МИНОБРНАУКИ РФ

ФГБОУ ВО Тверской государственный технический университет

Кафедра “Программное обеспечение”

Курсовая работа

Дисциплина “Основы программирования”

Тема: «Модульное программирование в процедурах и функциях»

Выполнили: студенты группы

“Б.ПИН.РИС-22.06”

Кириллов Владислав Константинович

Богданов Никита Романович

Виноградов Глеб Евгеньевич

Миронова Анна Алексеевна

Невзоров Павел Евгеньевич

Калинин Павел Дмитриевич

Проверил: Биллиг Владимир Арнольдович

Тверь, 2023

**Оглавление**

[**Введение** 3](#_Toc133576975)

[**Проект 1: Задача Флойда** 4](#_Toc133576976)

[**Проект 2: Перевод из одной системы счисления в другую** 9](#_Toc133576977)

[**Проект 3: Строки** 12](#_Toc133576978)

[**Заключение** 19](#_Toc133576979)

# **Введение**

Основы программирования – это дисциплина, которая знакомит студентов с основными концепциями и инструментами программирования. Она является фундаментом для дальнейшего изучения программирования и создания сложных программных продуктов.

Одним из главных критериев построения проекта является его корректность. Корректность означает, что программа работает так, как ожидается, и выполняет задачи, для которых она предназначена. В процессе обучения на первом курсе студенты учатся создавать программы, которые работают корректно, используя правильный синтаксис и логику программирования.

Тестирование является важным способом обеспечения корректности проекта. Оно позволяет проверить, что программа работает правильно во всех возможных ситуациях и не содержит ошибок. В процессе обучения студенты учатся создавать тесты для своих программ, чтобы убедиться, что они работают правильно.

Особенности построения интерфейса, ориентированного на пользователя, также являются важными в программировании. Интерфейс должен быть интуитивно понятным и удобным для пользователя. В процессе обучения студенты изучают, как создавать простые и понятные пользовательские интерфейсы для своих программ.

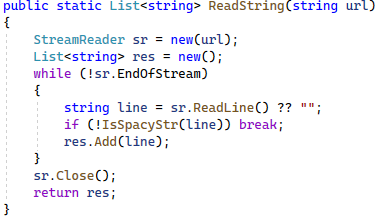
В процессе обучения на первом курсе студенты учатся не только основам программирования, но и другим важным навыкам, таким как работа с командной строкой, использование систем контроля версий, и написание документации для своих программ. Эти навыки помогают студентам создавать более сложные и профессиональные программы в будущем.

Миронова Анна

# **Проект 1: Задача Флойда**

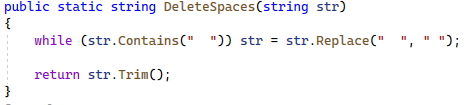
Проект “Задача Флойда” получает на вход текстовый файл и производит его форматирование. Проект был разобран и декомпозирован на следующие методы:

**Метод ReadString**

****

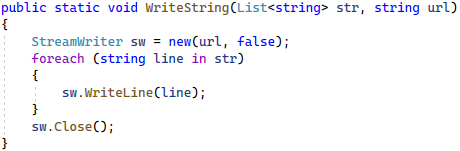
Данный метод считывает текст из файла и заполняет список строк, с которым в дальнейшем будут работать методы по удалению лишних пробелов.

**Метод DeleteSpaces**



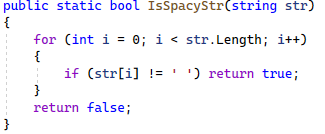
Данный метод служит для удаления подряд идущих пробелов, вполне простой и отлично работающий.

