МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА САПР

Практическая работа №1

по дисциплине

«Объектно-ориентированное программирование»

Тема: «Консольная программа для поиска в Википедии»

Кирсова В.В.
Студентки гр. 4354 _____ Коробцова Д.А.

Преподаватель _____ Кулагин М.В.

Санкт-Петербург
2025

1. Задание

Требуется разработать программу, которая с консоли считывает поисковый запрос пользователя, и выводит результат поиска по Википедии. После выбора нужной статьи программа должна открывать ее в браузере. Программа должна реагировать корректно на любой пользовательский ввод.

2. Спецификация программы

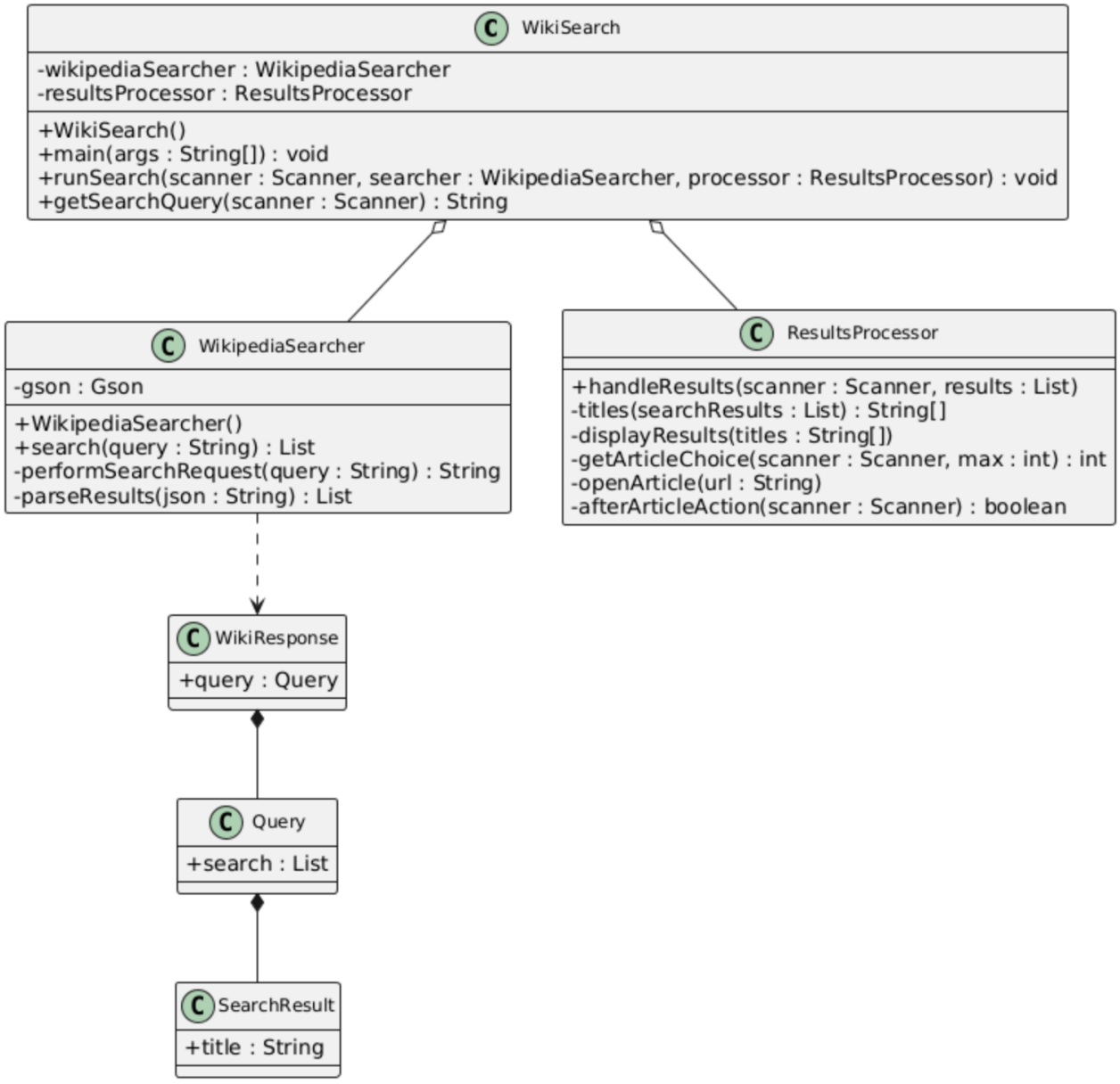

Рис. 1 – Диаграмма классов

3. Описание интерфейса пользователя программы

Outp1 = “Введите поисковый запрос (или 0 для выхода):”

Inp1 = любая строка текста(или 0)

Outp2 = “Выход.”

Outp3 = “Пустой запрос. Попробуйте снова.”

Outp3 = “Ошибка сети: <сообщение>”

Outp4 = “Ничего не найдено”

Outp5 = “Ошибка при подключении к Википедии. Код: <код_ответа>”

Outp6 = “Результаты поиска:

1. ____

2. ____

3. ____

…

N. ___ (N≤10)

Outp7 = “Введите номер статьи для открытия (0 - новый поиск):”

Inp2 = “Число от 0 до N”

Outp8 = “Нет такой статьи. Попробуйте снова.”

Outp9 = “Некорректный ввод. Попробуйте снова.”

Outp10 = “Открываю статью: <ссылка на статью>”

Outp11 = “Некорректный URL: <сообщение>”

Outp12 = “Ошибка при открытии браузера: <сообщение>”

Outp13 = “Открытие браузера не поддерживается.”

Outp14 = “Ошибка при кодировании URL: <сообщение>”

Outp15 = “Остаться в этом поиске (1), сделать новый поиск (2), выйти (0):”

Inp3 = “Число от 0 до 2”

4. Текст программы

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.URLEncoder;
import java.net.URL;
import java.net.URI;
import java.net.URISyntaxException;
import java.awt.Desktop;
import java.util.List;
import java.util.Scanner;
import com.google.gson.Gson;

// классы для парсинга JSON через Gson
