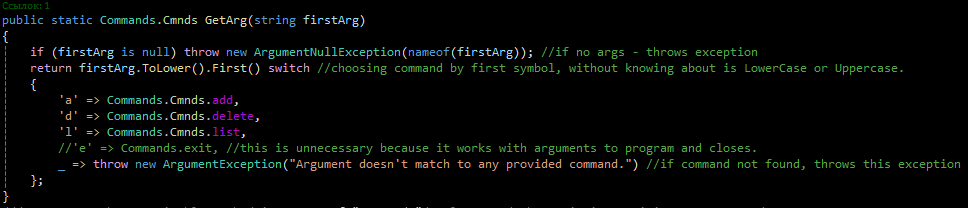
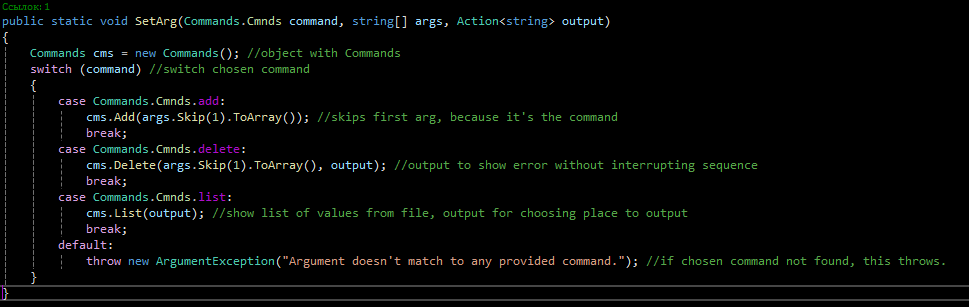
**Отчёт**

Создан метод определяющий полученные аргументы, в котором по первой букве первого аргумента определяется действие и возвращает его.



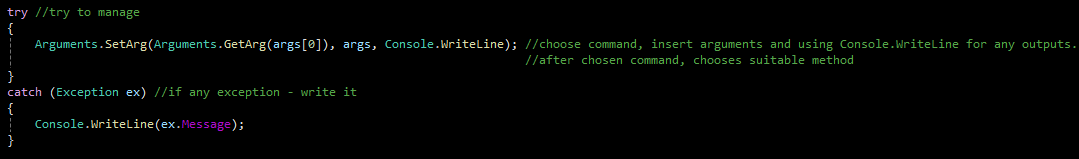
Создан метод, который по выбранному действию выбирает нужный метод управления задачами.



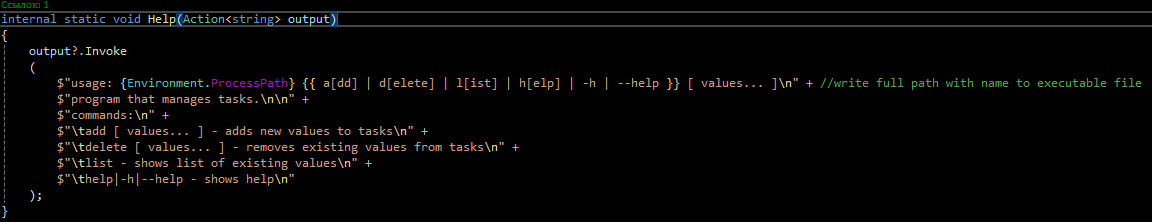
Список доступных команд, определённых в отдельном классе, где определены методы управления задачами.



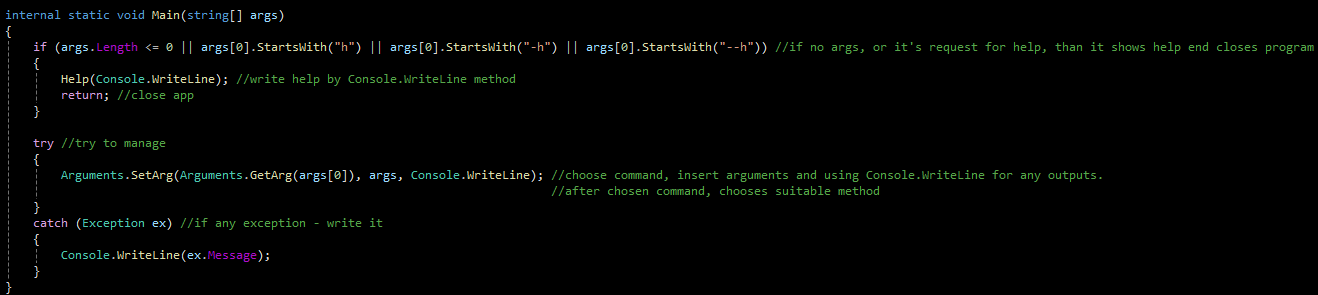
Оба метода добавлены в главный метод – точку входа.