**Метод WriteString**

****

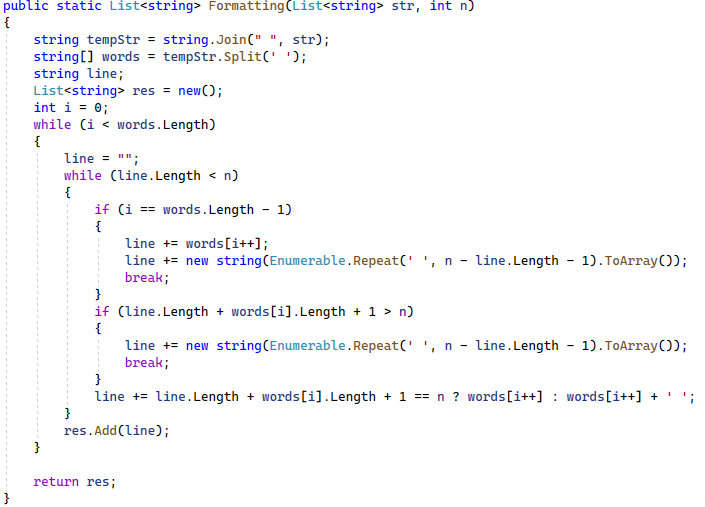
Метод для вывода текста в файл, который поочередно добавляет отформатированные строки.

**Метод IsSpacyStr**

****

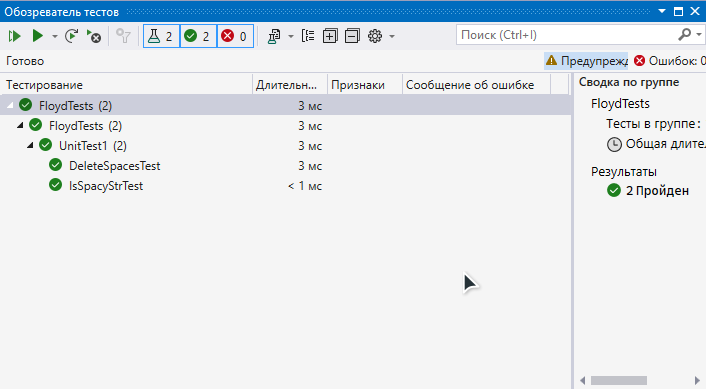
Данный метод проверяет наличие пустых строк в файле, которые также необходимо удалить.

**Метод Formatting**

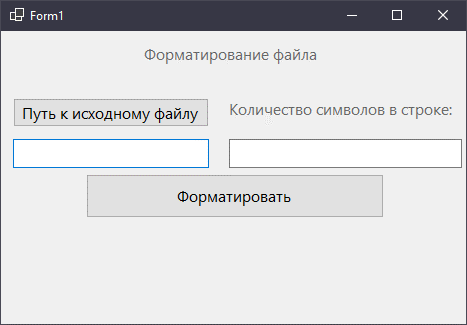
****

Основной метод для форматирования текста.

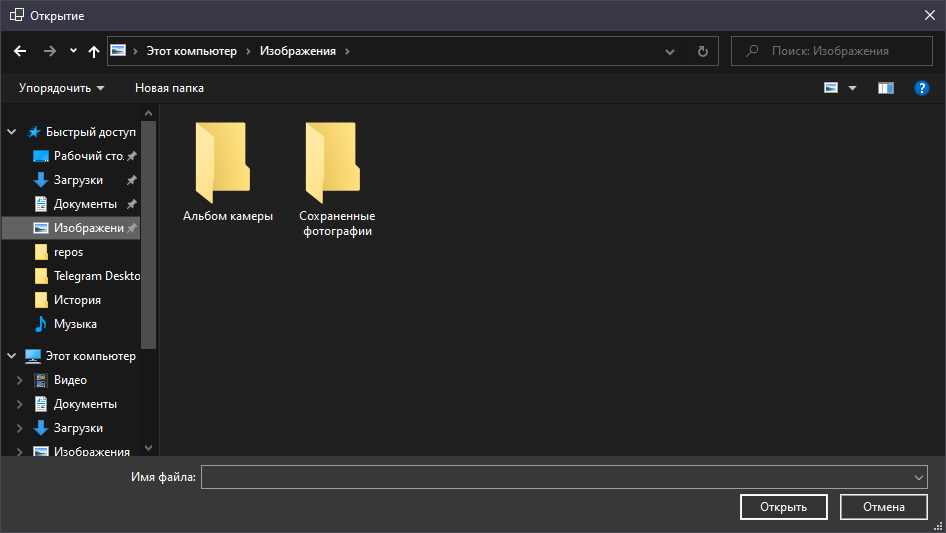
Для проверки корректности основных методов IsSpacyStr и DeleteSpaces были разработаны тесты, которые наши методы проходят без проблем:



Единственные данные, получаемые от пользователя это путь к исходному файлу и количество символов в строке, поэтому пользовательский интерфейс создан максимально простым, без излишеств.



