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 - 2*

(назва дисципліни)

Варіант №\_19\_

Виконав: студент групи \_\_\_\_ІП-81\_\_\_\_\_\_

\_\_\_\_\_Савастру Станіслав\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Прізвище та ініціали)

Прийняла: доцент Іванова Л.М.

Київ-2020

Зміст

[**1.** **Постановка задачі** 3](#_Toc42035102)

[**2.** **Опис застосування** 4](#_Toc42035103)

[**2.1.** **Архітектура застосування** 4](#_Toc42035104)

[**2.2.** **Діаграма варіантів використання** 5](#_Toc42035105)

[**2.3.** **Діаграма класів та таблиці** 6](#_Toc42035106)

[**3.** **Рекомендації щодо використання застосування** 16](#_Toc42035107)

[**3.1.** **Технічна характеристика** 16](#_Toc42035108)

[**3.2.** **Рекомендації по встановленню та налаштуванню** 17](#_Toc42035109)

[**3.3.** **Рекомендації користувачеві** 17](#_Toc42035110)

[**4.** **вихідний код застосування** 19](#_Toc42035111)

[**Перелік посилань** 20](#_Toc42035112)

# **Постановка задачі**

Написати консольне застосування за архітектурним шаблоном MVC з розподіленням відповідальності між компонентами, яке виконує обробку набору даних згідно з варіантом 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 завершені 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.

Варіант: 19

Опис класу: АвіаТур: Назва, Туроператор, Пункти відвідування, Ціна, Кількість місць (вільних та зайнятих), Дата від’їзду. Конструктор, Методи доступу, Метод toString().

Критерій для виведення інформації:

1. Отримати список авіатурів заданого туроператора.
2. Отримати список авіатурів по заданому пункту відвідування.
3. Отримати список авіатурів по заданій назві, до вказаної дати та наявності вільних місць

# **Опис застосування**

## **Архітектура застосування**

Консольний додаток виконаний за архітектурою MVC (Модель-представлення-контроллер). Згідно з нею застосування має поділятися на 3 шари: Model, View, Controller. Модель відображає поведінку застосунку незалежно від інтерфейсу користувача. Вона керує логікою роботи з даними у застосунку. Представлення відповідає за інтерфейс, з яким безпосередньо взаємодіє користувач. Контролер зв’язує між собою представлення та модель.

Архітектуру нашого застосунку ми поділили на 3 шари, кожен з яких один або декілька компонентів (Рис. 2.1). Розглянемо кожний компонент більш детально (Табл. 2.1).

Табл. 2.1 – Опис компонентів застосунку

|  |  |  |
| --- | --- | --- |
| Шар | Компонент | Функції |
| Представлення | Представлення | Відповідає за відображення інформації користувачу. Взаємодіє з компонентом Locale manager, щоб отримати текст повідомлень відповідно до обраної користувачем мови. Передає запит користувача до шару контролера та отримує від нього дані для відображення користувачу |
| Сканер команд | Відповідає за читання даних, що вводить користувач, та за перевірку правильності вводу. Передає компоненту View дані, отримані від користувача. |
| Мовний менеджер | Надає текст повідомлень на обраній користувачем мові за запитом |
| Контролер | Контролер | Відповідає за отримання запиту від з шару View та його передачу до шару Model. Відповідає за отримання даних за запитом від з шару Model та їх передачу до шару View. |
| Модель | Сервіс бази даних | Отримує запити від контролера та повертає йому дані згідно до запиту. Взаємодіє з компонентом File IO: передає йому дані для запису у файл або отримує від нього дані з файлу. |
| Вивід з файлу та вивід до файлу | Відповідає за читання даних з файлу та за запис даних до файлу. |

