**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**КАФЕДРА САПР**

**Практическая работа №1**

**«Консольная программа для поиска в Википедии»**

**по дисциплине**

**«Объектно-ориентированное программирование»**

|  |  |  |
| --- | --- | --- |
| Студентки гр. 3354 |  | Чикарёва М.Д.  Запольская Н.Д. |
| Преподаватель |  | Кулагин М.В. |

**Санкт-Петербург**

**2024**

1. Задание.

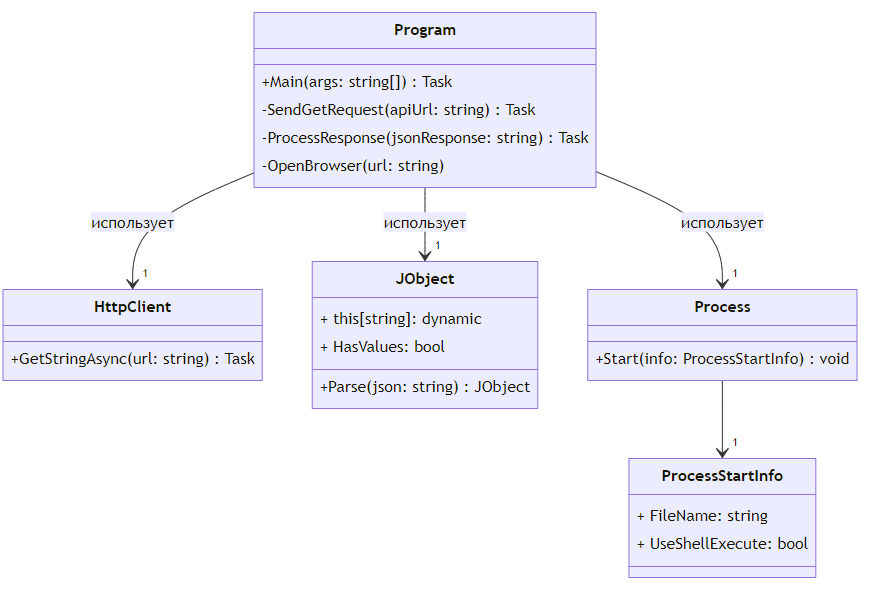
Написать программу, которая с консоли будет считывать поисковый запрос пользователя и выводить результат поиска по Википедии. После выбора нужной статьи программа должна открывать её в браузере, также программа должна корректно реагировать на любой пользовательский ввод.

Задача разбивается на 5 этапов:

1. Считать введённые пользователем данные.
2. Сделать запрос к серверу.
3. Распарсить ответ.
4. Вывести результат поиска.
5. Открыть нужную страницу в браузере.

Использовать готовые библиотеки для работы с Википедией нельзя.

1. Спецификация программы (алгоритм работы).



Основные компоненты программы:

1. Импорт библиотек:

Импортируются модули System, System.IO, System.Net.Http, System.Threading.Tasks, System.Diagnostics, Newtonsoft.Json.Ling, System.Linq.

1. Классы:

- Класс Program: основной класс, содержащий методы для управления выполнением программы. Методы:

- Main(args): запрашивает у пользователя ввод поискового запроса, кодирует запрос для безопасной передачи в URL, создает URL для API, вызывает метод SendGetRequest(apiUrl) для выполнения GET-запроса, вызывает метод ProcessResponse(jsonResponse) для обработки ответа.

- SendGetRequest(apiUrl): создает экземпляр HttpClient, отправляет GET-запрос по указанному URL и возвращает ответ в виде строки.

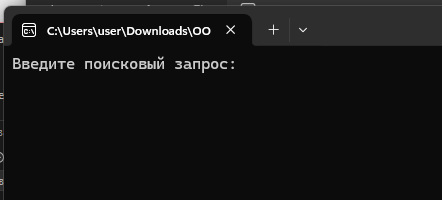
- ProcessResponse(jsonResponse): парсит JSON-ответ, полученный от API, проверяет наличие результатов поиска, выводит результаты поиска на консоль, запрашивает номер статьи, которую необходимо открыть, обрабатывает выбор пользователя и вызывает OpenBrowser(url) для открытия статьи.

- OpenBrowser(url): запускает браузер с указанным URL для открытия статьи.

