Модуль ввода вывода

Анализ и постановка задачи

На данном этапе нам необходимо входной файл разбить на лексемы, в соответствии с синтаксисом языка, а именно его ключевыми словами и специальными символами, определив для каждой лексемы её тип и сохраняя их в определённый буфер.

Проектирование

Концепция данного модуля заключается на обработке отдельных групп символов. Набор всех допустимых символов, был разбит на 4 подмножества.

1. Множество букв в нижнем и вернем регистре и нижнее подчёркивание
2. Множество цифр
3. Множество арифметических символов {“+”, ”-“, “/”, “\*”, “(“, “)” }
4. Множество специальных символов { ";", ":", "=", ",", ".", ":=", "{", "}" }

А так же выделены группы специальных слов

Системные { "begin", "var", "end", "program", "if", "else", "then", "for", "while" }

Арифметические слова { "div", "mod" }

Для работы предполагается система считывающего модуля и буфера символов.

Проводится анализ того что считано и что находится в буфере на данный момент.

Зная правила синтаксиса языка, мы можем ввести в модуль ввода вывода несколько правил:

1. Если после ввода ряда символов 1 или 2 типа на вход приходят символы 3 или 4 типа значит в буфере находится или идентификатор или специальное слово или константа
2. Если в буфере есть символ типа 3 значит в буфере уже полная лексема
3. Знак перед константой будем считать операцией
4. Все операции с переменными заканчиваются спец символом 4 типа “;”
5. Идентификатор может состоять из символов 1 или 2 группы

Для получения идентификатор, констант и спец. слов будем пользоваться этими правилами.

То есть вводить символы 1 и 2 группы до тех пока не встретим пробел или символы 3 или 4 группы, отправляя их в буфер.

И за тем проверяя, чем именно является буфер чтения, отправляя его значение в буфер хранения.

А считанный символ 3 или 4 группы отправляем в буфер чтения.

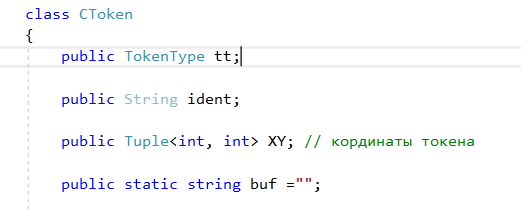
Потом так же вводя символы до тех пор, пока не появится символ 1,2,3 группы.

Повторяем эти действия, пока не кончится файл.

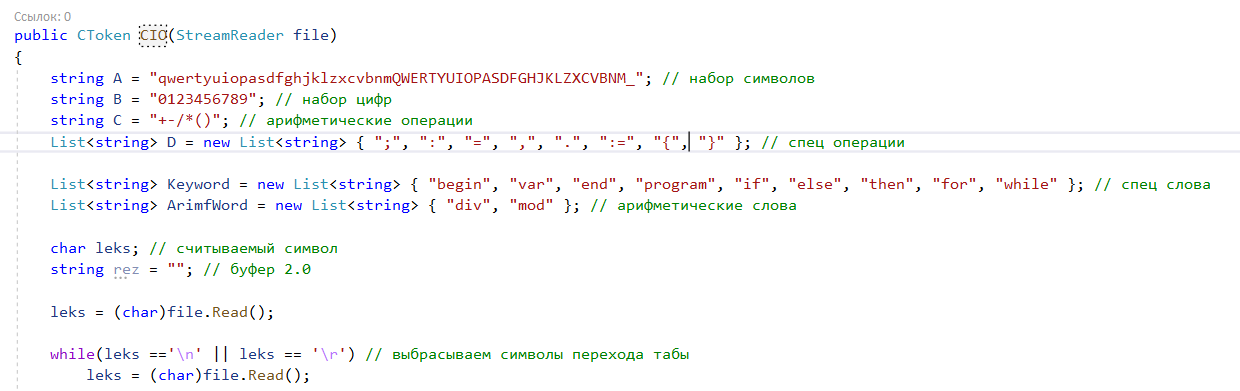
Разработка

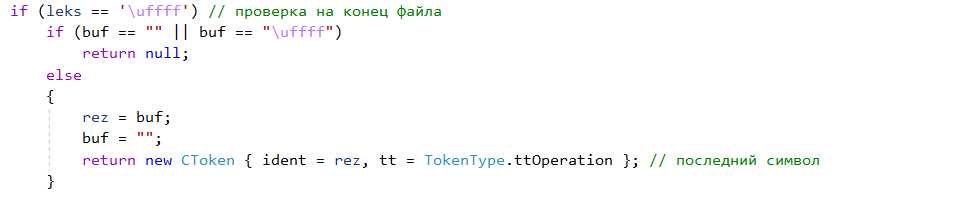
Для разработки был выбран язык С#.