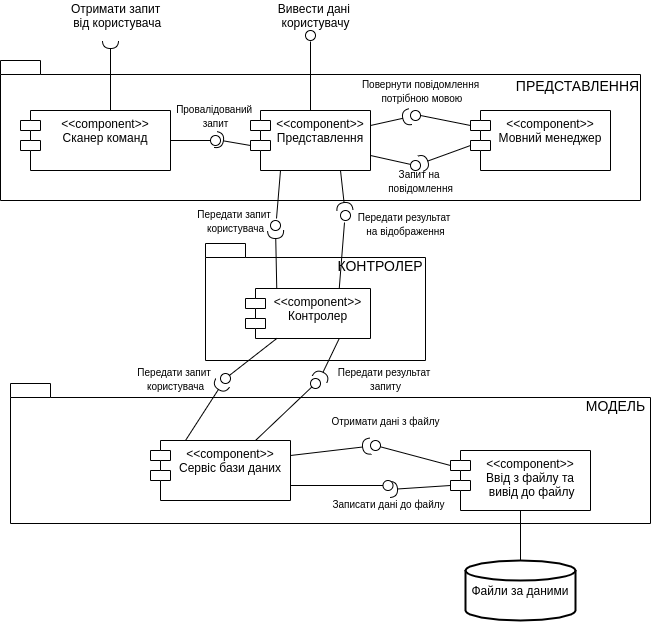


Рис. 2.1 – Діаграма компонентів застосунку

## **Діаграма варіантів використання**

Діаграма варіантів використання наведена на рисунку 2.2. Детальний опис варіантів використання міститься у таблиці 2.2.

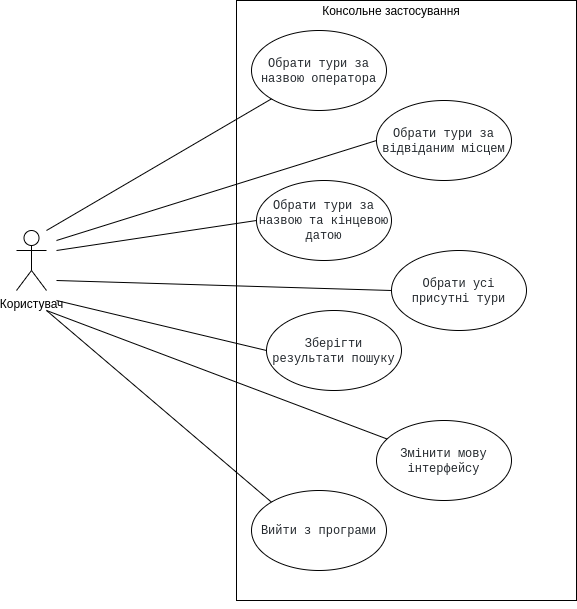


Рис. 2.2 – Діаграма варіантів використання

Табл. 2.2 – Опис варіантів використання

|  |  |  |
| --- | --- | --- |
| Актор | Варіант використання | Опис |
| Користувач | Обрати тури за назвою оператора | 1. Користувач обирає відповідний пункт у меню. 2. Користувач вводить назву потрібного оператора. |
| Обрати тури за відвіданим місцем | 1. Користувач обирає відповідний пункт у меню. 2. Користувач вводить відвіданого місця. |
| Обрати тури за назвою та кінцевою датою | 1. Користувач обирає відповідний пункт у меню. 2. Користувач вводить назву туру. 3. Користувач вводить кінцеву дату відліту |
| Обрати усі присутні тури | 1. Користувач обирає відповідний пункт у меню. |
| Зберігльтати пошукути резу | 1. Користувач обирає відповідний пункт у меню. 2. Користувач вводить шлях до файлу, в який хоче зберігти результат пошуку |
| Змінити мову інтерфейсу | 1. Користувач обирає відповідний пункт у меню. 2. Користувач обирає потрібну мову інтерфейсу з присутніх |
| Вийти з програми | 1. Користувач обирає відповідний пункт меню. |

## **Діаграма класів та таблиці**

Діаграма класів нашого застосунку Зображена на рисунку 2.3. Опис класів та їх методів міститься у таблиці 2.3.