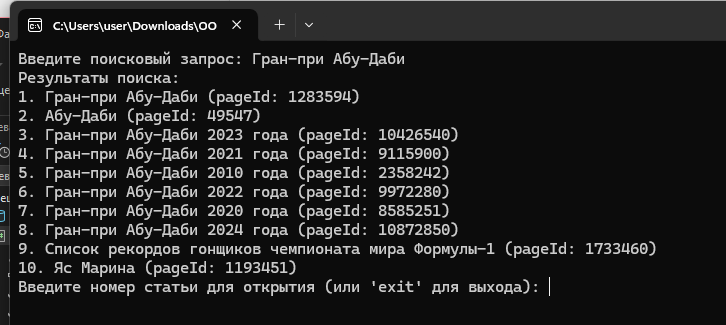
Таким образом, программа работает последовательно, начиная с запроса у пользователя поискового запроса, выполнения запроса к API, обработки и вывода результатов, а также открытия выбранной статьи.

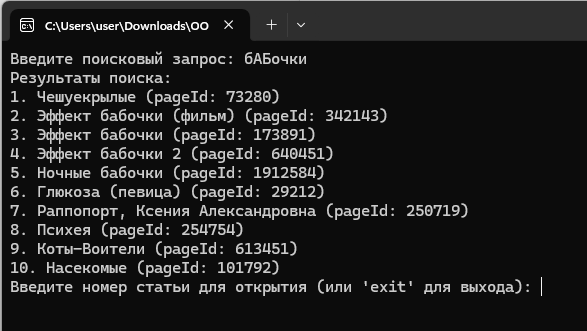
1. Описание интерфейса пользователя программы.

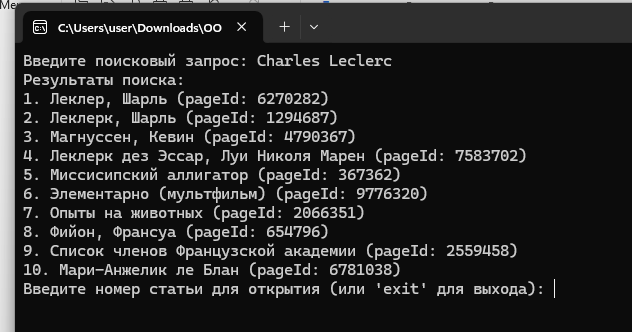
- При запуске программы выводится строка с просьбой пользователю ввести желаемую поисковую строку/слово.



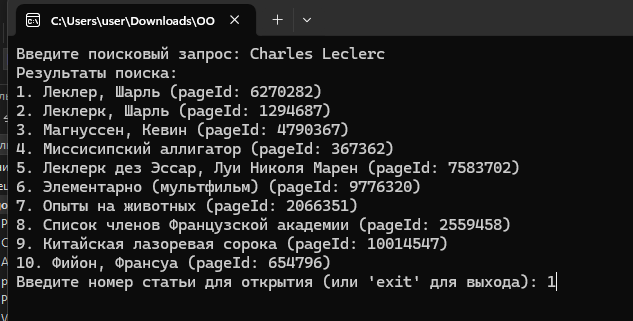
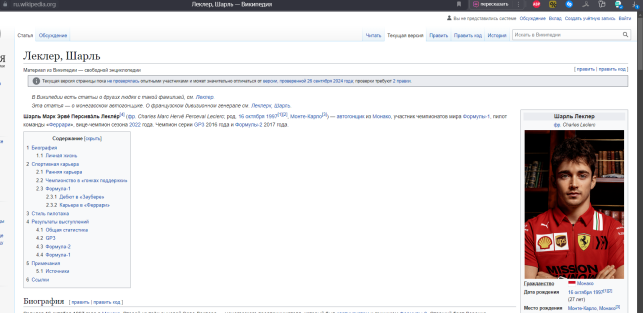
- После ввода выводятся все запросы из Википедии, содержащие запрос пользователя. Работает при любом вводе:



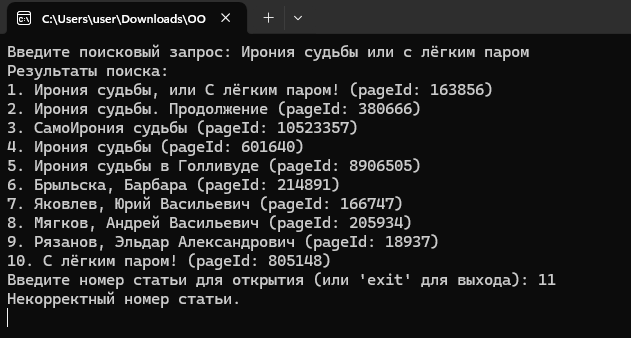




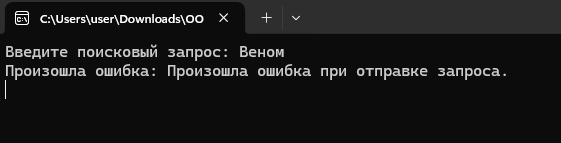
- Дальше пользователю можно либо выйти, либо ввести номер любой из вышедших строк и после этого откроется страница в Википедии, которая была найдена.

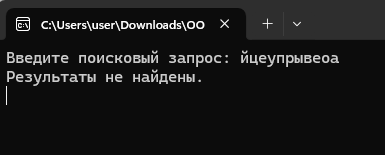
Если выбран неправильный номер, то об этом выйдет соответствующее сообщение:



Если нет интернета, то будет следующее:



- При неправильном вводе выйдет строка, что результаты не найдены.



1. Код программы

using System;

using System.IO;

using System.Net.Http;

using System.Threading.Tasks;

using System.Diagnostics;

using Newtonsoft.Json.Linq;

using System.Linq;

class Program

{

static async Task Main(string[] args)

{

Console.Write("Введите поисковый запрос: ");

string query = Console.ReadLine();

try

{

string encodedQuery = Uri.EscapeDataString(query);

string apiUrl = $"https://ru.wikipedia.org/w/api.php?action=query&list=search&utf8=&format=json&srsearch={encodedQuery}";

// Выполняем запрос к API

string jsonResponse = await SendGetRequest(apiUrl);

await ProcessResponse(jsonResponse);

}

catch (Exception e)

{

Console.WriteLine("Произошла ошибка: " + e.Message);

Console.ReadLine();

}

}

private static async Task<string> SendGetRequest(string apiUrl)

{

using (HttpClient client = new HttpClient())

{

return await client.GetStringAsync(apiUrl);

}

}

private static async Task ProcessResponse(string jsonResponse)

{

var jsonObject = JObject.Parse(jsonResponse);

var searchResults = jsonObject["query"]["search"];

if (searchResults.HasValues == false)

{

Console.WriteLine("Результаты не найдены.");

Console.ReadLine();

return;

}

Console.WriteLine("Результаты поиска:");

for (int i = 0; i < searchResults.Count(); i++)

{

string title = searchResults[i]["title"].ToString();

int pageId = (int)searchResults[i]["pageid"];

Console.WriteLine($"{i + 1}. {title} (pageId: {pageId})");

}

Console.Write("Введите номер статьи для открытия (или 'exit' для выхода): ");

string choice = Console.ReadLine();

if (choice.Equals("exit", StringComparison.OrdinalIgnoreCase))

{

Console.WriteLine("Выход из программы.");

Console.ReadLine();

return;

}

try

{

int choiceIndex = int.Parse(choice) - 1;

if (choiceIndex >= 0 && choiceIndex < searchResults.Count())

{

int pageId = (int)searchResults[choiceIndex]["pageid"];

string articleUrl = $"https://ru.wikipedia.org/w/index.php?curid={pageId}";

OpenBrowser(articleUrl);

}

else

{

Console.WriteLine("Некорректный номер статьи.");

Console.ReadLine();

}

}

catch (FormatException)

{

Console.WriteLine("Некорректный ввод.");

Console.ReadLine();

}

}

private static void OpenBrowser(string url)

{

try

{

Process.Start(new ProcessStartInfo

{

FileName = url,

UseShellExecute = true

});

}

catch (Exception e)

{

Console.WriteLine("Не удалось открыть браузер: " + e.Message);

Console.ReadLine();

}

}

}

1. Выводы.

Программа работает правильно при любых ситуациях. Ввод может быть как большим предложением, так и одним словом, притом если вместо, например, Бабочки или бабочки написать бАБочки, поиск всё равно сработает верно. Неверный ввод тоже обрабатывается и не вылетает, а выводит слова о том, что результаты не найдены, такая же ситуация происходит, если нет доступа к интернету.

Разработанный программный код собирался в Visual Studio, результаты были выложены на GitHub.

**<https://github.com/essslish/oop_chi_and_zap.git>**