НАЦИОНАЛЬНЫЙ ТЕХНІЧНИЙ

УНИВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра автоматизованої обробки інформації та управління

КУРСОВА РОБОТА

по дисципліні

Об’єктно–орієнтоване програмування — 1

(назва дисципліни)

Варіант № 12

Виконав: студент групи ІП–82

Лучкін І. Д.

(Прізвище та ініціали)

Прийняла: доцент Іванова Л.М.

Київ – 2020

Зміст

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:

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

Номер варіанту відповідає номеру варіанту з лабораторних робіт.
Дана лабораторна робота виконана для варіанту №12.

Рейс: Пункт призначення, Номер рейсу, Тип літака, Час вильоту, Дні тижня, Номер літака. Конструктор, Методи доступу, Метод *toString()*	1. Отримати список рейсів до заданого пункту призначення. 2. Отримати список рейсів для заданого дня тижню. 3. Отримати список рейсів для заданого дня тижню, час вильоту яких пізніше заданого.

Опис застосування
Архітектура застосування

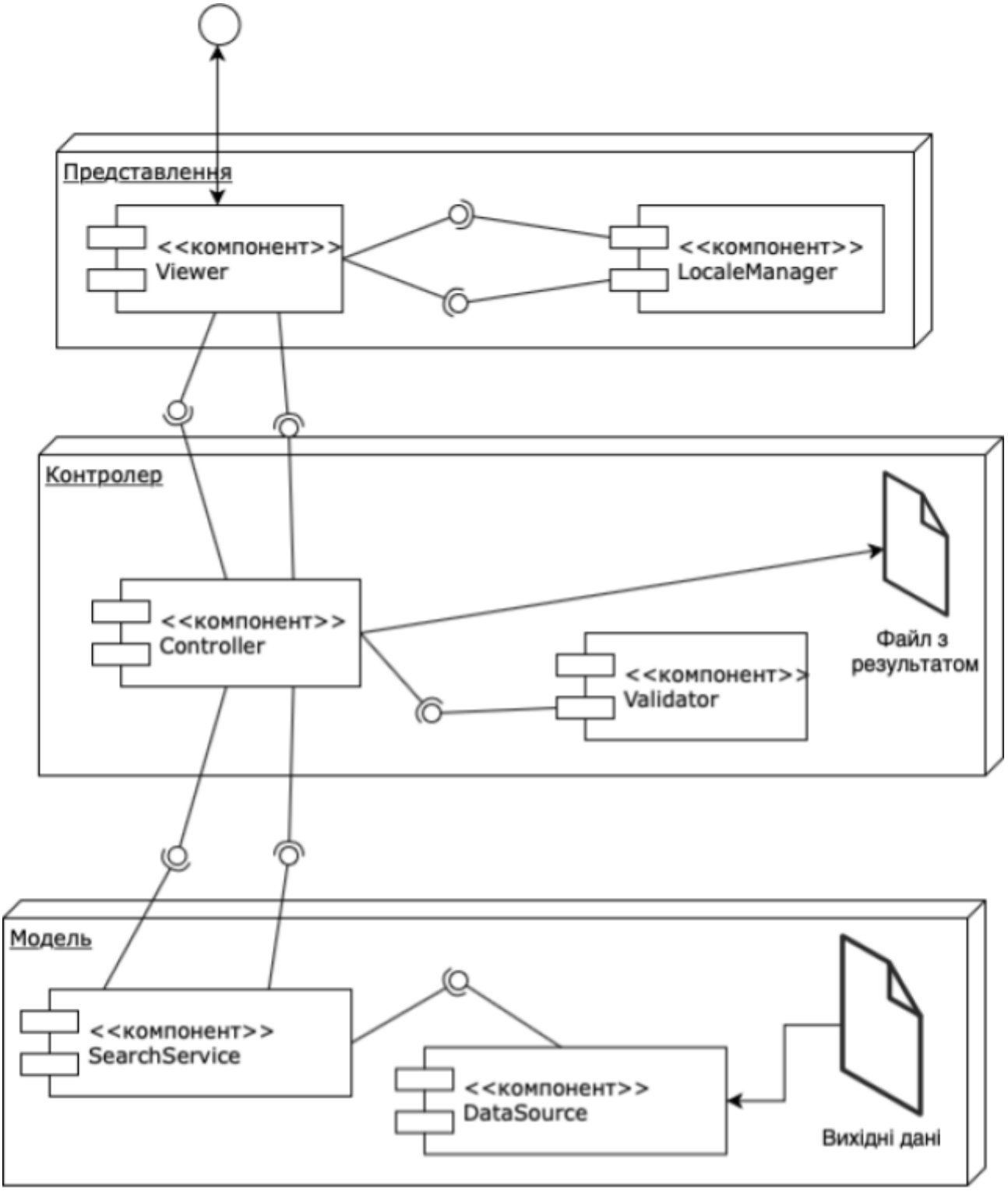

Рис. 1 – Архітектура застосування

Із самого початку користування програмою користувач працює з рівнем взаємодії. На цьому рівні існує компонент View, який реалізує форматоване

виведення результату роботи додатку, та компонент Controller, який реалізує користувацький інтерфейс та передачу команд користувача рівню логіки.

Після того, як Controller передав до рівня логіки користувацькі команди, компонент Search, який містить всі необхідні алгоритми, здійснює пошук по набору даних згідно із запитом користувача.

Після обробки вмісту компонента DataSource, який знаходиться на рівні даних, компонент View використовує власну функціональність для форматованого виводу результату роботи застосування.

На кожному рівні архітектури присутні обробки всіх можливих виключень, пов’язаних із даними, введеними користувачем, алгоритмами пошуку, читанням та записом у файл. Ці виключення описані на рівні логіки. Логування всіх подій, що виникають під час роботи програми, відбувається на кожному рівні.