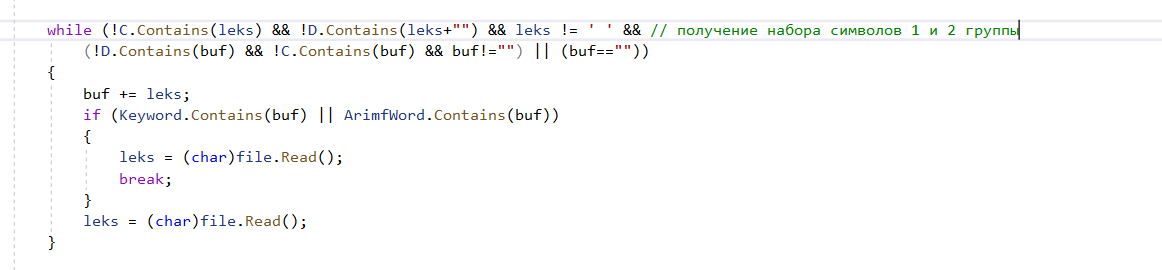
Реализация токена проводилась через объект CToken



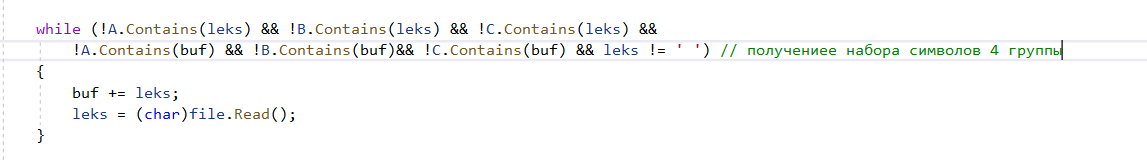
И метод для получения токенов

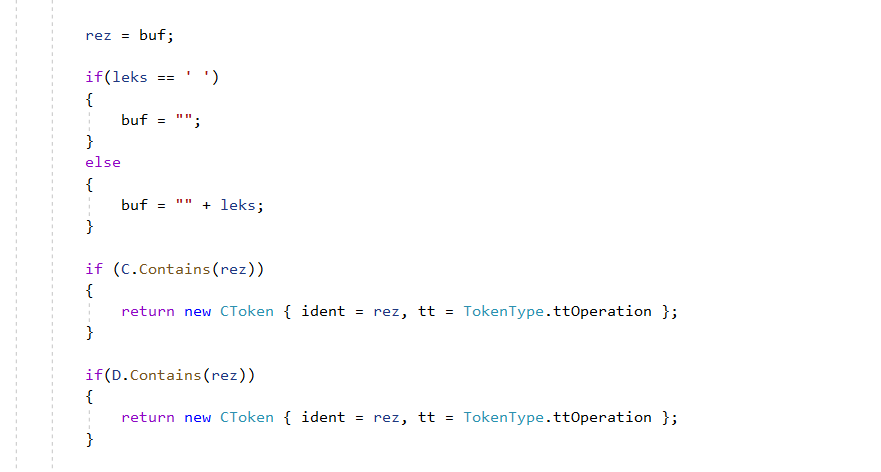


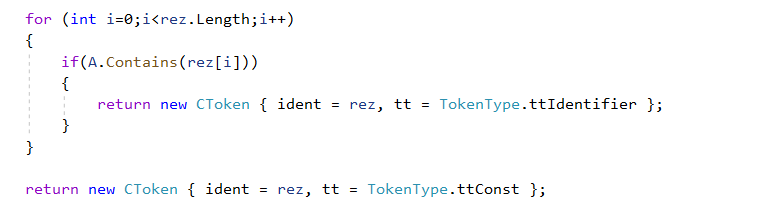










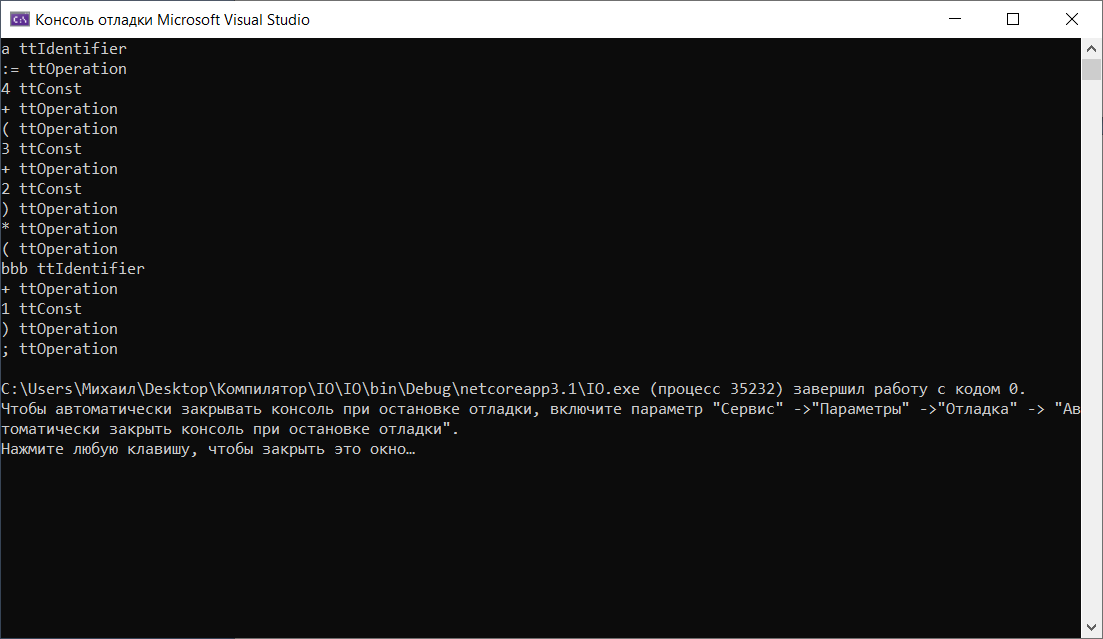


