

Контрольное домашнее задание 3, модуль 3

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. Вовремя сдать в SmartLMS заархивированное решение, включающее в себя код проекта консольного приложения и библиотеки классов, определённые заданием и вариантом.

Формат сдачи работы

Для проверки предоставляется решение, содержащие два проекта (консольное приложение и библиотека классов) и исходный файл с данными. Решение и файл с данными должны быть заархивировано в один (общий) архив и приложены в качестве ответа на задание в SmartLMS. **Если работа претендует на оценку 10, дополнительно нужно приложить в комментарии ссылку на опубликованного бота (см. подробности в задании).**

Срок выполнения и загрузки работы

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

Задание

Основное задание

В данном домашнем задании мы хотим вернуться к старому опыту с новыми знаниями. Во втором модуле в КДЗ 2-1 мы разрабатывали библиотеку для работы с CSV файлом. Теперь, когда мы знаем как работать с различными форматами сериализации и LINQ предлагаем попробовать применить все новые знания к старому заданию.

В рамках задания вам предлагается разработать Telegram-бота для обработки CSV файлов. В индивидуальном варианте вы найдете название вашего csv-файла с данными и функционал, который необходимо реализовать.

Требования к библиотеке классов

Библиотека классов должна содержать:

- 1) Класс **MyType**¹ представляет объекты, описанные в CSV файле индивидуального варианта. В таблице с индивидуальными вариантами приведены названия некоторых полей, остальные поля должны называться согласно правилам нейминга Microsoft. Классы должны содержать конструктор(ы) для инициализации своих полей.
- 2) Нестатический класс CSVProcessing. Содержит методы для чтения csv файла и записи в него.
 - a. Метод *Write*: принимает на вход коллекцию объектов типа MyType и возвращает объект типа Stream, который будет использован для отправки csv документа Telegram-ботом.
 - b. Метод *Read*: принимает на вход Stream с csv файлом из Telegram-бота и возвращает коллекцию объектов типа MyType.
- 3) Нестатический класс JSONProcessing. Содержит методы для чтения json файла и записи в него.
 - a. Метод *Write*: принимает на вход коллекцию объектов типа MyType и возвращает объект типа Stream, который будет использован для отправки json файла Telegram-ботом.
 - b. Метод *Read*: принимает на вход Stream с json файлом из Telegram-бота и возвращает коллекцию объектов типа MyType.
- 4) Реализации классов не должны нарушать принцип открытости / закрытости (Open Close Principle) и принцип единственной ответственности (Single Responsibility Principle).
- 5) Иерархии типов не должны нарушать принципа подстановки Лисков (Liskov Substitution Principle) и проектируются, исходя из соблюдения принципа инверсии зависимостей (Dependency Inversion Principle).
 - a. Архитектурные принципы (<https://learn.microsoft.com/ru-ru/dotnet/architecture/modern-web-apps-azure/architectural-principles>)
- 6) Реализации классов не должны нарушать инкапсуляцию и отношения, заданные между типами, например, предоставлять внешние ссылки на поля или изменять состояние объекта без проверок.
- 7) Классы библиотеки должны быть доступны за пределами сборки.
- 8) Каждый нестатический класс (при наличии) обязательно должен содержать, в числе прочих, конструктор без параметров или эквивалентные описания, допускающие его прямой или неявный вызов.
- 9) Запрещено изменять набор данных для классов, которые строятся на основе CSV-представлений из индивидуальных вариантов (например, добавлять поля, не содержащиеся в CSV-представлении).
- 10) Допускается расширение открытого поведения или добавление закрытых функциональных членов класса.
- 11) Допускается использование собственных (самописных) иерархий классов в дополнение к предложенным в индивидуальном варианте, также с соблюдением ООП принципов.

¹ имя **MyType** – является заглушкой и должно быть заменено вами на более подходящее имя класса, на основе понимания данных из файла индивидуального варианта

Требования к интерфейсу Telegram-бота

Для разработки Telegram-бота вам предлагается воспользоваться NuGet-пакетом TelegramBots. Вам необходимо самостоятельно изучить инструменты данного пакета используя его документацию. Следует обратить внимание на следующие разделы:

- Установка - <https://telegrambots.github.io/book/1/quickstart.html>
- Создание первого бота - <https://telegrambots.github.io/book/1/example-bot.html>
- Отправка текста - <https://telegrambots.github.io/book/2/send-msg/text-msg.html>
- Скачивание файла - <https://telegrambots.github.io/book/3/files/download.html>
- Загрузка файла - <https://telegrambots.github.io/book/3/files/upload.html>
- Поток записи в память (может быть полезным при работе с загрузкой и скачиванием файлов) - <https://learn.microsoft.com/ru-ru/dotnet/api/system.io.memorystream?view=net-6.0>

Бот должен предоставлять следующие функциональность:

1. Загрузить CSV файл на обработку, бот получает файл который будет обрабатываться дальнейшими командами.
2. Произвести выборку по одному из полей файла. Поля, по которым можно произвести выборку, указаны в таблице индивидуального варианта. Для реализации этого пункта необходимо использовать объекты MyType и LINQ запросы.
3. Отсортировать по одному из полей. Поля, по которым можно сортировать выборку, указаны в таблице индивидуального варианта. Для реализации этого пункта необходимо использовать объекты MyType и LINQ запросы.
4. Скачать обработанный файл в формате CSV или JSON.
5. Загрузить JSON файл на обработку (ваша программа должна уметь принимать обратно те JSON файлы, которые она создала, CSV-выгрузок это тоже касается).