class WikiResponse {
    Query query;
}

class Query {
    List<SearchResult> search;
}

class SearchResult {
    String title;
```

```
}


public class WikiSearch {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        WikipediaSearcher searcher = new WikipediaSearcher();
        ResultsProcessor processor = new ResultsProcessor();


        try {
            runSearch(scanner, searcher, processor);
        } catch (IOException e) {
            System.out.println("Ошибка сети: " + e.getMessage());
        } finally {
            scanner.close();
        }
    }


    private static void runSearch(Scanner scanner,  WikipediaSearcher searcher, ResultsProcessor
        processor) throws IOException {
        while (true) {
            String query = getSearchQuery(scanner);
            if (query == null) break;
            if (query.isEmpty()) continue;


            List<SearchResult> searchResults = searcher.search(query);
            if (searchResults == null || searchResults.isEmpty()) {
                System.out.println("Ничего не найдено.");
                continue;
            }
```

```
            processor.handleResults(scanner, searchResults);
        }
    }

    private static String getSearchQuery(Scanner scanner) {
        System.out.print("\nВведите поисковый запрос (или 0 для выхода): ");
        String query = scanner.nextLine().trim();

        if (query.equals("0")) {
            System.out.println("Выход.");
            return null;
        }

        if (query.isEmpty()) {
            System.out.println("Пустой запрос. Попробуйте снова.");
            return "";
        }

        return query;
    }
}
// класс для работы с Wikipedia API
class WikipediaSearcher {
    private Gson gson = new Gson();

    public List<SearchResult> search(String query) throws IOException {
        String jsonResponse = performSearchRequest(query);
        return parseResults(jsonResponse);
    }
```

