Контрольное домашнее задание 1, модуль 2

Контрольное домашнее задание предполагает самостоятельную домашнюю работу. Вам потребуется:

1. Изучить предложенные теоретические материалы самостоятельно.
2. Самостоятельно поработать с документацией по языку C#, в т.ч. осуществлять информационный поиск.
3. Разработать программы, определённые основной задачей и индивидуальным вариантом.
4. Сдать в SmartLMS вовремя заархивированный проект с кодом проекта консольного приложения и библиотеки классов, определённые заданием и вариантом.

Формат сдачи работы

Для проверки предоставляется решение, содержащие два проекта: консольное приложение и библиотеку классов. Решение должно быть заархивировано и приложено в качестве ответа на задание в SmartLMS.

Срок выполнения и загрузки работы

Две недели (фактический дедлайн смотреть по SmartLMS)

Опоздания и штрафы

Дедлайн является мягким и еще на протяжении суток работу можно будет отправить на проверку, с учётом штрафов.

Опоздание в часах	**Максимальная оценка, которую можно получить**
1	8
2-3	7
4-5	6
6-7	5
8-9	4
9 и более	1

Основная задача

Разработать решение, содержащие проект *консольного приложения* и *библиотеку классов*, включающую статические классы для работы с данными CSV-файла, определённого индивидуальным вариантом (см. таблицу 1).

Консольное приложение

В консольном приложении:

1. Запросить у пользователя абсолютный путь к файлу с ***csv***-данными. Допустимый формат данных в файле определён индивидуальным вариантом. Для получения данных из файла использовать метод **Read** класса **CsvProcessing** из библиотеки классов.
2. После получения данных из файла предоставить пользователю экранное меню (пример см. ниже), пункты которого отвечают за манипуляции с файлом и его данными:
    a. выборка по значениям полей, указанным в индивидуальном варианте (таблица 1, столбец 3). Для организации выборки использовать методы класса **DataProcessing** из библиотеки классов. Конкретные значения полей для организации выборки получать от пользователя. Результат выводить на экран в табличном, читаемом виде. Если результат пуст, то оповещать пользователя.
    b. сортировка по значениям полей, указанным в индивидуальном варианте (таблица 1, столбец 4). Для организации сортировки использовать методы класса

DataProcessing из библиотеки классов. Результат сортировки выводить на экран в табличном, читаемом виде.

c. сохранение результатов выборок и сортировок в ***csv***-файле при помощи перегруженных методов **Write** класса **CsvProcessing** (правила работы с содержимым файла уточнены в разделе «Библиотека классов»). Требуется использовать в работе обе перегрузки: для записи одной строки – одну, для записи массива строк – вторую. Структура файла, т.е. набор полей, должны быть идентичны исходному, включая заголовки. Разрабатываемое консольное приложение должно без ошибок загружать данные из любого созданного ей файла. Имя файла и необходимость сохранения данных запрашивать у пользователя после вывода данных на экран. Сохранять файл рядом с исполнимым файлом консольного приложения. Если пользователь указал некорректное имя файла, то оповещать его и запрашивать повторный ввод.

3. Реализовать обработку всех исключительных ситуаций, которые могут возникать при работе с вызываемыми методами, включая методы разрабатываемой в рамках КДЗ библиотеки.

Пример экранного меню

```
Укажите номер пункта меню для запуска действия:

1. Произвести выборку по значению Area
2. Произвести выборку по значению Name
3. Произвести выборку по значению Area и Name
4. Отсортировать таблицу по значению Year (прямой порядок)
5. Отсортировать таблицу по значению Name (прямой порядок)
6. Выйти из программы
```

Библиотека классов

У библиотеки классов должно быть осмысленное имя. Библиотека содержит статические классы со статическими методами:

1. Класс **CsvProcessing**: содержит методы для чтения и записи данных в csv файл. Для хранения пути к файлу в классе использовать статическое поле **fPath**, к данным этого поля обращаются все методы:
   - Метод **Read** возвращает массив строк (**string[]**) файла, доступного по **fPath**. Если файл отсутствует, или его структура не соответствуют варианту, то метод выбрасывает исключение с типом **ArgumentNullException**.
   - Метод **Write** отвечает за запись данных в файл и имеет две перегрузки:
     - Метод с параметрами: строкой (**string**) и путём к новому файлу **nPath**. Метод дописывает (не стирая уже записанные в файл данные) в конец файла данные из параметра-строки после последней строки, если файл по **nPath** уже присутствует на диске. Если файл отсутствует, то он должен быть создан по указанному пути. Файлы, путь до которых указан некорректно, метод не создаёт, продумайте и реализуйте самостоятельно вариант программного оповещения для вызывающего кода.
     - Метод с параметром массивом строк (**string[]**) записывает новые данные, переданные в параметре массиве строк в уже существующий по пути **fPath** файл, стирая его исходное содержимое. Если файл по пути **fPath** отсутствует, то он должен быть создан по указанному пути. Файлы, путь до которых указан некорректно, метод не создаёт, продумайте и реализуйте самостоятельно вариант программного оповещения для вызывающего кода.

Класс **DataProcessing** содержит методы для работы с данными, полученными из файла: **методы получения выборок**, указанных в таблице 1, в столбце 3 и **методы сортировок** по данным полей, указанных в столбце 4. Способ упорядочения также указан в столбце 4 таблицы 1.

Комментарии по работе с данными

В данных есть пропуски и пустые поля. Выводить на экран такие данные не нужно, но их требуется учитывать, если пропуски встречаются при сортировках или фильтрах. Например, размещать вначале или конце списка при сортивке.

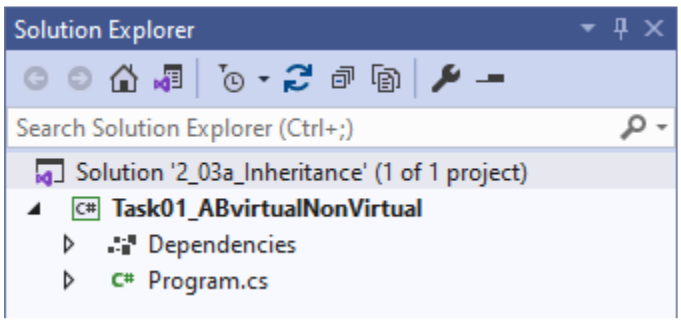
Требования по совместимости и качеству для основной задачи:

- весь программный код должен быть написан на языке программирования C# с учётом использования .net 6.0;
- исходный код должен содержать комментарии, объясняющие неочевидные фрагменты и решения, резюме кода, описание целей кода (см. материалы лекции 1, модуль 1);
- использованные в программе идентификаторы должны соответствовать правилам и соглашениям об именовании идентификаторов C# (https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/coding-style/identifier-names);
- представленный к проверке код должен отвечать общим соглашениям о коде C# Microsoft (https://learn.microsoft.com/ru-ru/dotnet/csharp/fundamentals/coding-style/coding-conventions);
- при перемещении папки проекта библиотеки (копировании / переносе на другое устройство) файлы должны открываться программой также успешно, как и на компьютере создателя, т.е. по относительному пути;
- текстовые данные, включая данные на русском языке, успешно декодируются при представлении пользователю и человекочитаемы;
- программа не допускает пользователя до решения задач, пока с клавиатуры не будут введены корректные данные;
- консольное приложение обрабатывает исключительные ситуации, связанные (1) со вводом и преобразованием / приведением данных как с клавиатуры, так и из файлов; (2) с созданием, инициализацией, обращением к элементам массивов и строк; (3) вызовом методов библиотеки.
- представленная к проверке библиотека классов должна решать все поставленные задачи, успешно компилироваться.
- в качестве структур данных использовать только массивы (тип, производный от **Array**).
- для работы с csv-файлами запрещено использовать сторонние библиотеки и nuget-пакеты, код должен быть подготовлен самостоятельно.
- консольное приложение должно обрабатывать аварийные ситуации и содержать цикл повторения решения.
- каждый метод библиотеки может быть использован отдельно от кода консольного приложения, например, при вставке в другой проект и обращении к методам библиотеки из него по ссылке на библиотеку и указанной в задании сигнатуре метода.