На данном этапе в объект были заложены поля координат токена это будет необходимо в дальнейшем для работы лексического анализатора.

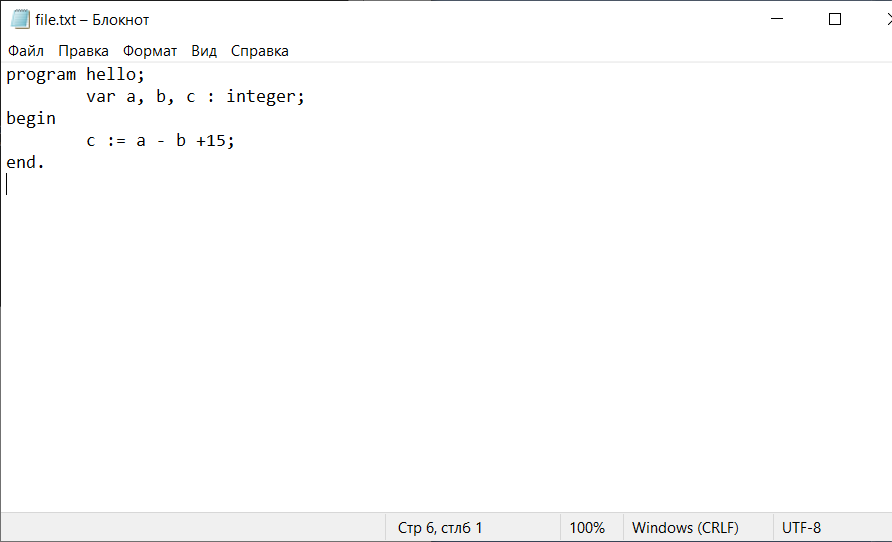
Тестирование

Строка с арифметическим выражением

Входные данные: a:=4+(3+2)\*(bbb+1);

Выходные данные: 

Полная программа

Входные данные: 

Выходные данные:

