При читанні може виникати помилка FailedToLoadApartmentsException, яка винесена до пакету exceptions. У пакеті services знаходиться ApartmentSearchService, який реалізує логіку пошуку квартири за заданими

критеріями. Також тут присутній клас `MessagesService`, який передає повідомлення з урахуванням локалі.

У пакеті `validation` знаходяться класи, які виконують валідацію даних, введених користувачем (реалізують інтерфейс `Validator`): `AllowedOptionValidator`, `NonNegativeNumberValidator`, `NumberInBoundsValidator`. При валідації можуть виникати помилки, які винесені в пакет `exceptions`: `FailedToParseNumberException`, `NumberValueInvalidException`, `UnknownOptionException`.

У пакеті `views` знаходяться класи, які входять до `View` частини та виконують відображення даних: `ApartmentSearchUserPromptView`, `ApartmentsView`, `MenuPromptView`, `MenuView`.

Детальний опис методів усіх класів подається у наступній таблиці:

№ п/п	Назва класу	Назва методу	Призначення методу	Опис вхідних параметрів	Опис вихідних параметрів
1	<code>ApartmentController</code>	<code>runShowAll</code>	Запускає процес показу усіх квартир	-	<code>void</code>
2		<code>runSearchByRooms</code>	Запускає процес пошуку по кількості кімнат	-	<code>void</code>
3		<code>runSearchByAreaAndFloor</code>	Запускає процес пошуку по площі та поверху	-	<code>void</code>
4		<code>saveApartments</code>	Запускає зберігання квартир до файлу	-	<code>void</code>
5	<code>MenuController</code>	<code>run</code>	Запускає процес взаємодії з користувачем у меню	-	<code>void</code>
6		<code>handlerByOption</code>	Повертає обробник деякого пункту меню	<code>MenuOption</code> option - пункт меню	<code>MenuOptionHandler</code>
7		<code>handlersInit</code>	Створює усі обробники меню	<code>ApartmentController</code> <code>apartmentController</code>	<code>Map<MenuOption, MenuOptionHandler></code>

8	ApartmentSearchUserInput	requestNumberOfRooms	Запросити у користувача кількість кімнат	-	int
9		requestArea	Запросити у користувача площу квартири	-	double
10		requestFloor	Запросити у користувача поверх квартири	-	int
11		readInt	Прочитати ціле число з стандартного потіку вводу	-	int
12		readDouble	Прочитати дробове число з стандартного потіку вводу	-	double
13	MenuInput	requestMenuOptionName	Прочитати назву опції меню з стандартного потіку вводу	-	String
14	FileApartmentReader	readApartments	Прочитати квартири з файлу	-	Apartment[]
15	FileApartmentWriter	saveApartments	Записати квартири до файлу	Apartment[] apartments - квартири для запису	void
16	Apartment	getFlatNumber	Повертає значення номеру квартири	-	int
17		getArea	Повертає значення площі квартири	-	double
18		getFloor	Повертає значення номеру поверху	-	int
19		getTotalRooms	Повертає значення кількості кімнат	-	int
20		getBuildingType	Повертає значення типу будівлі.	-	BuildingType
21		getLifetimeYears	Повертає значення скільки років цьому будинку.	-	int
22		toString	Конвертує об'єкт квартири в рядок	-	String
23	BuildingType	maxNumberOfFlats	Повертає максимальна кількість квартир у будинку	-	int

24		maxArea	Повертає максимальну площу квартири у будинку	-	double
25		maxTotalRooms	Повертає максимальну кількість кімнат квартири будинку	-	int
26		maxLifetime	Повертає максимальний вік будинку	-	int
27	MenuOption	byCommand	Повертає команду меню по її назві	String command - назва команди	MenuOption
28		command	Повертає назву команди	-	String
29	FileApartmentProvider	allApartments	Повертає усі квартири, що було прочитано з файлу	-	Apartment[]
30		addNew	Додає нову квартиру до списку квартир	Apartment apartment - квартира, яку потрібно додати	void
31	ApartmentSearchService	numberOfRoomsCriteria	Повертає предикат, що виконує пошук за кількістю кімнат	int numberOfRooms - кількість кімнат	Predicate<Apartment>
32		minAreaCriteria	Повертає предикат, що виконує пошук за площею	double minArea - мінімальна площа	Predicate<Apartment>
33		minFloorCriteria	Повертає предикат, що виконує пошук за поверхом	int minFloor - мінімальний поверх	Predicate<Apartment>
34		allApartments	Повертає усі квартири	-	Apartment[]
35		apartmentsByCriteria	Повертає усі квартири, для яких виконується заданий предикат критерії	Predicate<Apartment> criteria - критерія.	Apartment[]
36	MessagesService	getString	Повертає локалізований рядок за ключом	String key - ключ	String
37		switchToEnglish	Переключити мову на англійську	-	void
38		switchToRussian	Переключити мову на російську	-	void