Опис варіантів використання

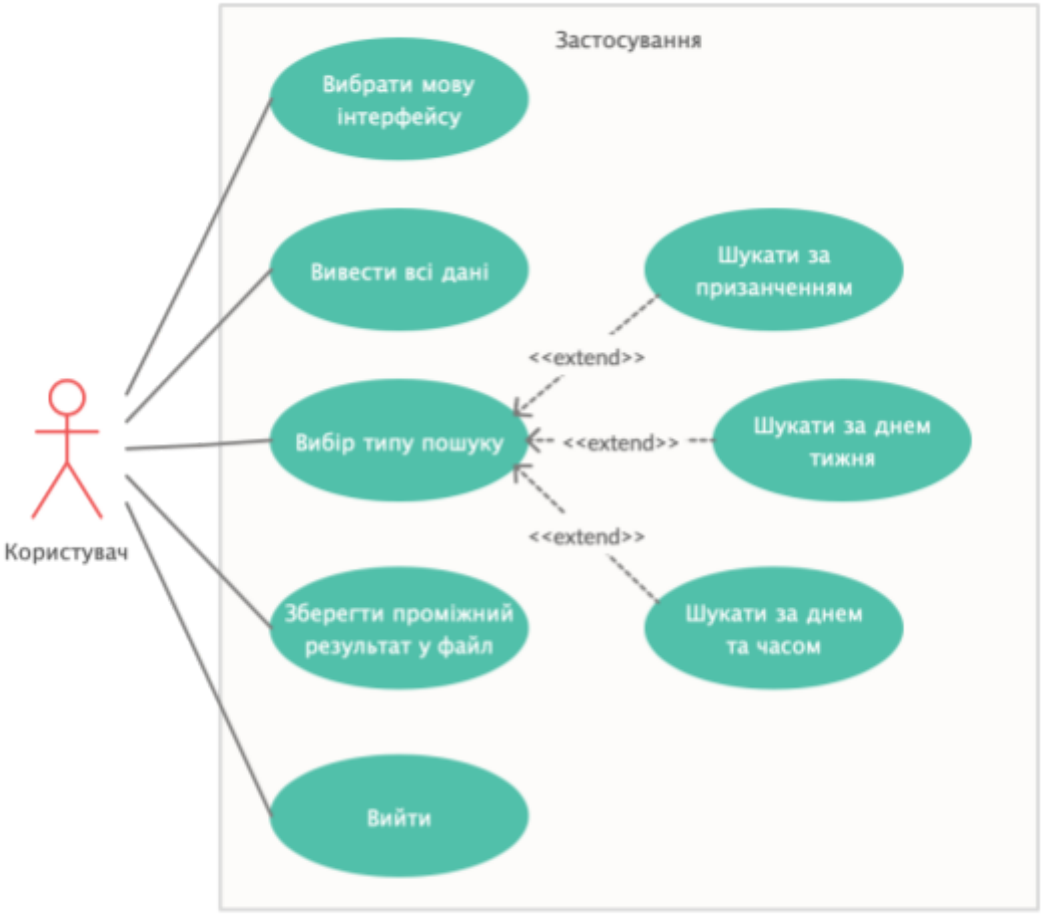

Рис. 2 – Діаграма варіантів використання

На старті роботи користувачу пропонується обрати мову інтерфейсу (англійська або українська). При цьому інтерфейс вибору написаний англійською мовою (за замовчуванням). Після вибору мови, користувач бачить власне локалізований інтерфейс застосування, де він може обрати необхідний параметр пошуку, зберегти проміжний результат у файл, вивести всі дані з первинного набору або вийти з програми.

Після виконання будь-якої дії, окрім виходу, користувачу пропонується такий самий інтерфейс. Таким чином, користувач може безперервно користуватися програмою та виходити з неї тільки тоді, коли програмі буде надана команда “Вихід”.

Користувач може здійснити пошук в початковому наборі за місцем призначення рейсу, за днем тижня, коли цей рейс відбувається (рейси є регулярними), або і за днем тижня, і за часом. Коли користувач обирає третій варіант параметру, результатом є всі рейси, які відбуваються в заданий день у заданий час **або** пізніше в цей же день.