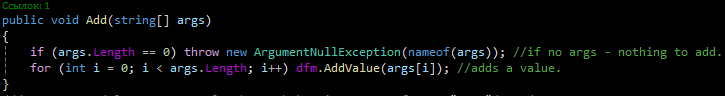
Также для точки входа создан метод выводящий информацию о доступных аргументах пользователю.



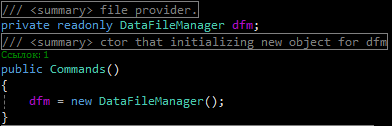
Этот метод включён в точку входа при определённых условиях.



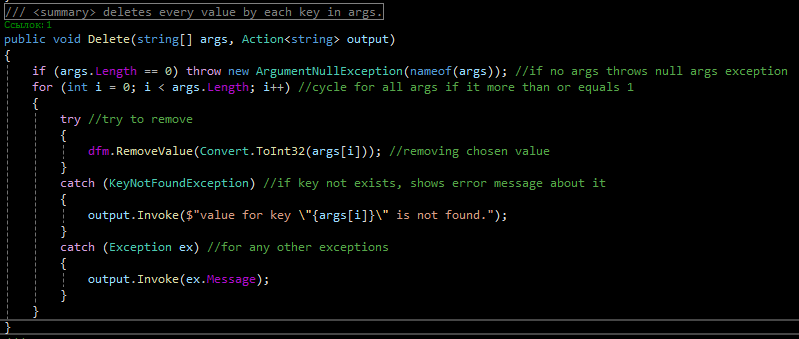
Разработан метод добавления задачи из полученных аргументов, использующий объект разработанного класса, с помощью которого происходит запись в файл.



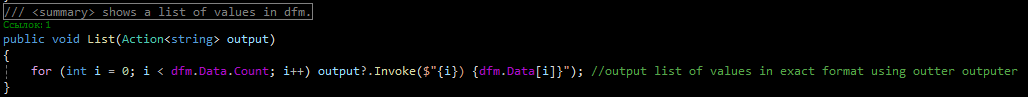
Объявление объекта и присваивание значения с помощью конструктора для класса управления файлом



Метод удаления задачи. в этом методе также обработаны ситуации, когда появляются ошибки. И с помощью делегата, указанного в параметрах метода, выводится текст ошибки, но работа метода не прекращается.

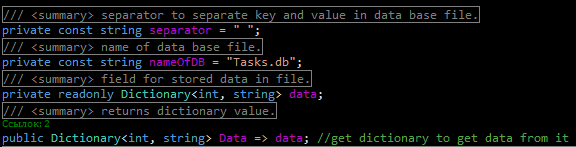


Метод для вывода списка задач.



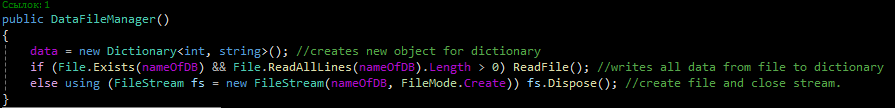
Реализация класса для работы с файлом.

Поля, в которых хранятся данные класса. В поле «separator» хранится знак-разделитель который обозначает разделение данных. В пол «nameOfDB» хранится имя файла, в котором будет находится список задач. В поле «data» хранятся задачи, полученные из файла, или добавленные пользователем.

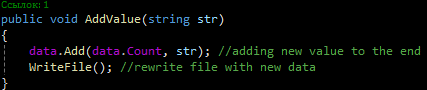


Так как для того чтобы получить значение из закрытого поля, нужно добавить свойство, с помощью которого будет возвращаться объект.

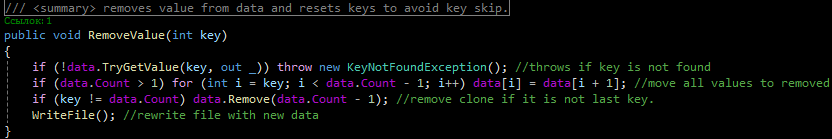
Конструктор класса создаёт новый объект для поля «data» и, если существуют данные в файле, сразу устанавливает их в созданный объект. Иначе создаёт файл в котором будут хранится задачи.



