Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Алтайский государственный технический университет им. И. И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с оценкой \_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Отчет

По лабораторной работе №4

**«Модификация программ из предыдущих лабораторных работ»**

по дисциплине «Программирование – 3 семестр»

Студент группы ПИ-32Б Самохвалов Даниил Алексеевич

Преподаватель доцент, Старовойтов Никита Александрович - ассистент

Барнаул 2024

**Задание:**

В проектах на C++ и Java

1. Придумать для чего хотя бы в одном вашем классе может быть разумным использование статического поля и статического метода;
2. Создать эти статические поля и методы;
3. Продемонстрировать их использование;

В проекте на C++

1. Продемонстрировать возврат значения из метода через указатель (\*) и через ссылку (&);
2. Продемонстрировать разумное использование оператора this;
3. Придумать и реализовать разумное использование дружественной функции;
4. Выполнить перегрузку операторов '+', '++' (два варианта, префиксный и постфиксный);
5. Заменить массивы char на std::string, продемонстрировать работу с этим классом;

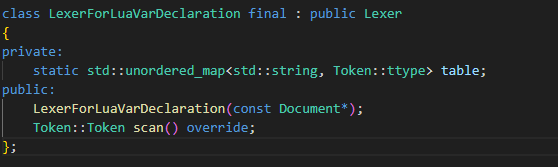
В проекте на Java

1. Продемонстрировать работу с массивом объектов;
2. Продемонстрировать возврат целочисленного значения из метода через вспомогательный класс;
3. Продемонстрировать разумное использование оператора this;
4. Продемонстрировать обработку строк (String);

Часть C++

1. Придумать для чего хотя бы в одном вашем классе может быть разумным использование статического поля и статического метода;
2. Создать эти статические поля и методы;

Статическое поле класса.

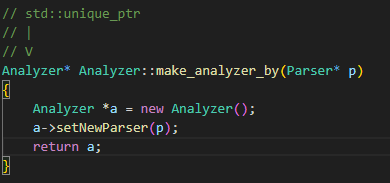


Примечание. Неважно сколько объектов этого класса будет, для всех этих объектов есть единая таблица с ключевыми словами. Использование static обусловлено повышением читабельности и оптимизацией, так как таблица будет одна для всех лексеров этого типа, что и логично.

Статический метод класса.

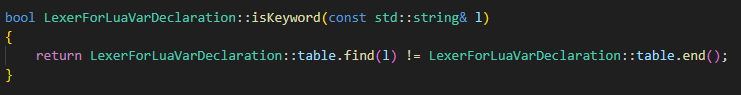
1)





2)



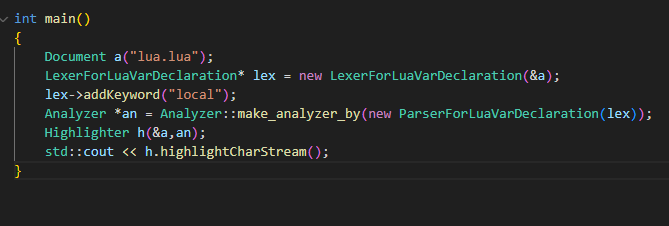


Примечание. 1) **make\_analyzer\_by** Этот метод по сути можно заменить конструктором, но тогда пользователь столкнется с достаточно неудобным синтаксисом (new Analyzer(Parser\*)), -использование new влечет за собой вызов delete(кстати, не всегда корректный). В нашем случае, метод может возвращать RAII-объект, в частности, std::unique\_ptr. При таком решении вызов new будет хорошо скрыт от пользователя, а о вызове delete позаботиться std::unique\_ptr.

2) **isKeyword** вспомогательная функция для проверки есть ли такое ключевое слово в таблице.

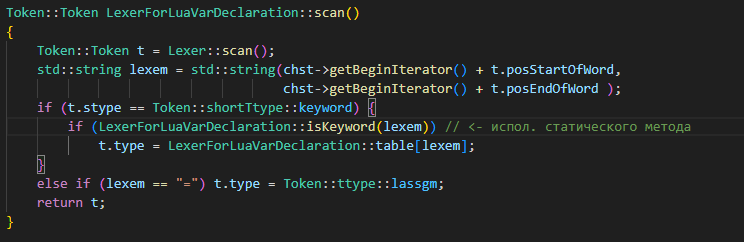
1. Продемонстрировать их использование;