```
private String performSearchRequest(String query) throws IOException {
    String apiUrl = "https://ru.wikipedia.org/w/api.php?action=query&list=search&srsearch="
            + URLEncoder.encode(query, "UTF-8")
            + "&utf8=&format=json";

    URL url = new URL(apiUrl);
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setRequestMethod("GET");
    conn.setRequestProperty("User-Agent", "WikiSearchApp/1.0 (https://example.com/)");

    int responseCode = conn.getResponseCode();
    if (responseCode != 200) {
        System.out.println("Ошибка при подключении к Википедии. Код: " + responseCode);
        return null;
    }

    BufferedReader in = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-
    8"));
    StringBuilder response = new StringBuilder();
    String line;
    while ((line = in.readLine()) != null) {
        response.append(line);
    }
    in.close();

    return response.toString();
}
private List<SearchResult> parseResults(String json) {
    if (json == null) return null;
```

```
        WikiResponse wikiResponse = gson.fromJson(json, WikiResponse.class);

        return (wikiResponse.query != null && wikiResponse.query.search != null)
                ? wikiResponse.query.search
                : null;
    }
}

// класс для управления результатами поиска и связи с пользователем
class ResultsProcessor {
    private String[] titles(List<SearchResult> searchResults) {
        int count = Math.min(searchResults.size(), 10);
        String[] titles = new String[count];
        for (int i = 0; i < count; i++) {
            titles[i] = searchResults.get(i).title;
        }
        return titles;
    }
    public void handleResults(Scanner scanner, List<SearchResult> searchResults) {
        String[] titles = titles(searchResults);
        boolean stayInThisSearch = true;

        while (stayInThisSearch) {
            displayResults(titles);

            int choice = getArticleChoice(scanner, titles.length);
            if (choice == 0) {
                break;
```

```
        }

        if (choice > 0) {
            openArticle(titles[choice - 1]);
            stayInThisSearch = afterArticleAction(scanner);
        }
    }
}
private void displayResults(String[] titles) {
    System.out.println("\nРезультаты поиска:\n");
    for (int i = 0; i < titles.length; i++) {
        System.out.println((i + 1) + ". " + titles[i]);
    }
}

private int getArticleChoice(Scanner scanner, int maxChoice) {
    System.out.print("\nВведите номер статьи для открытия (0 - новый поиск): ");
    String choiceStr = scanner.nextLine().trim();
    try {
        int choice = Integer.parseInt(choiceStr);
        if (choice >= 0 && choice <= maxChoice) {
            return choice;
        } else {
            System.out.println("Нет такой статьи. Попробуйте снова.");
            return -1;
        }
    } catch (NumberFormatException e) {
        System.out.println("Некорректный ввод. Попробуйте снова.");
        return -1;
```

```
    }
}
private void openArticle(String title) {
    try {
        String article = title.replace(" ", "_");
        String articleUrl = "https://ru.wikipedia.org/wiki/" + URLEncoder.encode(article, "UTF-8");

        System.out.println("Открываю статью: " + articleUrl);

        if (Desktop.isDesktopSupported()) {
            try {
                Desktop.getDesktop().browse(new URI(articleUrl));
            } catch (URISyntaxException e) {
                System.out.println("Некорректный URL: " + e.getMessage());
            } catch (IOException e) {
                System.out.println("Ошибка при открытии браузера: " + e.getMessage());
            }
        } else {
            System.out.println("Открытие браузера не поддерживается.");
        }
    } catch (IOException e) {
        System.out.println("Ошибка при кодировании URL: " + e.getMessage());
    }
}

private boolean afterArticleAction(Scanner scanner) {
    while (true) {
        System.out.print("\nОстаться в этом поиске (1), сделать новый поиск (2), выйти (0): ");
        String action = scanner.nextLine().trim();
```