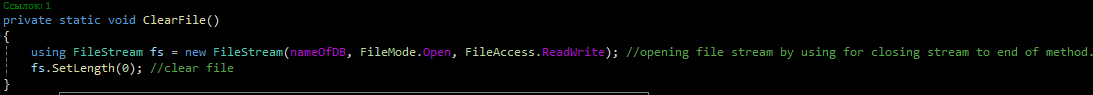
Метод добавления значения в поле, и изменения файла с новыми значениями.



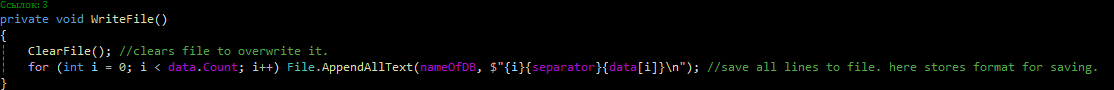
Метод удаления значения. Если не существует указанного значения в списке задач – возвращается сообщение об ошибке. Если количество элементов больше одного, то все элементы переставляются на один к первому элементу, а последний элемент, который является дублем, удаляется. После проделанной работы – перезаписывается файл со списком задач.



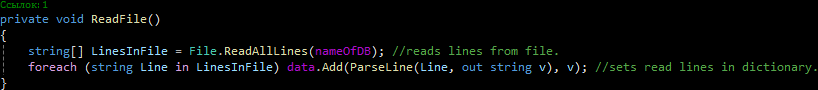
Вспомогательный метод очистки файла от предыдущих значений.



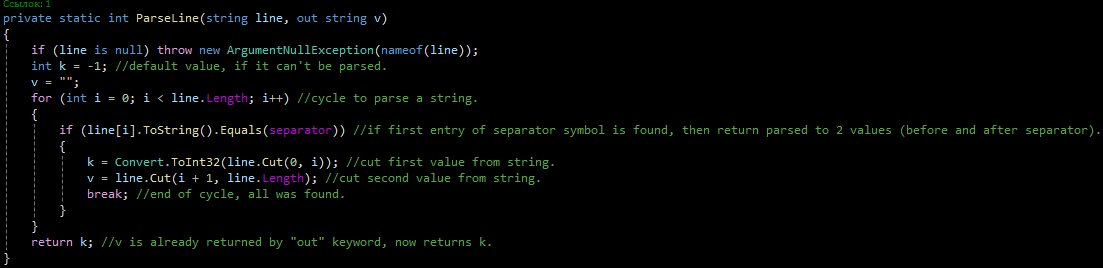
Метод записи всех строк задач, записанных в поле «data», предварительно очистив файл.



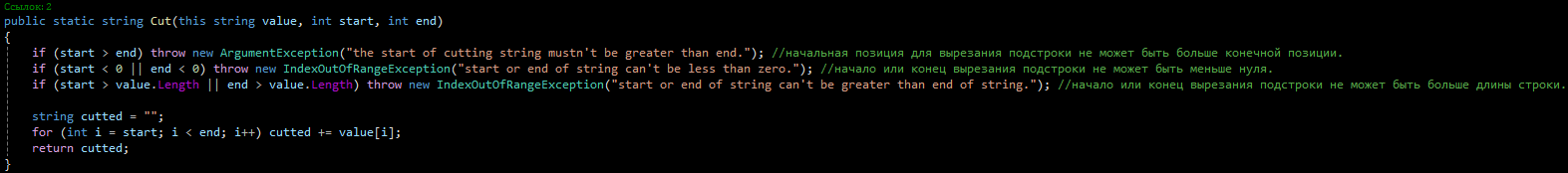
Метод чтения файла, и записи полученных строк в поле «data».



Вспомогательный метод для чтения значений из файла.



Специальноый метод «расширение» которое помогает правильно вырезать подстроку из строки.

