Содержание

[Содержание 1](#_Toc97470531)

[1 Модуль ввода-вывода 2](#_Toc97470532)

[1.1 Описание 2](#_Toc97470533)

[1.2 Тестирование 3](#_Toc97470534)

[2 Лексический анализатор 4](#_Toc97470535)

[2.1 Описание 4](#_Toc97470536)

[2.1.1 CLexer 4](#_Toc97470537)

[2.1.2 CToken 4](#_Toc97470538)

[2.1.3 CConstToken 4](#_Toc97470539)

[2.1.4 CVariant 4](#_Toc97470540)

[2.1.5 CIntVariant, CRealVariant, CStringVariant, CBooleanVariant 5](#_Toc97470541)

[2.1.6 CIdentToken 5](#_Toc97470542)

[2.1.7 CKeyWordToken 5](#_Toc97470543)

[2.1.8 CEmptyToken 5](#_Toc97470544)

[2.2 Тестирование 5](#_Toc97470545)

1. Модуль ввода-вывода
   1. Описание

Представляет модуль, позволяющий считывать содержимое программы и сохранять список ошибок в файл, если таковые имеются.

Содержит поля указателей на потоки ввода и вывода, текущего номера строки, номера символа строки, буфера текущей обрабатываемой строки. Включает в себя вектор указателей на объекты ошибок наследуемого от std::exception класса CError, который содержит поля номера строки и номера символа, а также перечисление типа ошибки.

В качестве указателей используются unique\_ptr.

Для того, чтобы учесть различные способы хранения исходной программы, в качестве потока ввода будем использовать istream.

Открытые методы:

char GetNextChar() – получить следующий символ программы;

void AddError(EErrorType eType) – добавить новую ошибку в список ошибок, вызвать throw;

void PrintErrors() – вывести список ошибок; проверка на конец файла;

bool IsEndOfInput() – проверка окончания потока ввода;

string GetOutputString() – получить строку вывода (если для вывода был выбран вывод в строку, в других случаях возвращает пустую строку).

Закрытые методы:

void ReadString() – считать новую строку из файла в буфер.

Конструкторы класса модуля ввода-вывода:

CIO(ifstream& input) – ввод из файла, вывод в строку;

CIO(string input) – ввод из строки, вывод в строку;

CIO(ifstream &input, ostream &output) – ввод из файла, вывод ошибок в переданный поток;

CIO(string input, ostream &output) – ввод из строки, вывод ошибок в переданный поток.

Так как при вводе-выводе из файла или выводе в консоль не происходит выделения памяти, появляется необходимость ограничения её очищения, потому что в остальных случаях (чтение и вывод в строку) память выделяется. Для этого был создан деструктор, освобождающий указатели unique\_ptr, если ввод-вывод происходит из файла, во избежание освобождения памяти.

Метод получения следующего символа построчно считывает данные из входного потока, временно храня их в буфере, и возвращает следующий символ.

Метод добавления новой ошибки получает на вход тип ошибки и добавляет в вектор ошибок новую ошибку с переданным типом и с текущими в модуле ввода-вывода номерами символа и строки. Вызывает ошибку с помощью throw.

Метод вывода списка ошибок построчно выводит все ошибки из в поток вывода.

Метод чтения новой строки записывает в буфер новую строку, считанную с помощью функции getline(), также добавляет к считанной строке символ переноса.

Метод проверки на окончание файла возвращает значение false при условии, что поток достиг конца файла и номер символа буфера равен размеру буфера.

Метод получения строки вывода возвращает все данные, записанные в поток вывода при условии его использования.

* 1. Тестирование

Был создан класс IOUnitTest для unit тестов модуля ввода-вывода. Проверка на корректность осуществлялась с помощью функций Assert::AreEqual и Assert::IsTrue.

Методы тестирования:

IFileOString – проверка работы модуля ввода-вывода, где ввод происходил из файла, а вывод в строку;

IStringOString – проверка работы модуля ввода-вывода, где ввод и вывод происходил из строки;

IFileOFile – проверка работы модуля ввода-вывода, где ввод и вывод происходил из файла;

IStringOFile – проверка работы модуля ввода-вывода, где ввод происходил из строки, а вывод в файл;

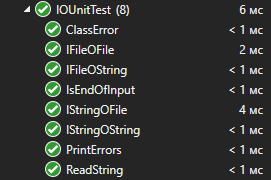
В каждом из перечисленных выше методов был считан один символ и выведена одна ошибка. Результаты сравнивались с прогнозируемыми.

ReadString – проверка считывания нескольких строк;

PrintErrors – проверка вывода нескольких ошибок;

IsEndOfInput – проверка работы метода IsEndOfInput класса CIO;

ClassError – проверка данных, хранящихся в классе CError.