Общие требования к КДЗ

1. Соблюдение определённых программой учебной дисциплины и требований к программной реализации работ – **обязательно**.
2. Соблюдение [соглашений](#) о качестве кода (осмысленное комментирование входит в этот пункт) – **обязательно**.
3. Весь программный код должен быть написан на языке программирования C# с учётом использования .net 6.0.
4. Исходный код должен содержать комментарии, объясняющие неочевидные фрагменты и решения, резюме кода, описание целей кода (см. материалы лекции 1, модуль 1).
5. 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.
6. Текстовые данные, включая данные на русском языке, успешно декодируются при представлении пользователю и человекочитаемы.
7. Ресурсы, выделяемые при работе с файлами, должны освобождаться программой.
8. Все созданные программой CSV-файлы при сохранении имеют такую же структуру, как и файл с примером, и должны без проблем обрабатываться программой в качестве входных данных.
9. Программа **не** допускает пользователя до решения задач, пока не будут введены корректные данные.
10. Приложение обрабатывает исключительные ситуации, связанные (1) со вводом и преобразованием / приведением данных (как с клавиатуры, так и из файла); (2) с созданием, инициализацией, обращением к элементам массивов и строк; (3) с вызовом методов библиотеки.

11. Представленная к проверке библиотека классов должна решать все поставленные задачи и успешно компилироваться.
12. Поскольку в описаниях классов присутствует «простор» для принятия решений, то каждое такое решение должно быть описано в комментариях к коду программы. Например, если выбран тип исключения, то должно быть письменно обосновано, почему вы считаете его наиболее подходящим в рамках данной задачи (если не выбран обоснование тоже должно присутствовать).

Требования на оценку 9 и 10

Решение, претендующее на оценку 9: требуется полное выполнение основной задачи и требований к качеству кода, полное соответствие изученным в дисциплине принципам ООП и SOLID, управление ходом выполнения программы на основе обработки исключений и логирования истории запросов при помощи Microsoft Extensions Logging (<https://learn.microsoft.com/en-us/dotnet/core/extensions/logging?tabs=command-line>). Логи должны быть сохранены в файл в директорию (папку) **var** на уровне проекта Telegram-бота.

Решение, претендующее на оценку 10: требуется выполнение требований на оценку 9 и, дополнительно, требуется опубликовать вашего бота на любом доступном хостинге, например Yandex Cloud. Ссылку на бота предоставить в комментарии к сдаваемой работе.

Индивидуальные варианты

Вариант	Набор данных	Название полей для выборки	Название полей для сортировок
1	2	3	4
1	<i>aeroexpress.csv</i>	<ul style="list-style-type: none"> StationStart StationEnd StationStart и StationEnd 	<ul style="list-style-type: none"> TimeStart в порядке увеличения времени TimeEnd в порядке увеличения времени
2	<i>attraction.csv</i>	<ul style="list-style-type: none"> AdmArea Geoarea District и Geoarea 	<ul style="list-style-type: none"> Name по алфавиту в прямом порядке Name по алфавиту в обратном порядке
3	<i>attraction-TC.csv</i>	<ul style="list-style-type: none"> District LocationType AdmArea и Location 	<ul style="list-style-type: none"> AdmArea по алфавиту в прямом порядке AdmArea по алфавиту в обратном порядке
4	<i>botanica-Zaradie.csv</i>	<ul style="list-style-type: none"> LandscapingZone LocationPlace LandscapingZone и ProsperityPeriod 	<ul style="list-style-type: none"> LatinName по алфавиту в прямом порядке LatinName по алфавиту в обратном порядке
5	<i>cult-objects.csv</i>	<ul style="list-style-type: none"> SecurityStatus ObjectType SecurityStatus и Category 	<ul style="list-style-type: none"> ObjectNameOnDoc по алфавиту в прямом порядке ObjectNameOnDoc по алфавиту в обратном порядке
6	<i>electrocar-power.csv</i>	<ul style="list-style-type: none"> AdmArea District AdmArea и пара Longitude_WGS84; Latitude_WGS84 	<ul style="list-style-type: none"> AdmArea по алфавиту в прямом порядке AdmArea по алфавиту в обратном порядке