Діаграма класів

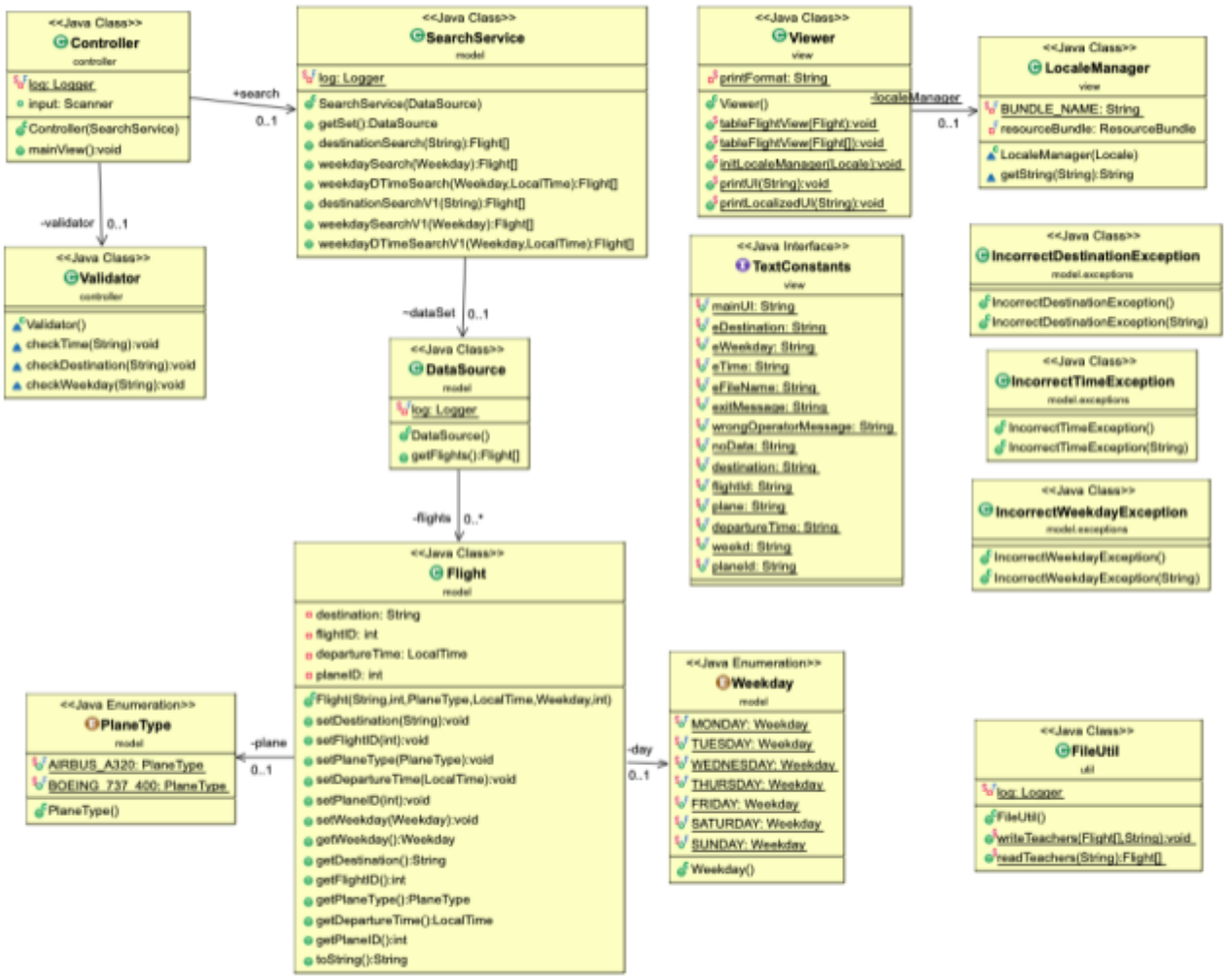

Рис. 2 – Діаграма класів

В пакеті view знаходяться класи Viewer, інтерфейс TextConstants та LocaleManager.

Клас Viewer відповідає за реалізацію форматованого виводу таблиць вихідних даних та локалізоване виведення елементів інтерфейсу. Елементи інтерфейсу та заголовки таблиць зберігаються в інтерфейсі TextConstants, який посилається на .properties файли з локалізованими текстовими константами. Для того, щоб виводити повідомлення, локалізовані під обрану користувачем мову, клас Viewer використовує клас LocaleManager.

Клас LocaleManager, в свою чергу, на основі заданого користувачем параметру (мовні налаштування) задає параметри локалізації для застосунку та реалізує інтерфейс для отримання відповідних текстових констант з різних .properties файлів, кожен з яких зберігає текстові константи окремої мови.

В пакеті controller знаходяться класи Controller та Validator.

Клас Controller відповідає за передачу параметрів, які вводить користувач, до логічного рівня застосунку, таким чином реалізуючи взаємодію із застосунком.

Клас Validator реалізує методи валідації введених користувачем даних для коректної роботи програми. Відповідно до введених даних використовуються різні методи валідації, а саме регулярні вирази для текстових значень та перевірка на належність до класів-перелічень.

В пакеті model знаходяться пакет exceptions, який містить всі необхідні виключення, які "викидаються" валідатором (за необхідності), класи DataSource, Flight, SearchService та нумератори (перелічення) Weekday і PlaneType.

Клас DataSource відповідає за збереження первинного набору даних, який складається із певної кількості об'єктів класу Flight.