```
            if (action.equals("1")) {
                return true;
            } else if (action.equals("2")) {
                return false;
            } else if (action.equals("0")) {
                System.out.println("Выход.");
                System.exit(0);
            } else {
                System.out.println("Некорректный ввод. Попробуйте снова.");
            }
        }
    }
}
```

Примеры работы:

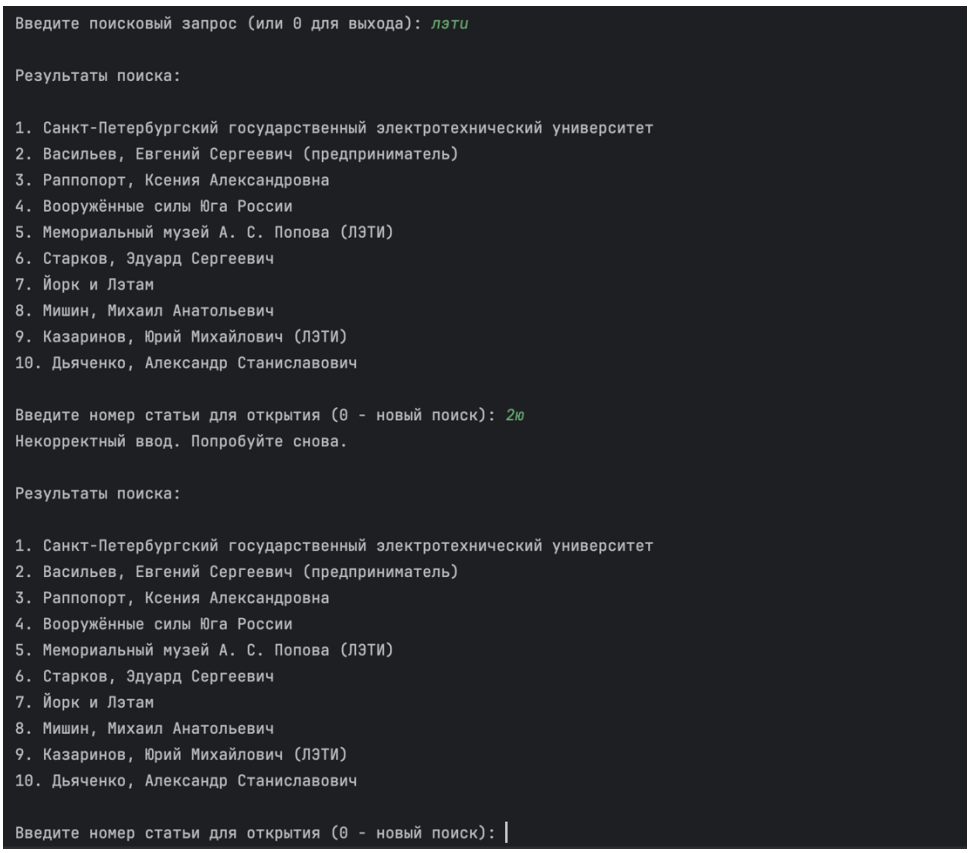

Рис. 2 – Пример работы кода

Рис. 3 – Пример работы кода

5. Выводы

В результате выполнения практической работы было разработано консольное приложение для поиска статей в Wikipedia с последующим открытием выбранной статьи в браузере. Программа позволяет пользователю ввести поисковый запрос, после чего отображает 10 статей с нумерацией. Пользователь может выбрать номер статьи для открытия в браузере, продолжить просмотр результатов текущего поиска, выполнить новый поиск или выйти из приложения.

В процессе разработки были освоены ключевые аспекты работы с сетевыми запросами в Java: отправка HTTP-запросов к Wikipedia API, обработка JSON-ответов с использованием библиотеки Gson, кодирование URL-параметров и автоматическое открытие веб-страниц в браузере через Desktop API. Проект был реализован с внешними зависимостями в виде JAR-библиотек. Результат размещен в репозитории на GitHub:

https://github.com/varkirsova/oop