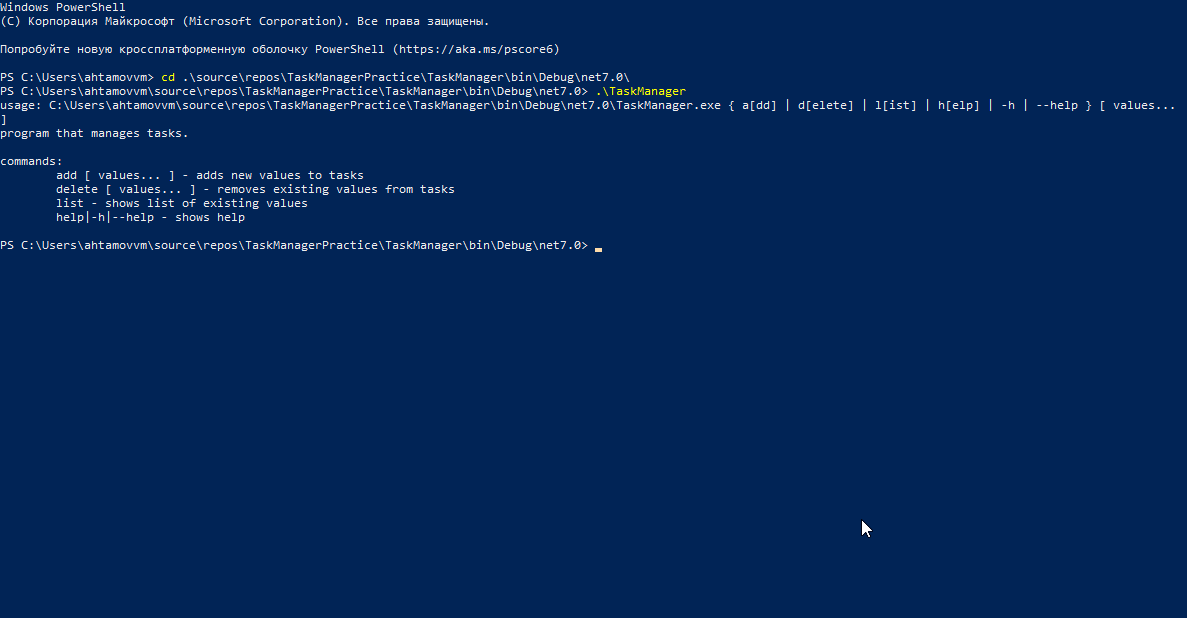


Исходный код на Github: <https://github.com/nefir2/TaskManagerPractice>

Тесты программы:

Открыв консоль (cmd, powershell, bash) и находясь в той же папке что и программа (либо установив путь до программы в специальную переменную среды «PATH») можно взаимодействовать со списком задач.

Введя ./TaskManager выведется информация о программе.



Строки:

./TaskManager.exe

./TaskManager h

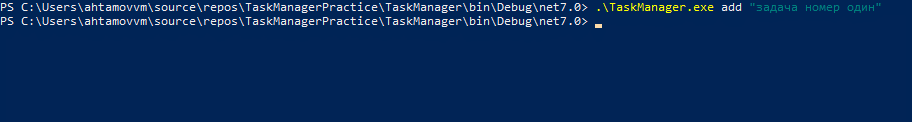
./TaskManager -h

./TaskManager --help

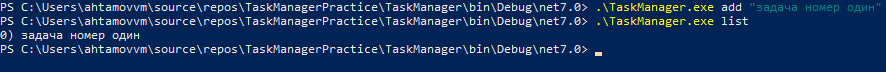
Будут иметь тот же эффект.

Также для консоли cmd и powershell не имеет значения в какую сторону идёт наклон символа «/».

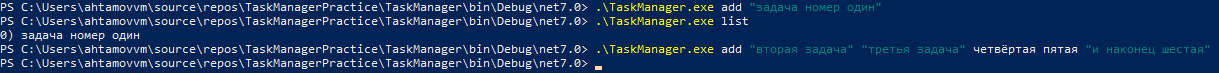
Для того чтобы добавить задачу, нужно ввести имя программы, и аргумент «add». а после и сами значения в той же строке, для добавления в список.



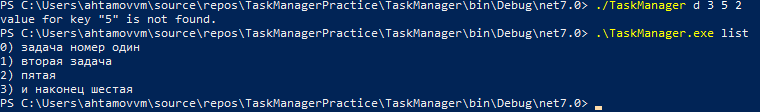
Чтобы просмотреть список задач, достаточно ввести имя программы и аргумент «list».



Также можно указать несколько задач через пробел



Для удаления строк надо указать аргумент «delete» и номера задач вначале строки.



Также поддерживаются сокращения команд «add», «delete», «list» до первой буквы. Для программы, команды «add» и «a» равнозначны.

