Задача:

Написать программу, которая с консоли считывает поисковый запрос, и выводит результат поиска по Википедии.

Задача разбивается на 4 этапа:

1. Считать запрос.
2. Сделать запрос к серверу.
3. Распарсить ответ.
4. Вывести результат.

Первый и четвертый пункт не сильно нуждаются в пояснении, остановимся на запросе к серверу.
Эту задачу тоже можно разбить на несколько этапов:

1. Генерация запроса
2. Запрос к серверу
3. Подготовка к обработке ответа
4. Обработка ответа

Рассмотрим это подробнее:

Генерация запроса

API предоставляет возможность делать поисковые запросы, без ключей и лицензий. Вот таким, примерно, образом:
https://ru.wikipedia.org/w/api.php?action=query&list=search&utf8=&format=json&srsearch="Java"

Вы можете открыть эту ссылку, и посмотреть на результат вывода.

Однако, чтобы запрос прошел удачно, следует убрать из ссылки недопустимые символы, то есть сделать Percent-encoding, он же URL Encoding.
Для этого в java можно воспользоваться статическим методом encode в классе URLEncoder, вот так:

street = URLEncoder.encode(street, "UTF-8");

Вот и всё, URL готов! Осталось теперь сделать запрос к северу…

Запрос к серверу

Можно воспользоваться классом URL. Это самое простое. Просто создать, открыть соединение и получить InputStream. Нам его даже не надо читать, за нас это сделает библиотека GSON.

Можно использовать и *retrofit*.

Подготовка к обработке ответа

Ответ к нам приходит в формате JSON. Это важно.
Но нам его не надо парсить вручную, для этого есть библиотека Gson от Google.
О том, как ей пользоваться, будет рассказано на занятии. Но, тем не менее, примеры есть тут:
https://habrahabr.ru/company/naumen/blog/228279/
http://www.javenue.info/post/gson-json-api

Получить Java-класс для JSON схемы можно несколькими путями: написать классы вручную (бросьте это сразу), использовать плагин к IDEA или online-сервисы вроде этого:
http://www.jsonschema2pojo.org/

Если всё уже работает

Тогда следует улучшить результат. Хорошо бы, чтобы запрос к серверу выполнялся в другом потоке. Это сделать не сложно. А вот для того, чтобы узнать что поток запрос сделал и распарсил результат, можно «повесить» на него Listener, вроде такого:

```
interface DownloadListener{
    void onError();
    void onDownload(List<Place> places);
}
```