Также, для облегчения пользования интерфейсом, была добавлена кнопка «Путь к исходному файлу», не требующая ручного ввода полного пути, а при помощи выбора файла стандартным способом Windows.



Над проектом “Задача Флойда” работали все участники команды. Каждый выполнял следующие обязанности:

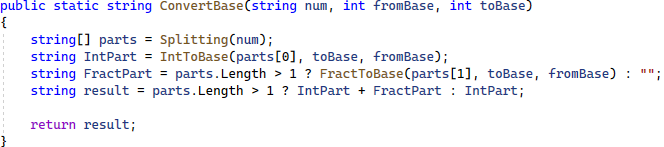
|  |  |
| --- | --- |
| Разработка библиотеки | Кириллов Владислав  Павел Невзоров  Виноградов Глеб |
| Разработка интерфейса | Калинин Павел  Богданов Никита |
| Разработка тестов | Миронова Анна |

Виноградов Глеб

# **Проект 2: Перевод из одной системы счисления в другую**

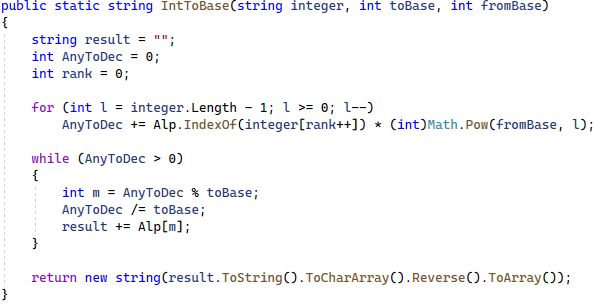
Проект “Перевод из одной системы счисления в другую” выполняет перевод чисел из одной системы счисления в другую. Перевод возможен во все системы счисления от 2 до 36. Проект был разобран и декомпозирован на следующие методы:

**Метод ConvertBase**



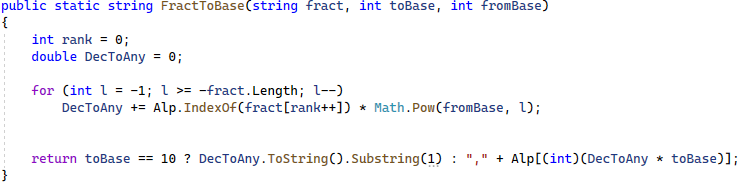
Главный метод для конвертации числа из одной системы счисления в другую. Данный метод учитывает, что число может быть, как целым, так и вещественным. Также он создает алфавит в виде текстовой строки, который нам понадобится далее.

**Метод IntToBase**



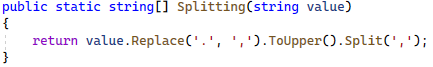
Данный метод переводит целую часть нашего числа, которое мы хотим конвертировать, из одной системы счисления в другую.

**Метод FractToBase**



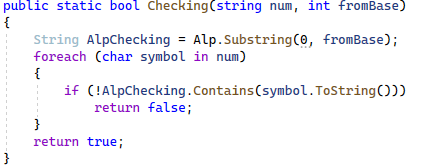
Данный метод необходим для перевода дробной части нашего искомого числа из одной системы счисления в другую.

**Метод Splitting**



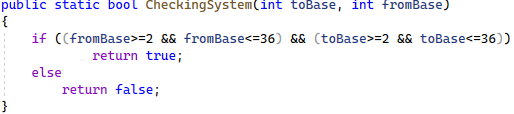
Данный метод необходим для разделения числа на целую и дробную часть.

**Метод Checking**



Данный метод необходим для проверки, правильно ли пользователь задал число для конвертации учитывая изначальную систему счисления. (Пользователь может ошибиться и написать, например, букву в числе с десятичной системы счисления и т.д.)

**Метод CheckingSystem**



Данный метод необходим для проверки корректности ввода систем счисления.