1) **make\_analyzer\_by**

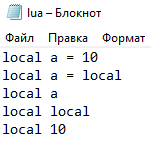


***Вместо new можно использовать такие же статические методы.***

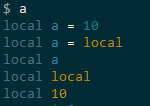
2) **isKeyword**



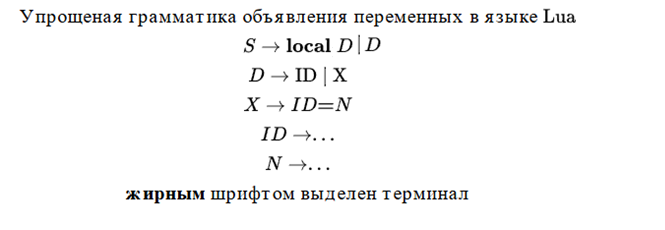
Файл



Вывод:

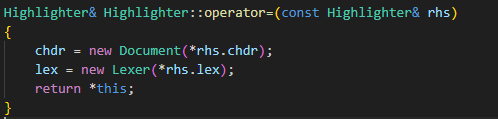


***Желтым цветом подкрашивается синтаксическая ошибка.***



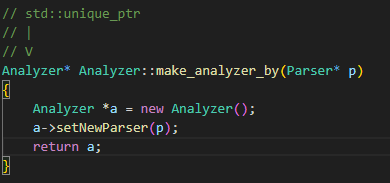
1. Продемонстрировать возврат значения из метода через указатель (\*) и через ссылку (&);
2. Продемонстрировать разумное использование оператора this;

Возврат значения из метода через ссылку (&);



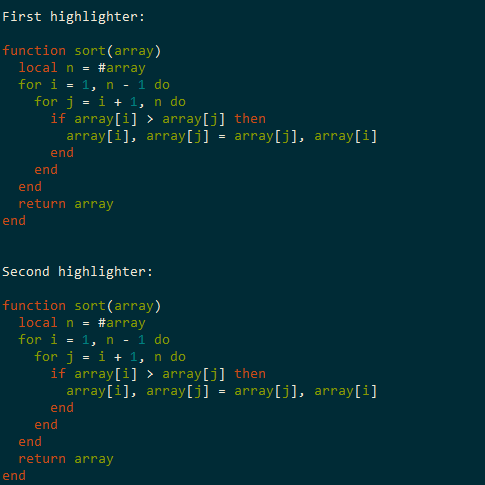
*Возврат this в данном контексте объясняется возможностью создавать цепочки присваиваний объекта класса. (a=a1=a2=…=an)*

Возврат значения из метода через указатель (\*);



*Пример использования оператора присваивания.*

**

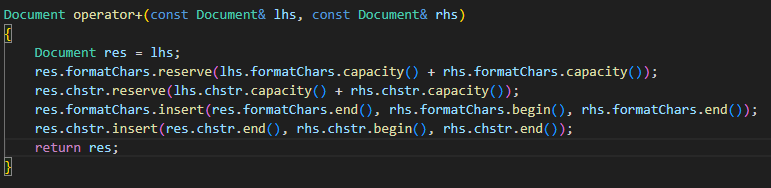


1. Придумать и реализовать разумное использование дружественной функции;



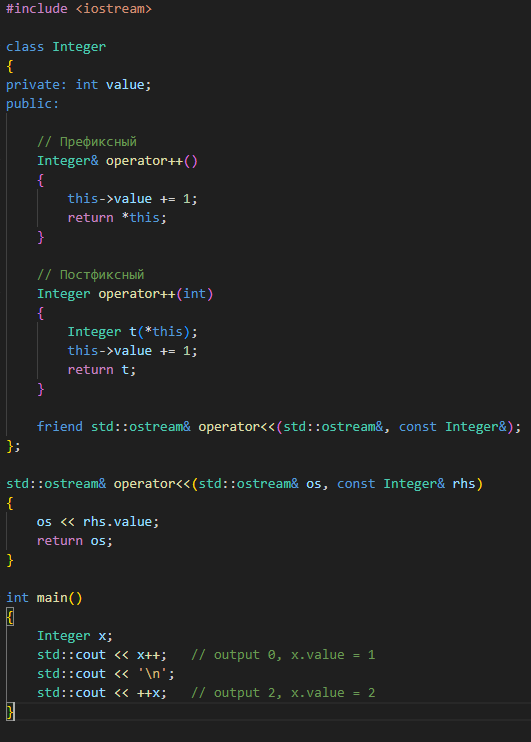
1. Выполнить перегрузку операторов '+', '++' (два варианта, префиксный и постфиксный);

Реализация оператора + для пользовательского типа.