39		reloadResourceBundle	Перестворює ResourceBundle (потрібно для переключення на іншу мову)	-	void
40		isEnglish	Повертає чи є вибрана в даний момент мова англійською	-	boolean
41	AllowedOptionValidator	validate	Перевірити, чи входить наведене значення до списку дозволених	String command - значення для перевірки	void
42	NonNegativeNumberValidator	validate	Перевірити, що число не є від'ємним	T input, T extends Number - число для перевірки	void
43	NumberInBoundsValidator	validate	Перевірити, що число належить інтервалу	T input, T extends Number - число для перевірки	void
44	ApartmentSearchUserPromptView	printNumberOfRoomsRequest	Повідомляє про необхідність введення кількості кімнат	-	void
45		printAreaRequest	Повідомляє про необхідність введення площі	-	void
46		printFloorRequest	Повідомляє про необхідність введення номеру поверху	-	void
47		printFilterCriteriaRequest	Повідомляє про необхідність введення критерію пошуку	-	void
48	ApartmentsView	showApartments	Виводить таблицю квартир	Apartment[] apartments - квартири	void
49		showFailedToSaveApartmentsErrorMessage	Повідомляє про помилку під час зберігання квартир у файл	-	void
50		showErrorMessage	Повідомляє про помилку під час роботи	FailedToParseNumberFormatException або	void

				NumberFormatException	
51		showErrorMessage	Повідомляє про помилку під час роботи	String message - текстовий опис помилки	void
52		notifyApartmentSaved	Повідомити про те, що квартири були записані у файл	-	void
53	MenuPromptView	printSelectMenuOptionPrompt	Повідомляє можливі команди для виконання	-	void
54	MenuView	notifyLocaleChanged	Повідомляє про зміну мови	-	void
55		showFailedToReadApartmentsError	Повідомляє про помилку під час читання квартир з файлу	-	void
56		showError	Повідомляє про невідому команду	UnknownOptionException - помилка, яка виникла	void
57		showError	Повідомляє про помилку	String error - опис помилки	void
58		showMenu	Вивести меню	-	void
59		possibleActionsAsString	Повертає список доступних команд як рядок	-	String

2 РЕКОМЕНДАЦІЇ ЩОДО ВИКОРИСТАННЯ ЗАСТОСУВАННЯ

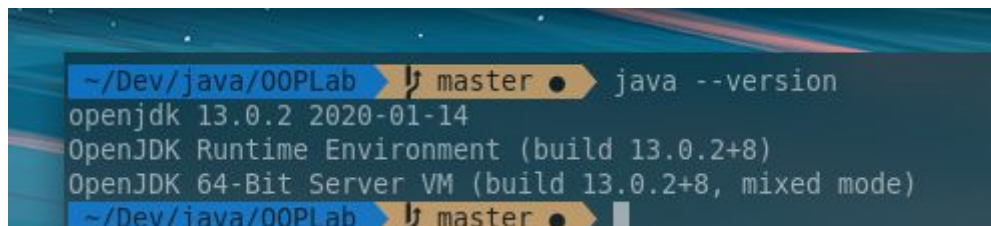
2.1. Технічні характеристики

Для запуску застосування є наступні вимоги до ПК:

- ОС: Windows Vista, 7, 8, 10; Ubuntu Linux 12.04 або новіше, Arch Linux; Mac OS 10.8+
- RAM: як мінімум 128 MB
- Дисковий простір: >150 MB вільного простору
- Процесор: як мінімум Pentium 2 266 MHz
- Браузер: Chrome, Firefox, Edge, Safari