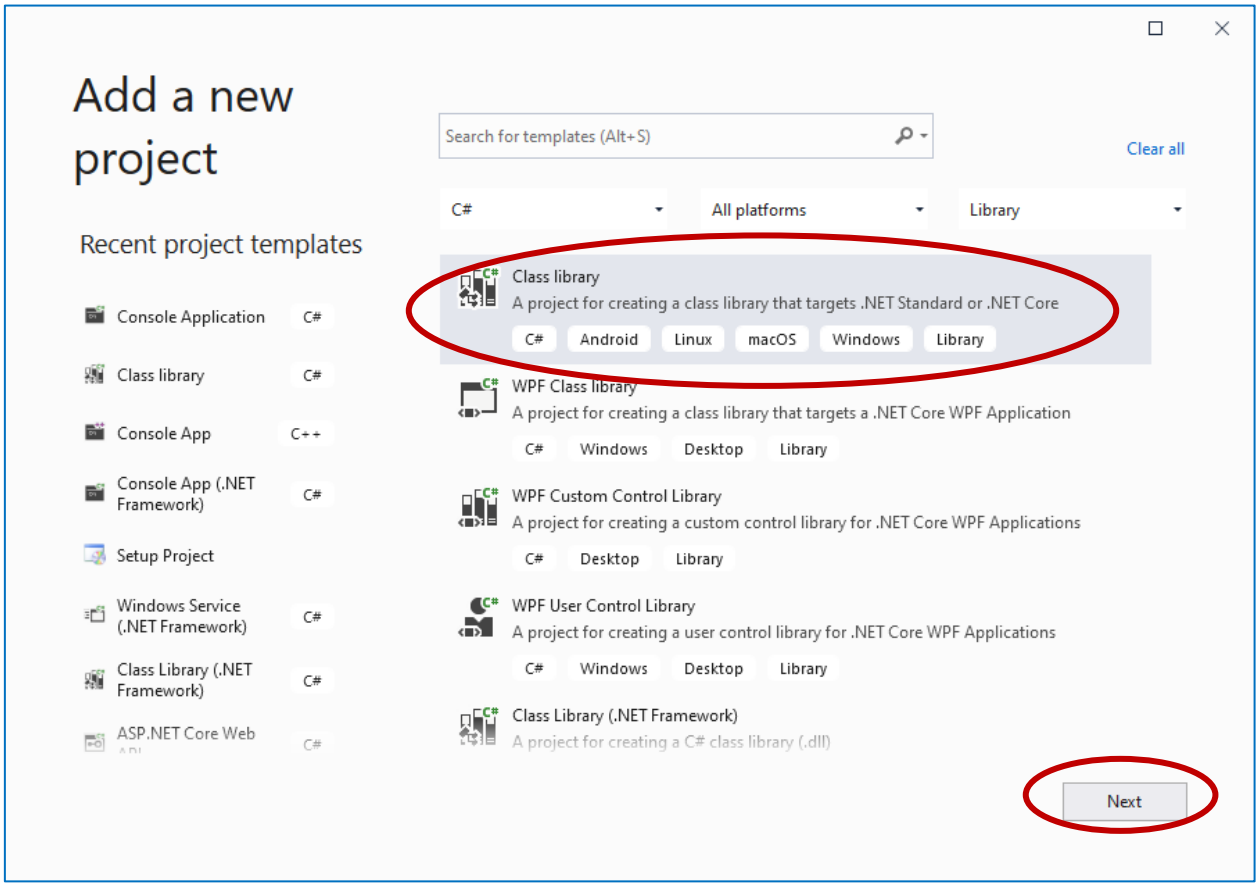
Добавление библиотеки классов в проект

Шаг 1. В решение добавить новый проект Class Library

Для элемента *Решение* в *Обозревателе решений* (Solution Explorer) выбираем контекстное меню для *Решения* и пункт *Добавить новый проект* (Add new project).