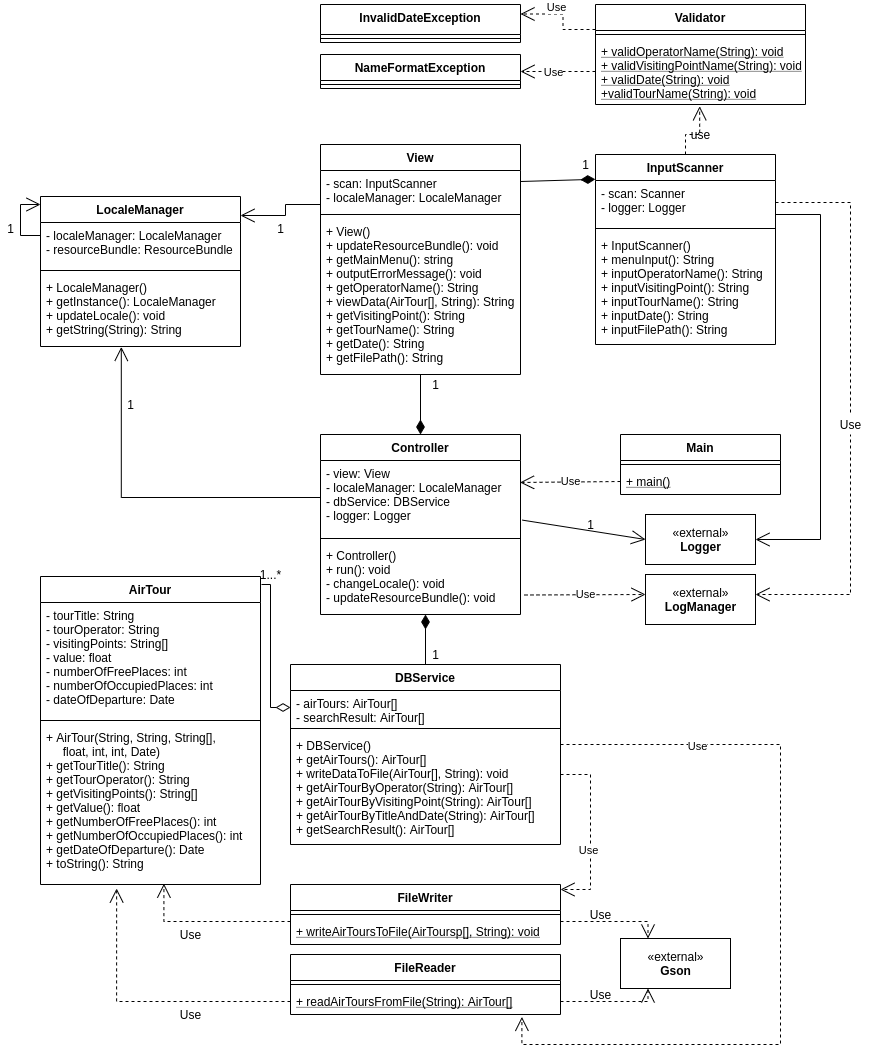


Рис. 2.3 – Діаграма класів застосунку

Табл. 2.3 — Опис класів застосування

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 1 | Main | main | Вхідна точка застосунку. Створює об’єкт класу контролера та викликає його метод run() | - | - |
| 2 | Controller | Controller | Конструктор. Ініціалізує поля екземпляру класу. | - | - |
| 3 | run | Обробляє запити, що надходять від View та напрявляє їх до DBService. Повертає дані отримані від DBService до View. | - | - |
| 4 | changeLocale | Обробляє запит на зміну мови інтерфейсу | - | - |
| 5 | updateResourceBundle | Оновлює мову інтерфейсу | - | - |
| 6 | View | View | Конструктор. Ініціалізує поля екземпляру класу | - | - |
| 7 | updateResourceBundle | Оновлює мову інтерфейсу | - | - |
| 8 | getMainMenu | Відображає головне меню. Визиває метод класу InputScanner для зчитування обраного пункту меню. Повертає обраний пункт | - | String – обраний пункт меню |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 9 | View | outputErrorMessage | Виводить повідомлення про помилку | - | - |
| 10 | getOperatorName | Відображає пропозицію ввести назву оператора Визиває метод класу InputScanner для зчитування обраного назви оператора. Повертає назву | - | String – назва обраного оператора |
| 11 | getVisitingPoint | Відображає пропозицію ввести назву відвіданого місця. Визиває метод класу InputScanner для зчитування обраного місця. Повертає його назву | - | String – назва обраного місця призначення |
| 12 | getTourName | Відображає пропозицію ввести назву туру. Визиває метод класу InputScanner для зчитування назви туру. Повертає його назву | - | String – назва обраного туру |
| 13 | getDate | Відображає пропозицію ввести кінцеву датутуру. Визиває метод класу InputScanner для зчитування дати. Повертає дату | - | Date – обрана дата |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 14 | View | getFilePath | Відображає пропозицію ввести шлях до файлу. Визиває метод класу InputScanner для зчитування шляху до файлу. Повертає шлях до файлу | - | String – введений шлях до файлу |
| 15 | changeLanguageMenu | Відображає меню зміни мови. Визиває метод класу InputScanner для зчитування обраної мови. Повертає обраний пункт меню | - | String |
| 16 | viewData | Відображає авіатури користувачу | AirTour[]-  Масив авіатурів,  String –  Додаткове повідомлення для виводу користувачу | - |
| 17 | outputExceptionMessage | Відображає повідомлення про помилку | String –  Повідомлення помилки | - |
| 18 | InputScanner | InputScanner | Конструктор. Ініціалізує поля екземпляру класу | - | - |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 19 | InputScanner | menuInput | Зчитує введені з клавіатури користувачем дані про обраний пункт меню | - | String –  результат введення користувача |
| 20 | inputOperatorName | Зчитує введену з клавіатури користувачем назву оператора. Валідує за допомогою методу класу Validator | - | String –  назва оператора |
| 21 | inputVisitingPoint | Зчитує введену з клавіатури користувачем назву місця. Валідує за допомогою методу класу Validator | - | String –  назва місця |
| 22 | inputTourName | Зчитує введену з клавіатури користувачем назву туру. Валідує за допомогою методу класу Validator | - | String –  назва туру |
| 23 | inputDate | Зчитує введену з клавіатури користувачем дату. Валідує за допомогою методу класу Validator | - | Date –введена дата |