2.2. Рекомендації по встановленню та налаштуванню

Перед встановленням програмного забезпечення, потрібно переконатися, що вже встановлений JRE версії не нижче ніж 8:



```
~/Dev/java/00PLab > master ● java --version
openjdk 13.0.2 2020-01-14
OpenJDK Runtime Environment (build 13.0.2+8)
OpenJDK 64-Bit Server VM (build 13.0.2+8, mixed mode)
~/Dev/java/00PLab > master ●
```

Мал. 4 - Перевірка версії java

У разі необхідності встановлення Java, це можна зробити за наступними посиланнями:

https://java.com/en/download/help/index_installing.xml

<https://openjdk.java.net/install/index.html>

Наступним кроком буде скачування .jar файлу зі сторінки випусків версій репозиторія розробника ПЗ. Це можна зробити завантаживши файл наведений за посиланням:

<https://github.com/nikitavbv/CourseWork2ndYearOOP/releases>

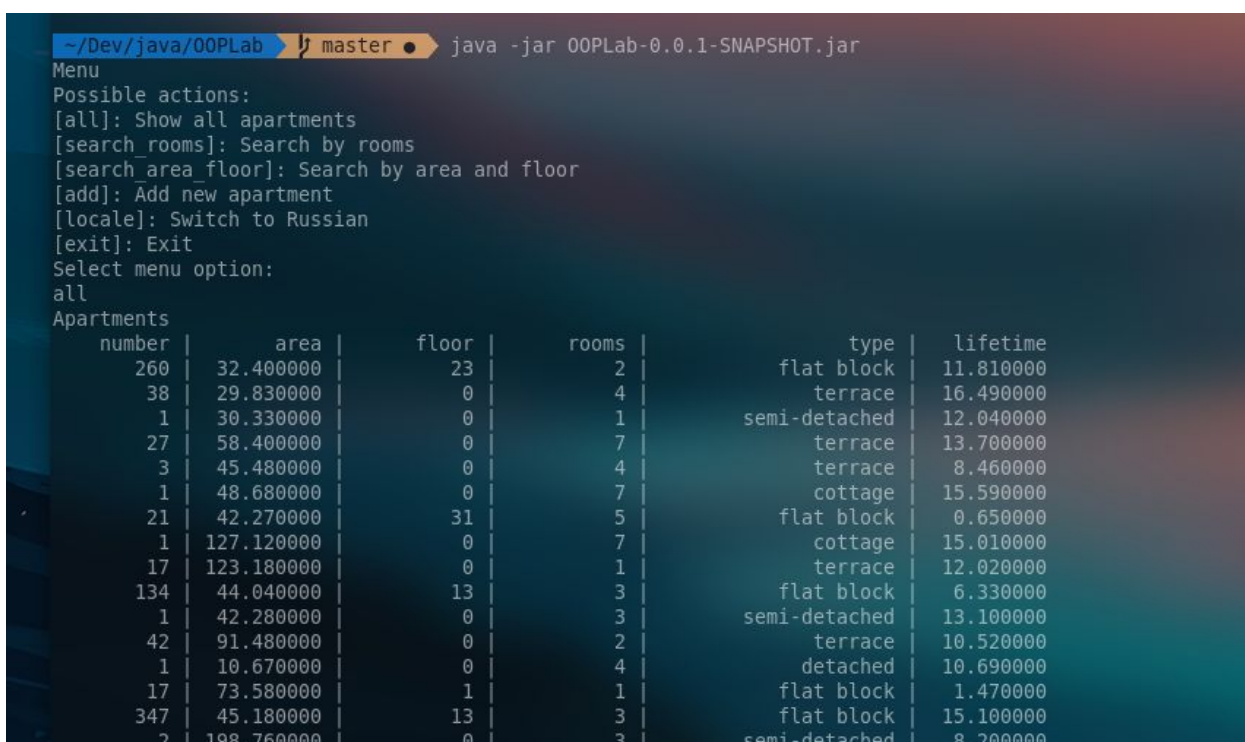
Або завантаження за допомогою wget:

```
wget
```

```
https://github.com/nikitavbv/CourseWork2ndYearOOP/releases/  
download/v0.0.1/OOPLab-0.0.1-SNAPSHOT.jar
```