Клас Flight, в свою чергу, реалізує основну сутність програми. Об'єкти цього класу містять всю інформацію про рейс згідно із завданням. Також поля класу plane та day використовують об'єкти перечислень PlaneType та Weekday відповідно для збереження відповідних даних.

Клас SearchService містить функціональність здійснення різних типів пошуку по набору даних. Фільтрація на основі введеного користувачем параметру може відбуватися за пунктом призначення рейсу, днем, коли він відбувається, або днем та часом відправлення.

В пакеті util знаходиться єдиний клас FileUtil, який реалізує функціональність запису та зчитування об'єктів класу Flight у вигляді масиву в файл або з файлу відповідно.

Опис методів

Таблиця 1. – Опис методів класів

№	Назва класу	Назва функції	Призначення функції	Опис вхідних параметрів	Опис вихідних параметрів
1	DataSource	DataSource	Створює об’єкт класу DataSource	—	void
2	DataSource	getFlights	Повертає набір даних з об’єкту класу DataSource	—	Масив об’єктів класу Flight
3	Flight	Flight	Створює об’єкт класу Flight	Параметри рейсу (відповідно до завдання)	void
4	Flight	setDestination	Встановлює об’єкту класу Flight нове значення поля *destination*	Місто призначення у форматі String	void
5	Flight	setFlightID	Встановлює об’єкту класу Flight нове значення поля *flightID*	Номер рейсу у форматі int	void
6	Flight	setPlaneType	Встановлює об’єкту класу Flight нове значення поля *plane*	Модель літака, що виконує рейс, типу PlaneType	void
7	Flight	setDepartureTime	Встановлює об’єкту класу Flight нове	Час відльоту у форматі	void

			значення поля *departureTime*	LocalTime	
8	Flight	setPlaneID	Встановлює об’єкту класу Flight нове значення поля *planeID*	Номер літака у форматі ing	void
9	Flight	setWeekday	Встановлює об’єкту класу Flight нове значення поля *weekday*	День тижня у форматі Weekday	void
10	Flight	getDestination	Встановлює об’єкту класу Flight нове значення поля *destination*	—	Місто призначення у форматі String
11	Flight	getFlightID	Повертає значення поля *flightID*	—	Номер рейсу у форматі int
12	Flight	getPlaneType	Повертає значення поля *plane*	—	Модель літака, що виконує рейс, типу PlaneType
13	Flight	getDepartureTime	Повертає значення поля *departureTime*	—	Час відльоту у форматі LocalTime
14	Flight	getPlaneID	Повертає значення поля *planeID*	—	Номер літака у форматі ing
15	Flight	getWeekday	Повертає значення поля *weekday*	—	День тижня у форматі

					Weekday
16	Flight	toString	Перетворює об’єкт класу Flight в рядок на основі інформації полів.	—	Форматована інформація з об’єкту Flight типу String
17	SearchService	SearchService	Створює новий об’єкт класу SearchService	Об’єкт класу DataSource	void
18	SearchService	getSet	Повертає значення поля dataSet	—	Об’єкт класу DataSource
19	SearchService	destinationSearch	Виконує фільтрацію набору даних на основі введеного міста призначення	Місто призначення у форматі String	Масив об’єктів класу Flight
20	SearchService	weekdaySearch	Виконує фільтрацію набору даних на основі введеного дня тижня	День тижня у форматі Weekday	Масив об’єктів класу Flight
21	SearchService	weekdayDTimeSearch	Виконує фільтрацію набору даних на основі введених дня тижня та часу відльоту	День тижня у форматі Weekday та час відльоту у форматі LocalTime	Масив об’єктів класу Flight
22	Controller	Controller	Створює об’єкт класу Controller	Об’єкт классу SearchServi	void