| 24 | inputFilePath | Зчитує введений з клавіатури користувачем шлях до файлу. Валідує за допомогою методу класу Validator | - | Strring – введений шлях до файлу |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 25 | Validator | validOperatorName | Валідує назву оператора. Викликає Exception, якщо невалідний ввід | String –  Назва оператора для валідації | - |
| 26 | validVisitingPointName | Валідує назву назву місця. Викликає Exception, якщо невалідний ввід | String –  Назва місця | - |
| 27 | validDate | Валідує дату. Викликає Exception, якщо невалідний ввід | String –  Дата для валідації | - |
| 28 | validTourName | Валідує назву туру. Викликає Exception, якщо невалідний ввід | String –  Назва туру для валідації | - |
| 29 | LocaleManager | LocaleManager | Приватний конструктор. Ініціалізує поля екземпляру класу | - | - |
| 30 | getInstance | Повертає екземпляр классу LocaleManager, що зберігається у приватному статичному полі. Якщо його не існує, то створює, записує до того статичного поля та повертає. | - | LocaleManager – екземпляр класу LocaleManager |
| 31 | updateLocale | Оновлює мову інтерфейсу | - | - |
| 32 | getString | Повертає потрібне повідомлення поточною мовою інтерфейсу | String – ключ-назва повідомлення | String – повідомлення поточною мовою |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 33 | AirTour | AirTour | Конструктор. Ініціалізує поля екземпляру класу | String –  назва туру,  String –  назва оператора,  String[] –  масив місць відвідування,  float – ціна туру,  int – к-ть вільних місць,  int – к-ть зайнятих місць,  Date – дата відльоту | - |
| 34 | getTourTitle | Повертає назву туру | - | String –  назва туру |
| 35 | getTourOperator | Повертає назву оператора | - | String –  назва оператора |
| 36 | getVisitingPoints | Повертає місця відвідування | - | String[] – місця відвідування |
| 37 | getValue | Повертає ціну туру | - | float – ціна туру |
| 38 | getNumberOfFreePlaces | Повертає к-ть вільних місць | - | int – к-ть вільних місць |
| 39 | getNumberOfOccupiedPlaces | Повертає к-ть зайнятих місць | - | int – к-ть зайнятих місць |
| 40 | getDate | Повертає дату відправлення | - | Date – дата відправлення |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 41 | AirTour | toString | Повертає текстове представлення об’єкту класу у відформатованому вигляді | - | String - текстове представлення об’єкту класу у відформатованому вигляді |
| 42 | DBService | DBService | Конструктор. Ініціалізує поля екземпляру класу | - | - |
| 43 | getAirTours | Повертає всі присутні авіатури | - | AitTour[] –  всі присутні авіатури |
| 44 | writeDataToFile | Записує дані до файлу | AirTours[] – авіатури для запису,  String – шлях до файлу | - |
| 45 | getAirTourByOperator | Повертає авіатури обраного оператора | String – ім’я оператора | AitTour[] –  обрані авіатури |
| 46 | getAirTourByVisitingPoing | Повертає авіатури з обраним місцем відвідування | String – місце відвідування | AitTour[] –  обрані авіатури |
| 47 | getAirTourByTitleAndDate | Повертає авіатури з обраною назвою та кінцевим терміном відльоту | String – назва туру,  Date – кінцева дата | AitTour[] –  обрані авіатури |
| 48 | getSearchResult | Повертає результати останнього пошуку | - | AitTour[] –  об результати останнього пошуку |