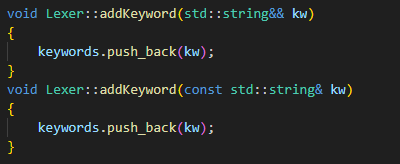
Соединяет содержимое двух файлов.

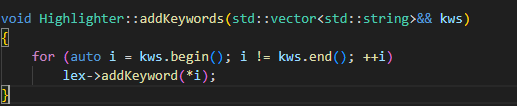
Реализация оператора инкремента для пользовательского типа.



1. Заменить массивы char на std::string, продемонстрировать работу с этим классом;





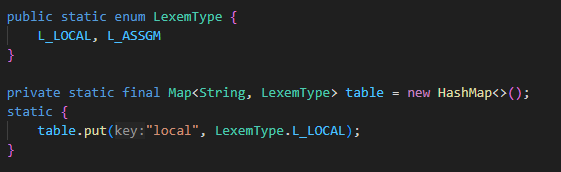


*std::string является замечательной заменой Си-строк, которые представляли из себя массив символов, заканчивающийся символом ‘\0’. В случае std::string, а точнее basic\_string<char>, мы имеем множество методов для работы со строками, реализующие алгоритмы сравнения, поиска подстроки и т.д.*

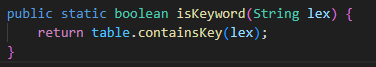
Часть Java

1. Придумать для чего хотя бы в одном вашем классе может быть разумным использование статического поля и статического метода;
2. Создать эти статические поля и методы;

Статическое поле класса.

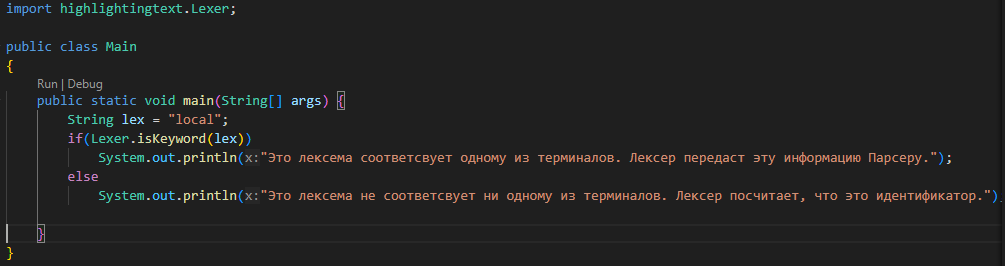


Статический метод класса.

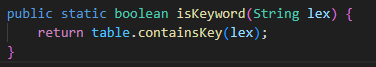


1. Продемонстрировать их использование;

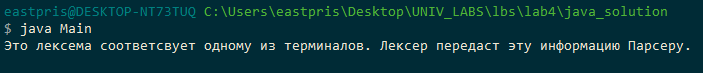
Использование isKeyword



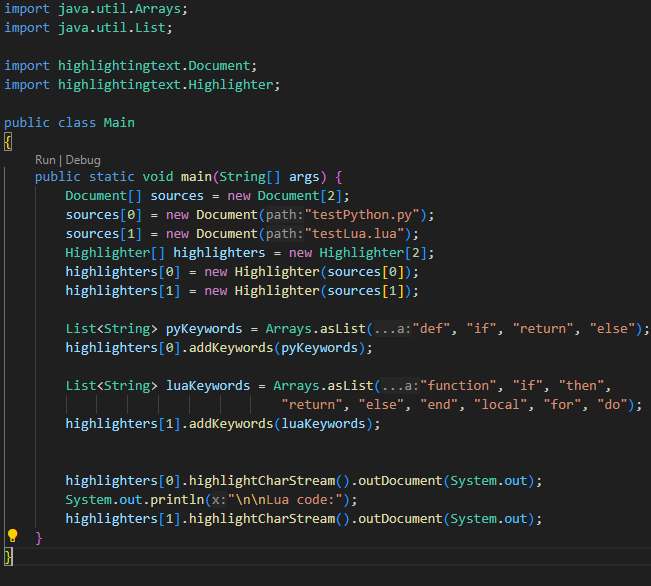
Использование table



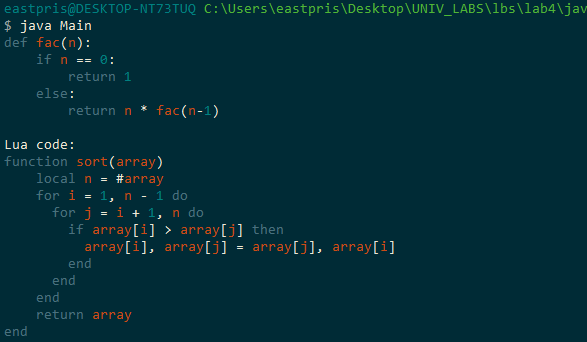
Вывод



1. Продемонстрировать работу с массивом объектов;