Над проектом “перевод из одной системы счисления в другую” работали все участники команды. Каждый выполнял следующие обязанности:

|  |  |
| --- | --- |
| Разработка библиотеки | Кириллов Владислав  Виноградов Глеб  Невзоров Павел |
| Разработка интерфейса | Богданов Никита  Калинин Павел |
| Разработка тестов | Миронова Анна |

Богданов Никита

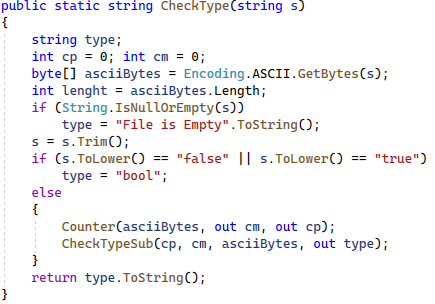
# **Проект 3: Строки**

Проект “Строки” представляет собой программу, способную выполнять следующие действия со строками:

* Найти вхождение подстроки в строку
* Вставка и удаление подстроки
* Замена подстроки другой подстрокой

Также программа должна определять тип содержимого строки и конвертировать строку к нужному типу данных. Проект был разобран и декомпозирован на следующие методы:

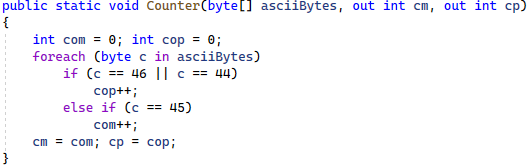
**Метод CheckType**



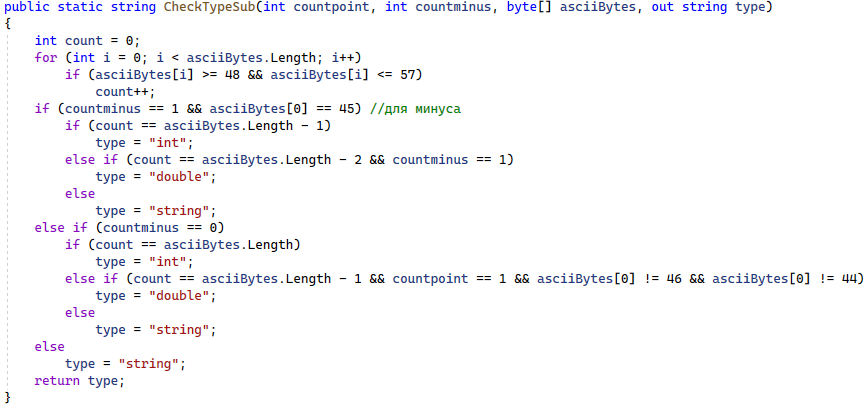
Данный метод нужен для определения типа данных записанных в строке. “CheckType” разбивает строку на массив ASCII символов. Дальше вся работа пойдет с помощью этого массива.

Также внутри него вызываются 2 вспомогательные функции

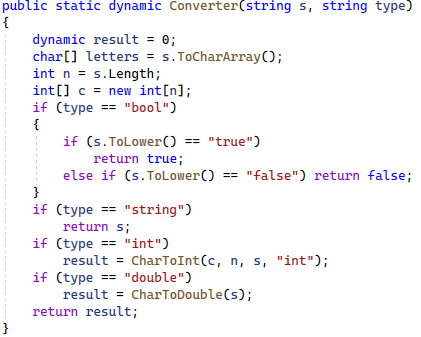
1. Counter – выполняет подсчет точек и запятых в строке



1. CheckTypeSub – метод, содержащий основной код, который нужен для определения типа данных.

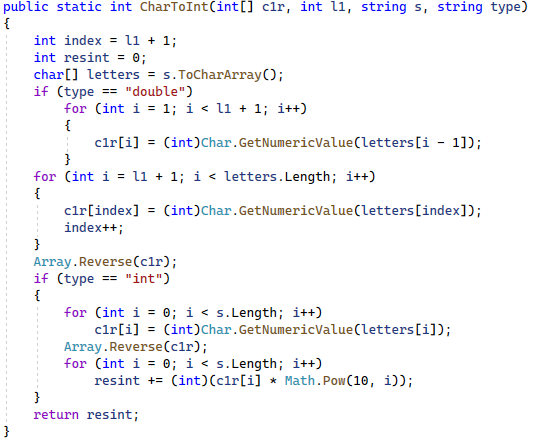


**Метод Converter**

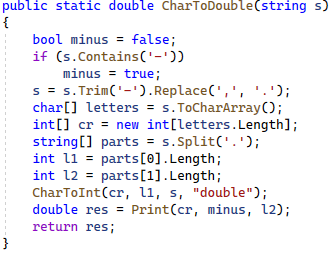


Данный метод получает указание на тип данных, содержащийся в строке, и в соответствии с ним вызывает определенные функции, такие как:

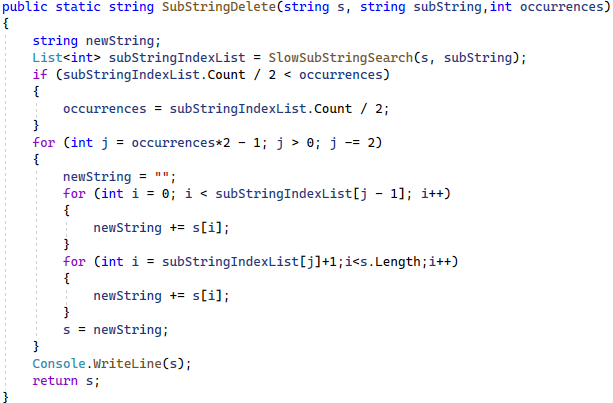
1. CharToInt – функция, предназначенная для конвертации строки в целые числа



1. CharToDouble – функция, предназначенная для конвертации строки в вещественное число



**Метод SubStringDelete**



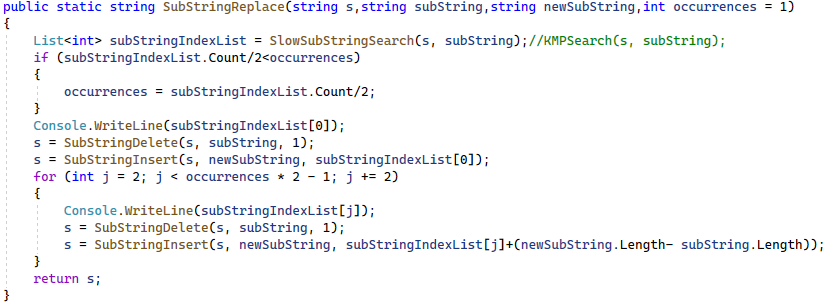
Данный метод служит для удаления подстроки

**Метод SubStringInsert**



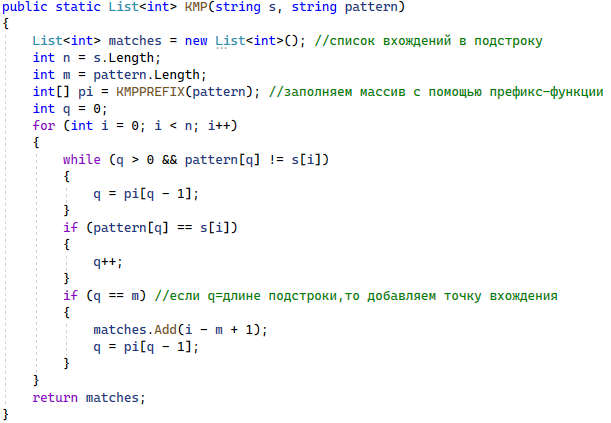
Данный метод служит для поиска подстроки

**Метод SubStringReplace**

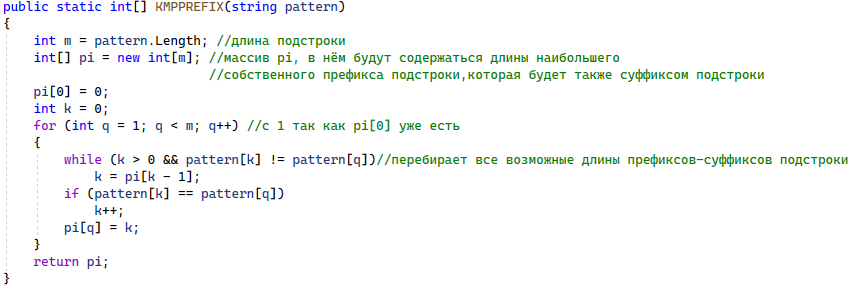


Данный метод, при помощи вызова функций, описанных выше, реализует замену одной подстроки другой подстрокой.