Продовження таблиці 2.3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Назва класу** | **Назва методу** | **Призначення функції** | **Опис вхідних параметрів** | **Опис вихідних параметрів** |
| 49 | FileWriter | writeAirToursToFile | Записує авіа тури до файлу у форматі JSON | AirTour[] – авіатури,  String – шлях до файлу | - |
| 50 | FileReader | readAirToursFromFile | Зчитує авіа тури з файлу у форматі JSON | String – шлях до файлу | AirTour[] – авіатури |

# **Рекомендації щодо використання застосування**

## **Технічна характеристика**

Для користування даним застосунком потрібно мати встановлену JDK 11. Тому комп'ютер користувача повинен задовільняти системним вимогам JDK 11 відповідно до встановленої операційної системи.

Windows

- Windows 10 (8u51 та вище)

- Windows 8.x

- Windows 7 з пакетом оновлення 1 (SP1)

- Windows Vista SP2

- Windows Server 2008 R2 з пакетом оновлення 1 (SP1) (64-розрядна версія)

- Windows Server 2012 и 2012 R2 (64-розрядна версія)

- RAM: 128 МБ

- Простір на диску: 124 МБ для JRE; 2 МБ для оновлення Java

- Процесор: мінімальна вимога - Pentium 2 266 МГц

- Браузери: Internet Explorer 9 та вище, Firefox

Mac OS X

- Mac на базі процесора Intel під управлінням Mac OS X 10.8.3+, 10.9+

- Повноваження адміністратора для установки

- 64-бітний браузер. Для запуска Oracle Java для Mac вимагається 64-розрядний браузер (наприклад, Safari).