				ce	
23	Controller	mainView	Реалізує взаємодію користувача з інтерфейсом; передає користувацькі команди до SearchService	——	void
24	Validator	checkTime	Перевіряє правильність вводу часу	Введений час формату String	void
25	Validator	checkDestination	Перевіряє правильність вводу міста призначення	Введене місто призначення формату String	void
26	Validator	checkWeekday	Перевіряє правильність вводу дня тижня	Введений день тижня формату String	void
27	FileUtil	writeFlights	Записує результат пошуку у файл	Шлях до файлу у форматі String, масив об’єктів класу Flight	void
28	FileUtil	readFlights	Зчитує набір даних з файлу	Шлях до файлу у форматі String	Масив об’єктів класу Flight
29	LocaleManager	LocaleManager	Створює новий об’єкт класу LocaleManage	Об’єкт класу Locale	void

			r		
30	LocaleManager	getString	Повертає текстову константу відповідно до обраної мови (завдяки об’єкту Locale)	Ключ до текстової константи типу String	Текстова константа типу String
31	Viewer	tableFlightView	Виводить на екран форматовану інформацію з об’єкту класу Flight	Об’єкт класу Flight	void
32	Viewer	tableFlightView	Виводить на екран форматовану інформацію з масиву об’єктів класу Flight	Масив об’єктів класу Flight	void
33	Viewer	initLocaleManager	Створює об’єкт класу LocaleManager	Об’єкт класу Locale	void
34	Viewer	printUI	Виводить на екран надане рядкове значення	Значення типу String	void
35	Viewer	printLocalizedUI	Виводить на екран локалізоване рядкове значення	Значення типу String	void
36	IncorrectDestination	IncorrectDestinationException	Створює виключення	——	void

	Exception		типу IncorrectDestinationException		
37	IncorrectDestinationException	IncorrectDestinationException	Створює виключення типу IncorrectDestinationException	Повідомлення типу String	void
38	IncorrectTimeException	IncorrectTimeException	Створює виключення типу IncorrectTimeException	—	void
39	IncorrectTimeException	IncorrectTimeException	Створює виключення типу IncorrectTimeException	Повідомлення типу String	void
40	IncorrectWeekdayException	IncorrectWeekdayException	Створює виключення типу IncorrectWeekdayException	—	void
41	IncorrectWeekdayException	IncorrectWeekdayException	Створює виключення типу IncorrectWeekdayException	Повідомлення типу String	void

Рекомендації щодо використання застосування
Технічна характеристика

Застосування може працювати на будь-якому комп’ютері

- macOS X
- Windows (будь-яка версія 64-розрядної системи)
- Linux (будь-який дистрибутив Debian або RedHat)

із встановленим JDK 13+ версії. Використовування програмного забезпечення можливе лише через інтерфейс командного рядку.

Інструкції щодо використання

Для того, щоб користуватися застосунком, необхідно:

- Завантажити .jar файл за посиланням;
- Перейти до папки Downloads у командному рядку;
- Виконати команду *java -jar java-track.jar.*

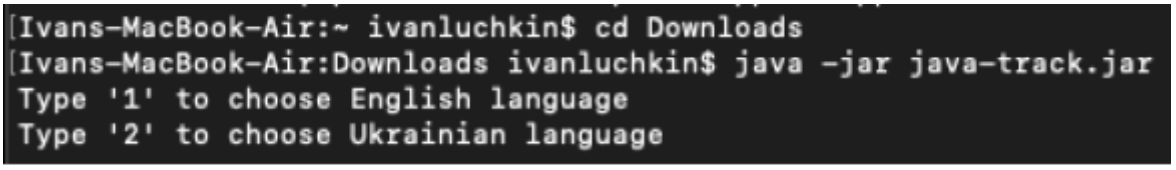

Рис. 3 – Візуалізація запуску програми

Застосунок надає можливість здійснювати пошук по базі даних з авіарейсами.