Результат выполнения тестов класса IOUnitTest

1. Лексический анализатор
   1. Описание
      1. CLexer

Класс CLexer представляет модуль лексического анализатора, который позволяет получать токены из указанного модуля ввода-вывода.

Содержит поля указателя на модуль ввода-вывода, последнего считанного символа и словаря имен ключевых слов (ключами является элементы перечисления ETokenType).

Открытые методы:

CIO\* GetIOPtr() – получить следующий токен;

unique\_ptr<CToken> – получить модуль ввода-вывода.

Содержит единственный конструктор принимающий на вход указатель на модуль ввода-вывода.

Метод получения следующего токена считывает необходимую последовательность символов из потока ввода и на её основе строит объект одного из производных классов класса CToken – CConstToken, CIdentToken, CKeyWordToken, CEmptyToken.

Метод получения модуля ввода-вывода возвращает указатель на модуль ввода-вывода, содержащегося в лексическом анализаторе.

* + 1. CToken

Абстрактный класс CToken представляет токен программы на языке Pascal. Содержит поле перечисления типа токена ETokenType (ttConst – константа, ttIdent – идентификатор, ttKeyword – ключевое слово).

Содержит конструктор принимающий тип токена.

Открытые методы:

ETokenType GetType() – получение типа токена;

virtual string ToString() = 0 – приведение к строке.

Также содержит виртуальный деструктор.

* + 1. CConstToken

CConstToken – класс токена константы, является производным от CToken. Включает в себя указатель на объект класса CVariant, который характеризует тип и значение константы. Содержит переопределенный метод ToString() и метод CVariant\* GetVariant(), возвращающий указатель на CVariant;

* + 1. CVariant

Абстрактный класс CVariant, представляющий константу. Содержит поле типа константы перечисления EVariantType (vtInt – целая, vtReal – вещественная, vtString – строковая, vtBoolean – булева).

Содержит конструктор принимающий тип константы.

Открытые методы:

EVariantType getVariantType() – получение типа константы;

virtual string ToString() = 0 – приведение к строке.

Также содержит виртуальный деструктор.

* + 1. CIntVariant, CRealVariant, CStringVariant, CBooleanVariant

CIntVariant, CRealVariant, CStringVariant, CBooleanVariant – классы константы, являются производным от CVariant. Содержат поле значения константы.

Открытые методы:

GetValue() – получение значения;

string ToString() – переопределенный метод приведения к строке.

* + 1. CIdentToken

CIdentToken – класс токена идентификатора, является производным от CToken. Включает в себя имя идентификатора. Содержит переопределенный метод ToString().

* + 1. CKeyWordToken

CKeyWordToken – класс токена ключевого слова, является производным от CToken. Включает в себя поле ключевого слова перечисления EKeyWordType, которое содержит элементы для всех возможных ключевых слов языка Pascal.

Открытые методы:

EKeyWordType GetKeyWordType () – получение типа ключевого слова;

string ToString() – переопределенный метод приведения к строке.

* + 1. CEmptyToken

CEmptyToken – класс отсутствия токена, является производным от CToken.

Необходим в тех ситуациях, когда вернуть токен другого типа не удается: окончание потока ввода, при наличии в нем комментариев. Содержит переопределенный метод ToString().

* 1. Тестирование

Был создан класс LexerUnitTest для unit тестов лексического анализатора. Проверка на корректность осуществлялась с помощью функций Assert::AreEqual и Assert::IsTrue.

Методы тестирования:

OneSymbolKeyWordToken – проверка возвращаемых лексическим анализатором токенов односимвольных ключевых слов.

TwoSymbolsKeyWordToken – проверка возвращаемых лексическим анализатором токенов ключевых слов, которые распознаются после считывания дополнительного символа.

WordKeyWordToken – проверка возвращаемых лексическим анализатором токенов ключевых слов, которые являются цифробуквенными словами.

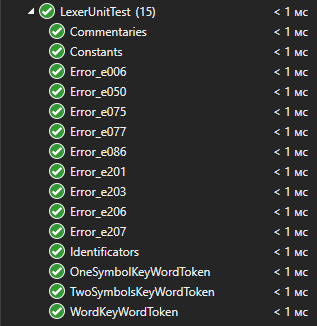
Commentaries – проверка пропуска комментариев.

Constants – проверка считывания констант различных типов.

Identificators – проверка считывания идентификаторов.

ErrorTestOutput – не тестируемый метод, возвращающий строку потока вывода на основании строки ввода, содержащую один токен с ошибкой.

Методы Error\_e006, Error\_e050, Error\_e075, Error\_e077, Error\_e086, Error\_e201, Error\_e203, Error\_e206, Error\_e207 проверяют содержимое потока вывода при получении соответствующей лексической ошибки.



Результат выполнения тестов класса LexerUnitTest