Для запуску застосунку потрібно ввести:

```
java -jar OOPLab-0.0.1-SNAPSHOT.jar
```



```
~/Dev/java/OOPLab master java -jar OOPLab-0.0.1-SNAPSHOT.jar  
Menu  
Possible actions:  
[all]: Show all apartments  
[search_rooms]: Search by rooms  
[search_area_floor]: Search by area and floor  
[add]: Add new apartment  
[locale]: Switch to Russian  
[exit]: Exit  
Select menu option:  
all  
Apartments  
number | area | floor | rooms | type | lifetime  
260 | 32.400000 | 23 | 2 | flat block | 11.810000  
38 | 29.830000 | 0 | 4 | terrace | 16.490000  
1 | 30.330000 | 0 | 1 | semi-detached | 12.040000  
27 | 58.400000 | 0 | 7 | terrace | 13.700000  
3 | 45.480000 | 0 | 4 | terrace | 8.460000  
1 | 48.680000 | 0 | 7 | cottage | 15.590000  
21 | 42.270000 | 31 | 5 | flat block | 0.650000  
1 | 127.120000 | 0 | 7 | cottage | 15.010000  
17 | 123.180000 | 0 | 1 | terrace | 12.020000  
134 | 44.040000 | 13 | 3 | flat block | 6.330000  
1 | 42.280000 | 0 | 3 | semi-detached | 13.100000  
42 | 91.480000 | 0 | 2 | terrace | 10.520000  
1 | 10.670000 | 0 | 4 | detached | 10.690000  
17 | 73.580000 | 1 | 1 | flat block | 1.470000  
347 | 45.180000 | 13 | 3 | flat block | 15.100000  
2 | 198.760000 | 0 | 3 | semi-detached | 8.200000
```

Мал. 5 - Приклад запуску програми та перегляду меню

Перед використанням користувач повинен створити файл “apartments.json” у директорії, у яку було встановлено jar-файл застосунку.

2.3. Рекомендації користувачу

Після запуску програма виводить перелік доступних дій. Біля кожної дії вказано слово для запуску цієї функції. Наприклад, щоб виконати пошук по кількості кімнат, користувач повинен ввести “search_rooms”. Після чого потрібно буде ввести значення критерію для пошуку. Після завершення користувача буде автоматично повернено до меню.

Під час роботи можуть виникати повідомлення про помилки. Нижче наводиться їх перелік та опис:

1. “Invalid number format” - число введено некоректно (присутні заборонені символи)
2. “Invalid number value” - число не входить до дозволених значень.
Наприклад, номер поверху не може бути від’ємним.
3. “Failed to save apartments to file” - виникла помилка при запису даних у файл. Після двокрапки повинно наводитись більше інформації щодо причини. Можливими причинами є відсутність прав на запис у файл або відсутність вільного місця на диску.
4. “Failed to read apartments from file” - виникла помилка при читанні даних квартир з файлу. Після двокрапки повинно надаватись більше інформації щодо причини. Можливою причиною є відсутність файлу з даними: “apartments.json”
5. “Oops, nothing found...” - не є помилкою, але повідомляє що за заданим запитом не було нічого знайдено.

3 ВИХІДНИЙ КОД ЗАСТОСУВАННЯ

Вихідний код доступний у репозиторії, що розміщено на Github:

<https://github.com/nikitavbv/Assignments2ndYearOOP/tree/master/coursework>

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. <https://www.codecademy.com/articles/mvc>
2. <http://cr.openjdk.java.net/~iris/se/11/latestSpec/api/>
3. <http://cr.openjdk.java.net/~iris/se/11/latestSpec/api/java.base/java/util/ResourceBundle.html>
4. <http://cr.openjdk.java.net/~iris/se/11/latestSpec/api/java.base/java/util/Locale.html>
5. <http://cr.openjdk.java.net/~iris/se/11/latestSpec/api/java.base/java/util/HashMap.html>
6. <https://sites.google.com/site/gson/gson-user-guide>
7. <http://logging.apache.org/log4j/2.x/log4j-users-guide.pdf>