Linux

- Oracle Linux 5.5+

- Oracle Linux 6.x (32-розрядна версія), 6.x (64-розрядна версія)

- Oracle Linux 7.x (64-розрядна версія) (8u20 або більш пізня)

- Red Hat Enterprise Linux 5.5+1, 6.x (32-розрядна версія), 6.x (64-розрядна версія)

- Red Hat Enterprise Linux 7.x (64-розрядна версія) (8u20 або більш пізня)

- Suse Linux Enterprise Server 10 SP2+, 11.x

- Suse Linux Enterprise Server 12.x (64-розрядна версія) (8u31 або більш пізня)

- Ubuntu Linux 12.04 LTS, 13.x

- Ubuntu Linux 14.x (8u25 або більш пізня)

- Ubuntu Linux 15.04 (8u45 або більш пізня)

- Ubuntu Linux 15.10 (8u65 або більш пізня)

- Браузери: Firefox

## **Рекомендації по встановленню та налаштуванню**

Для користування даним застосунком користувач повинен завантажити jar файл додатку та розмістити його у потрібній директорії. Якщо у користувача не встановлена JDK 11 (первірити це можна за допомогою команди у консолі операційної системи *java -version* ), то він повинен її завантажити та встановити.

Після цих кроків користувач може запустити додаток на виконання. Для цього користувач повинен перейти до директорії з файлом застосунку, відкрити там командний рядок та ввести команду *java -jar coursework.jar.* Далі користувач повинен користуватися інструкціями, що надає програма під час виконання.

Файл з вихідними даними потрібно розташувати у тій самій директорії, що й файл додатку та назвати “airTours.json”. Дані у файлі повинні бути записані у форматі JSON. Параметри кожного туру повинні бути записані у такому порядку:

1. Назва туру
2. Назва оператора
3. Місця відвідування
4. Ціна
5. Кількість вільних місць
6. Кількість зайнятих місць
7. Час відправлення

## **Рекомендації користувачеві**

Після запуску застосунку на екран виводиться меню. Відповідно до того, що хоче зробити користувач він обирає потрібний пункт меню введенням номеру меню та натисканням клавіши Enter.

Головне меню програми має таки пункти:

1. Обрати тури за ім’ям оператора

2. Обрати тури за відвідуваним місцем

3. За навою туру та кінцевою датою

4. Обрати усі доступні тури

5. Зберігти проміжні результати

6. Змінити мову інтерфейсу

7. Вийти

Якщо користувач обирає пункти 1-3, то після вищевказаних дій він повинен ввести дані, за якими відбудеться фільтрування відповідно до обраного пункту. Після цього оброблені дані виводяться на екран.

Якщо користувач обирає пункт 4, то йомувиводяться усі доступні тури на екран.

Якщо користувач обирає пункт 5, то йому пропонується ввести шлях до файлу, у який зберігти проміжні результати пошуку.

Якщо користувач обирає пункт 6, то йому пропонується змінити мову інтерфейсу.

Якщо користувач обирає пункт 7, то виконання програми завершується.

Після закінчення роботи з застосунком останній результати пошуку зберігається, тому потім користувач може продовжити роботу з того моменту, на якому закінчив. Також проміжні результати можна зберегти вручну. Для цього потрібно обрати відповідний пункт меню та ввести шлях до місця зберігання цього файлу. Дані також зберігаються у форматі JSON.

Під час роботи програми виконується логування подій до окремого файлу. Директорія з файлом логування знаходиться у директорії, в якій знаходиться файл з програмою.

# **В****ихідний код застосування**

Посилання на GitHub репозиторій: https://github.com/SavStanis/KPIJavaTrack

# **Перелік посилань**

1. Документація Oracle [Електронний ресурс]:

<https://docs.oracle.com/javase>

1. <https://github.com/google/gson>
2. <https://logging.apache.org/log4j/2.x/>