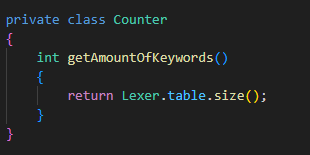


Вывод

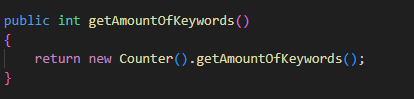


1. Продемонстрировать возврат целочисленного значения из метода через вспомогательный класс;

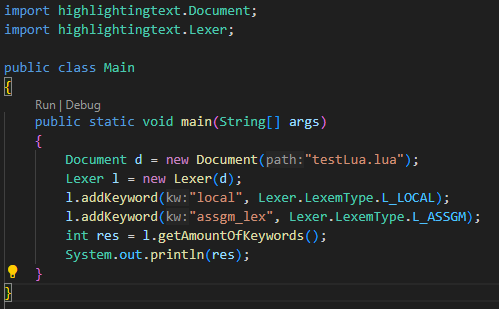
Класс, который считает.



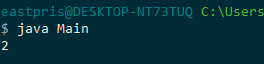
Делегируем обязанность на другой класс.



От лица пользователя:



Вывод:

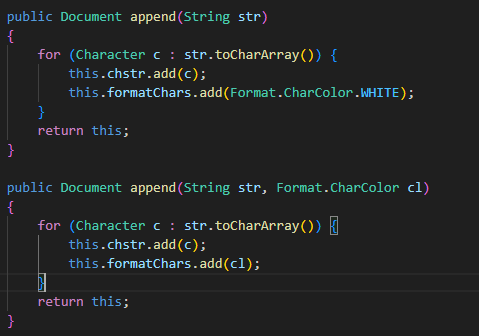


1. Продемонстрировать разумное использование оператора this;

*Использовать this можно как для указания, того, что используется конкретное поле, конкретного класса, в методе(нестатического) которого происходит действие.*

*А также для реализации цепочки вызовов.*

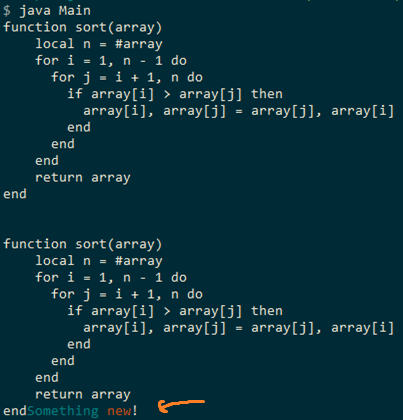
Ниже привел оба использования:



Со стороны пользователя:

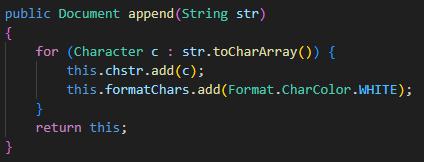


Вывод

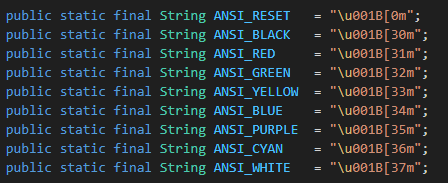


*К документу было добавлено 3 строки с разными цветами.*

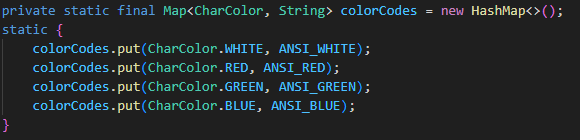
1. Продемонстрировать обработку строк (String);
2. Здесь реализован посимвольный проход по строке



1. Здесь определены объекты класса String



1. Здесь определена хэш-таблица с кодом цвета и его строковым эквивалентом в ANSI



1. Здесь получаем подмножество chstr от индекса it до forwardIt, получаем поток и преобразуем каждый его элемент в строку и соединяем в одну строк.

