****Закрепите полученные знания, выполнив практическую работу. Все задания обязательны для выполнения и сдачи на проверку.****

Выполнить работу гораздо проще после самостоятельной практики. Задания для неё мы разместили под некоторыми видео этого модуля.

#### Цель практической работы

Научиться:

* обходить дерево папок рекурсивно;
* читать файлы форматов CSV и JSON, работать с содержащимися в них данными;
* получать данные из HTML-страниц;
* создавать JSON-файлы.

#### Что входит в работу

Напишите программу, которая собирает данные из файлов разных форматов, лежащих в дереве папок, и с веб-страницы, находящейся в интернете.

#### Задание 1. Программа, собирающая данные из разных источников

##### Что нужно сделать

Выполните задание в отдельном проекте, создайте для проекта директорию FilesAndNetwork/DataCollector.

Напишите программу, которая будет собирать данные из разных источников и записывать два JSON-файла. Парсинг разных данных должен происходить в разных классах. Имена классов и их методов придумайте самостоятельно. По мере реализации проверяйте работу каждого созданного класса. В программе должны быть следующие классы:

* Класс парсинга веб-страницы. В нём должно происходить (реализуйте каждую операцию в отдельных методах):
  + получение HTML-кода страницы [«Список станций Московского метрополитена»](https://skillbox-java.github.io/" \t "/home/parviz/Documents\\x/_blank) с помощью библиотеки jsoup;
  + парсинг полученной страницы и получение из неё следующих данных (создайте для каждого типа данных отдельные классы):
    - линии московского метро (имя и номер линии, цвет не нужен);
    - станции московского метро (имя станции и номер линии).  
      Проверьте работу данного класса: напишите код, который будет его использовать и выводить полученные данные. Для удобства рекомендуем у каждого класса, предназначенного для хранения того или иного объекта, реализовать метод toString, который будет возвращать строку с данными объекта в понятном виде.
* Класс поиска файлов в папках. В методах этого класса необходимо реализовать обход папок, лежащих в [архиве](https://drive.google.com/file/d/1nb3NIfsIp7KLF0OfmZ_nzwYQLM9zfDUg/view?usp=sharing" \t "/home/parviz/Documents\\x/_blank). Разархивируйте его и напишите код, который будет обходить все вложенные папки и искать в них файлы форматов JSON и CSV (с расширениями \*.json и \*.csv). Метод для обхода папок должен принимать путь до папки, в которой надо производить поиск.  
  Проверьте работу данного класса: передайте ему на вход путь к папке и убедитесь, что он вывел информацию о трёх найденных JSON-файлах и о трёх CSV-файлах.
* Класс парсинга файлов формата JSON. Изучите структуру JSON-файлов, лежащих в папках, и создайте класс(ы) для хранения данных из этих файлов. Напишите код, который будет принимать JSON-данные и выдавать список соответствующих им объектов.  
  Проверьте работу данного класса: передайте ему на вход данные любого из JSON-файлов, найденных на предыдущем шаге, и убедитесь, что он выводит данные корректно.
* Класс парсинга файлов формата CSV. Изучите структуру CSV-файлов, лежащих в папках, и создайте класс(ы) для хранения данных из этих файлов. Напишите код, который будет принимать CSV-данные и выдавать список соответствующих им объектов.  
  Проверьте работу данного класса: передайте ему на вход данные любого из CSV-файлов, найденных двумя шагами ранее, и убедитесь, что он выводит данные корректно.
* Класс, в который можно добавлять данные, полученные на предыдущих шагах, и который создаёт и записывает на диск два JSON-файла:
  + со списком станций по линиям и списком линий по формату JSON-файла из проекта SPBMetro (файл [map.json](https://drive.google.com/file/d/1zsSAR_fLGK9Ka84AwUSr5CSOzuB2LIL4/view?usp=sharing" \t "/home/parviz/Documents\\x/_blank));
  + файл stations.json со свойствами станций в следующем формате:

{  
 "stations": [  
 {  
 "name": "Название станции",  
 "line": "Название линии",  
 "date": "Дата открытия в формате 19.01.2005",  
 "depth": "Глубина станции в виде числа",  
 "hasConnection": "Есть ли на станции переход"  
 },  
 …  
 ]  
}

Пример:

{  
  "stations": [  
 {  
  "name": "Ленинский проспект",  
  "line": "Калужско-Рижская",  
  "date": "13.10.1962",  
  "depth": -16,  
  "hasConnection": true  
 },  
 …  
 ]  
}

Если каких-то свойств для станции нет, то в файле не должно быть соответствующих ключей.

* Проверьте созданный класс: запустите получившуюся программу и убедитесь, что она создаёт и записывает два JSON-файла по описанным выше форматам.

##### Советы и рекомендации

* Все варианты подключения библиотеки jsoup в проект — [на странице скачивания библиотеки](https://jsoup.org/download" \t "/home/parviz/Documents\\x/_blank).
* Для подбора и проверки селекторов используйте [онлайн-сервис jsoup](https://try.jsoup.org/" \t "/home/parviz/Documents\\x/_blank).
* Прочитайте статью [«Что такое JSON»](https://habr.com/ru/post/554274/" \t "/home/parviz/Documents\\x/_blank).
* При изучении кода страницы удобно использовать консоль разработчика в браузере. Для этого нажмите F12, перейдите во вкладку Elements и найдите тег <div id="metrodata">. В нём содержатся таблицы с линиями, станциями и пересадками. Обращайте внимание на классы, напишите селекторы на основе найденных классов. Посмотрите [в документации jsoup](https://jsoup.org/cookbook/extracting-data/selector-syntax" \t "/home/parviz/Documents\\x/_blank), как получать элементы по селекторам.
* Обратите внимание на то, что данные в разных источниках могут пересекаться:
  + Одни и те же станции у разных веток при парсинге с сайта. Это могут быть как разные станции (например, в Москве две станции “Арбатская” и две станции “Смоленская”), так и одни и те же, если это станции пересадок.
  + Данные о датах открытия для одних и тех же станций в файлах. Если даты отличаются, то это разные станции с одинаковыми названиями.
  + Разные значения глубины для одних и тех же станций. Здесь приоритетной считайте значения с наибольшей глубиной.

##### Критерии оценки

****Принято****

Программа собирает данные из всех источников, создаёт и записывает на диск два JSON-файла в соответствии с форматом, показанным выше.

****На доработку****

Задание не выполнено.