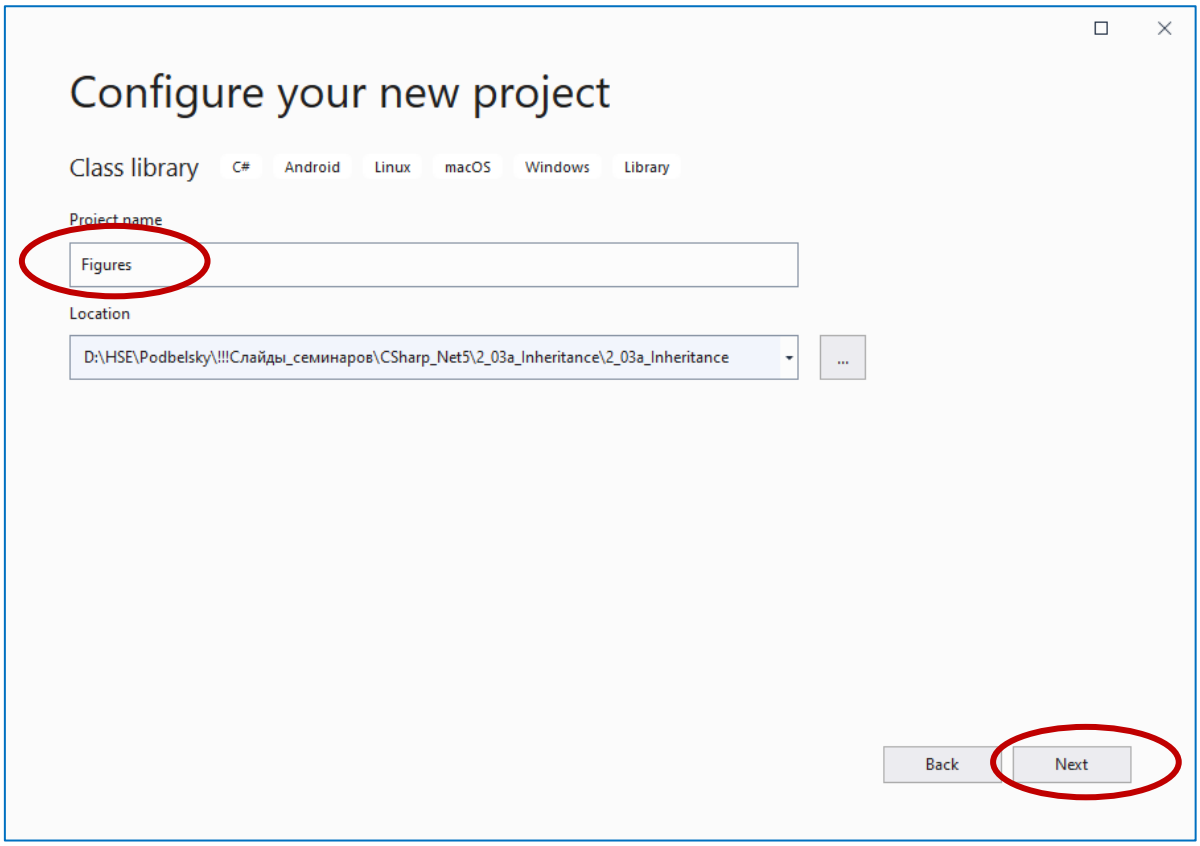


В диалоговом окне добавления нового проекта выбираем шаблон *Библиотека классов*:

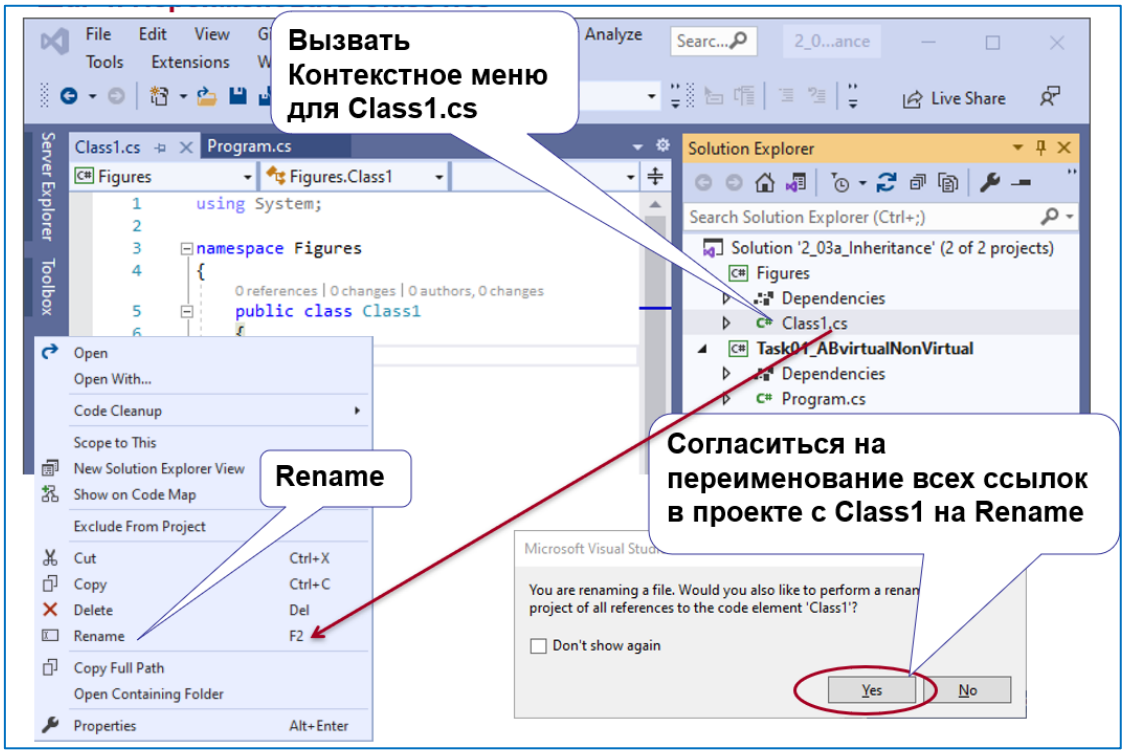


Шаг 2. Задайте имя библиотеки классов

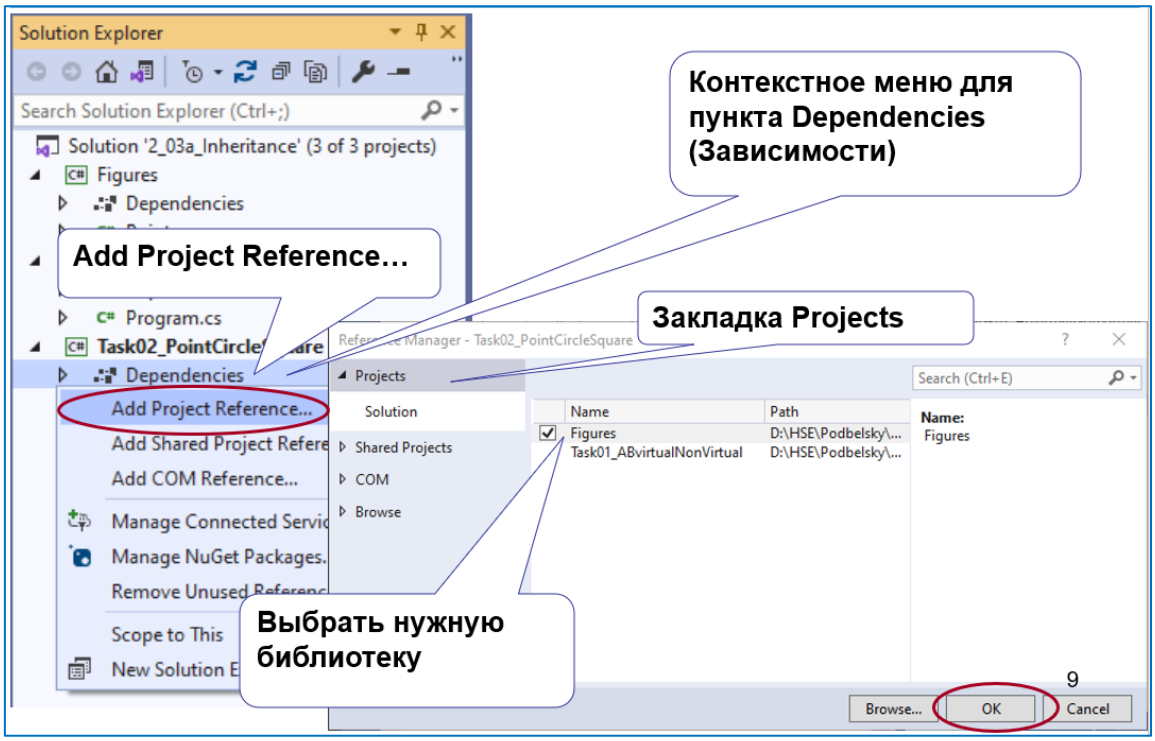
В поле *Название проекта* (Project Name) укажите название библиотеки классов. После нажатия кнопки *Далее* (Next) выбираем платформу .net 6.0.



Шаг 4. Переименовать Class1.cs



Шаг 5. Для подключения библиотеки классов к проекту необходимо в зависимостях проекта добавить ссылку на библиотеку классов.


Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution '2_03a_Inheritance' (3 of 3 projects)
Figures
Dependencies
Program.cs
Dependencies
Add Project Reference...
Add Shared Project Refere
Add COM Reference...
Manage Connected Servic
Manage NuGet Packages.
Remove Unused Referenc
Scope to This
New Solution E
Контекстное меню для пункта Dependencies (Зависимости)
Add Project Reference…
Закладка Projects
Projects
Solution
Shared Projects
COM
Browse
Search (Ctrl+E)
Name
Path
Figures
D:\HSE\Podbelsky\...
Task01_ABvirtualNonVirtual
D:\HSE\Podbelsky\...
Name:
Figures
Выбрать нужную библиотеку
Browse...
OK
Cancel

9

Индивидуальные варианты

Таблица 1. Индивидуальные варианты

Вариант	Набор данных	Название полей для выборок	Название полей для сортировок
1	*2*	*3*	*4*
1	***aeroexpress.csv***	• StationStart • StationEnd • StationStart и StationEnd	• TimeStart в порядке увеличения времени • TimeEnd в порядке увеличения времени
2	***attraction.csv***	• AdmArea • geoarea • District и Geoarea	• Name по алфавиту в прямом порядке • Name по алфавиту в обратном порядке
3	***attraction-TC.csv***	• District • LocationType • AdmArea и Location	• AdmArea по алфавиту в прямом порядке • AdmArea по алфавиту в обратном порядке
4	***botanica-Zaradie.csv***	• LandscapingZone • LocationPlace • LandscapingZone И ProsperityPeriod	• LatinName по алфавиту в прямом порядке • LatinName по алфавиту в обратном порядке
5	***cult-objects.csv***	• SecurityStatus • ObjectType • SecurityStatus и Category	• ObjectNameOnDoc по алфавиту в прямом порядке • ObjectNameOnDoc по алфавиту в обратном порядке
6	***electrocar-power.csv***	• AdmArea • District • AdmArea и пара Longitude_WGS84; Latitude_WGS84	• AdmArea по алфавиту в прямом порядке • AdmArea по алфавиту в обратном порядке
7	***gas-station.csv***	• District • Owner • AdmArea и Owner	• TestDate по возрастанию даты • TestDate по убыванию даты
8	***geraldic-signs.csv***	• Type • RegistrationDate	• RegistrationNumber в порядке возрастания номера

		• CertificateHolderName и RegistrationDate	• RegistrationNumber в порядке убывания номеров
9	***Hockey.csv***	• ObjectName • NameWinter • District и HasDressingRoom	• Lighting по алфавиту в прямом порядке • Seats в порядке возрастания
10	***hotels.csv***	• FullAvailable • PresenceParking • District и Available_k	• FullName по алфавиту в прямом порядке • FullName по алфавиту в обратном порядке
11	***ice-hills.csv***	• NameWinter • HasEquipmentRental • AdmArea и HasWifi	• ServicesWinter по алфавиту в прямом прядке • UsagePeriodWinter в порядке возрастания даты открытия периода
12	***monuments.csv***	• SculpName • LocationPlace • ManufactYear и Material	• SculpName по алфавиту в прямом порядке • ManufactYear по убыванию
13	***radiant-railway-st.csv***	• Station • RailwayLine • RailwayLine и WorkingHours	• ID по возрастанию • WorkingHours по возрастанию времени открытия
14	***recreators.csv***	• MainObjects • Workplace • RankYear и Photo	• По алфавиту по фамилиям, содержащимся в поле Name • RankYear по убыванию
15	***station-rate.csv***	• NameOfStation • Line • NameOfStation и Month	• Year по возрастанию • NameOfStation по алфавиту
16	***taxi-parking.csv***	• AdmArea • CarCapacity • District и Mode	• CarCapacity по увеличению ёмкости • CarCapacity по уменьшению ёмкости
17	***transport-changing-points.csv***	• District • CarCapacity • Status и NearStation	• AvailableTransfer по алфавиту • YearOfComissioning по убыванию
18	***wifi-cult-centres.csv***	• CoverageArea • WiFiName	• CulturalCenterName по алфавиту • NumberOfAccessPoints по возрастанию

		• District и AccessFlag	
19	***wifi-library.csv***	• AdmArea • WiFiName • FunctionFlag и AccessFlag	• LibraryName по алфавиту • CoverageArea по убыванию
20	***wifi-parks.csv***	• CoverageArea • ParkName • AdmArea и CoverageArea	• Name по алфавиту • CoverageArea по возрастанию