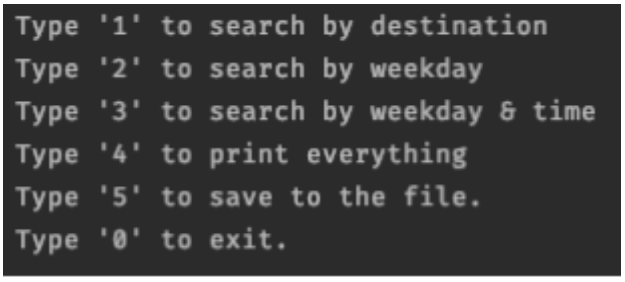

Рис. 4 – Головне меню

Destination	Flight ID	Plane	Departure time	Weekday	Plane ID
Kyiv	1111	AIRBUS_A320	15:00	FRIDAY	103
Kyiv	1112	BOEING_737_400	15:30	TUESDAY	102
Odesa	1113	BOEING_737_400	16:00	SATURDAY	103
Kharkiv	1114	AIRBUS_A320	17:05	MONDAY	101
Krakow	2123	BOEING_737_400	09:00	THURSDAY	204
Berlin	5562	BOEING_737_400	10:25	FRIDAY	205
London	3271	AIRBUS_A320	05:50	SUNDAY	206
Sydney	4486	AIRBUS_A320	00:00	SUNDAY	207
Tokyo	9937	AIRBUS_A320	23:10	WEDNESDAY	208
San-Diego	6112	BOEING_737_400	12:45	WEDNESDAY	209

Рис. 5 – Всі вихідні дані у табличному вигляді

Пошук може здійснюватися за містом призначення, днем вильоту або днем вильоту і часом вильоту одночасно. Після кожного пошуку користувач має можливість здійснити новий пошук за іншим або тим же параметром, вивести повний набір даних з бази, зберегти останній результат у файл або вийти з програми.

```
Enter destination:
Kyiv
```

Destination	Flight ID	Plane	Departure time	Weekday	Plane ID
Kyiv	1111	AIRBUS_A320	15:00	FRIDAY	103
Kyiv	1112	BOEING_737_400	15:30	TUESDAY	102

Рис. 6 – Приклад пошуку за містом призначення

```
Enter weekday:
FRIDAY
```

Destination	Flight ID	Plane	Departure time	Weekday	Plane ID
Kyiv	1111	AIRBUS_A320	15:00	FRIDAY	103
Berlin	5562	BOEING_737_400	10:25	FRIDAY	205

Рис. 7 – Приклад пошуку за днем тижня

```
Enter weekday:
WEDNESDAY
Enter departure time:
15:00
```

Destination	Flight ID	Plane	Departure time	Weekday	Plane ID
Tokyo	9937	AIRBUS_A320	23:10	WEDNESDAY	208

Рис. 8 – Приклад пошуку за днем тижня та часом

У випадку некоректного введення даних для пошуку програма надасть користувачу повідомлення про невірне введення та повернеться до головного меню.

Рекомендації користувачеві

- При введенні міста призначення коректна робота програми можлива тільки тоді, коли введене значення починається з великої літери латинського алфавіту, а власне значення не містить цифр та інших символів, крім літер.
- Введення часу вильоту можливе лише у 24-х годинному форматі “ГГ:ХХ”. При цьому значення 00:00 є коректним та прирівнюється до значення 24:00, яке не є коректним.
- День тижня необхідно вводити англійською мовою, використовуючи лише літери верхнього регістру
- Дані записуються та зчитуються у файл/з файлу лише у вигляді об’єктів, не у вигляді тексту.
- Можливий запис у файли будь-якого формату.
- Застосунок підтримує англомовну та україномовну локалізації.

Вихідний код застосування

Посилання на репозиторій GitHub

Список використаної літератури

- Паттерн MVC — https://techterms.com/definition/mvc
- Клас Arrays – https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Arrays.html
- Мануал з логування – https://logging.apache.org/log4j/1.2/manual.html
- Клас PrintStream – https://docs.oracle.com/javase/7/docs/api/java/io/PrintStream.html
- Клас ResourceBundle – https://docs.oracle.com/javase/7/docs/api/java/util/ResourceBundle.html