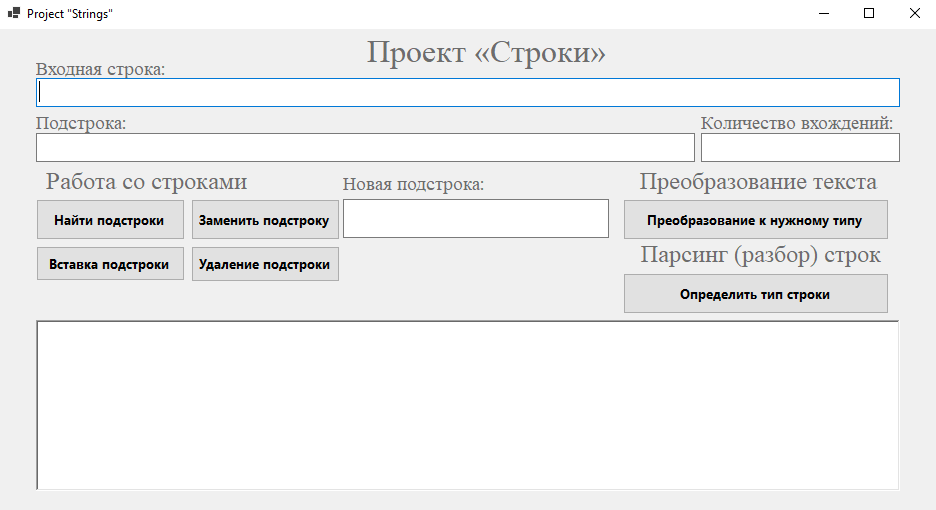
**Метод KMP**

****

Данный метод представляет собой эффективный алгоритм для нахождения точек входа в подстроку. Также вызывает вспомогательную функцию KMPPREFIX служащую для вычисления префикса функции. Ее код представлен на изображении снизу.



**Интерфейс**

****

Над проектом “строки” работали все участники команды. Каждый выполнял следующие обязанности:

|  |  |
| --- | --- |
| Разработка библиотеки | Кириллов Владислав  Богданов Никита  Калинин Павел |
| Разработка интерфейса | Виноградов Глеб  Невзоров Павел |
| Разработка тестов | Миронова Анна |

Кириллов Владислав и Невзоров Павел

# **Заключение**

В процессе выполнения проектов, перечисленных в данной курсовой, наша команда приобрела основные навыки, нужные при разработке программ, такие как умение разбивать сложную задачу на более простые для удобной работы над ней и простого доказательства корректности отдельных методов. Работа с декомпозицией позволяет значительно упростить сложный проект и сделать его более подходящим для работы в команде. Также мы научились строить простые, приятные глазу и интуитивно понятные пользователю интерфейсы и научились тестированию проектов, что является основным способом обеспечения корректности работы программы.

Хочется отметить, что помимо этих навыков самым важным опытом стала работа в команде. В процессе совместной работы на протяжении всего семестра мы поняли важность таких навыков как написание хорошо читаемого кода, снабжение комментариями основных частей программы, работа с системой контроля версий. Всё это не так сильно влияет на написание проекта, когда работа над ним предоставлена самому себе, но при командной разработке наличие этих навыков сильно влияет на корректность и скорость.

На скорость разработки также сильно сказывается способность команды к правильному тайм-менеджменту и распределению обязанностей, все члены команды по возможности должны работать параллельно для обеспечения максимальной скорости разработки и предоставления законченного проекта к назначенному времени.

В заключение хочется сказать, что наша команда получила не только индивидуальные фундаментальные навыки программирования, но и опыт работы в команде, важный для работы в сфере айти.

Калинин Павел

# **Ссылки на проекты**

1. [Floyd task](https://github.com/kindergartened/FloydTask.git)
2. [Stroki](https://github.com/kindergartened/Stroki.git)
3. [ConvertingNumericSystems](https://github.com/Pavelavl/ConvertingNumericSystems.git)