**Государственный комитет Российской Федерации по телекоммуникациям**

**Сибирский государственный университет телекоммуникаций и информатики**

**КУРСОВАЯ РАБОТА**

**По дисциплине “Технологии разработки программного обеспечения”**

**Вариант № 4**

Работу выполнил:

                                                                   студент 1 курса

                                                                   Группы:

ПБТ-01

                                                                   Чульдум Саян Андреевич

                                                                   Работу проверил:

### Остапкевич Михаил Борисович

### Работа защищена:

                                                                   «\_\_\_» \_\_\_\_\_\_\_\_\_\_2020г.

                                                                                                                                                                                      С оценкой «\_\_\_\_\_\_\_»

**г. Новосибирск**

**2020**

**СОДЕРЖАНИЕ**

1. Формирование требований
2. Проектирование
   1. Выбор платформы
   2. Выбор языка программирования
   3. Проектирование алгоритма
   4. Модули программы
3. Реализация
   1. Программирование
   2. Документирование – разработка модели
   3. Документирование
4. Оценка производительности

Выводы

Список используемой литературы

1. **ФОРМИРОВАНИЕ ТРЕБОВАНИЙ**

Необходимо разработать шестнадцатеричный калькулятор, который принимает входные значения по имени файла. Входными данными являются выражения с инфиксной формой записи. Программа должна быть совместима с 86-ти разрядной Windows 10.

Программа должна выполнять следующие основные действия:

1. считывать текстовый файл по его имени, файл содержит выражения в шестнадцатеричной системе счисления, файл может содержать несколько выражений, каждая в отдельной строке;
2. открывать исходный файл, обрабатывая возможные ошибочные ситуации;
3. в случае ошибок в выражении, выводить сообщение об ошибке;
4. производит вычисления;
5. печатать результаты вычисления в отдельный файл в шестнадцатеричной системе счисления и завершать работу.

Выражение для калькулятора – это:

* беззнаковая шестнадцатеричная константа;
* (выражение);
* выражение + выражение;
* выражение – выражение;
* выражение \* выражение;
* выражение / выражение;
* not(выражение) – операция побитовой инверсии;
* and(выражение, выражение) – операция побитового И;
* or(выражение, выражение) – операция побитового ИЛИ;
* xor(выражение, выражение) – операция исключающего побитового ИЛИ.

Для задания 4 ось X – размер входного файла со строкой выражения.

1. **ПРОЕКТИРОВАНИЕ**
   1. **Выбор платформы**

Согласно требованиям заказчика, программу необходимо реализовать под операционную систему Windows.

* 1. **Выбор языка программирования**

Для написания кода программы, будет использован язык программирования C, как наиболее подходящий для данной задачи, как и наиболее известный заказчику. Среда программирования – Microsoft Visual Studio Community 2019 version 16.8.3.

* 1. **Проектирование алгоритма**

По запросу заказчика запишем конструкцию выражения для шестнадцатеричного калькулятора по расширенной форме Бэкуса-Наура (рБНФ). При разработке алгоритма будем основываться на данной конструкции:

Выражение = Слагаемое {Сложение Слагаемое}.

Слагаемое = Множитель {Умножение множитель}.

Множитель = Число | Выражение | (Инверсия Выражение) |

Побит\_оп (Выражение, Выражение).

Сложение = “+” | “-”.

Умножение = “\*” | “/”.

Инверсия = not.

Побит\_оп = and | or | xor.

Число = Цифра {Цифра}.

Цифра = “0” | “1” | “2” … |”9” | “A” | “B” … | “F”.

Над самим выражением мы проведем лексический и синтаксический анализы. За лексический анализ отвечает следующая часть конструкции:

Сложение = “+” | “-”.

Умножение = “\*” | “/”.

Инверсия = not.

Побит\_оп = and | or | xor.

Число = Цифра {Цифра}.

Цифра = “0” | “1” | “2” … |”9” | “A” | “B” … | “F”.

Выражение разбивается на лексемы, тип которых определяется исходя из вышеперечисленных требований.

За лексический анализ будет отвечать функция void getToken(void). Лексема, которая была отобрана из выражения записывается в глобальный символьный массив token, а его тип будет записан в char tokType.

Определим три типа лексем:

DELIMITER = Сложение | Умножение | “,” | “(” | “)”.

IDENTIFIER = Инверсия | Побит\_оп.

NUMBER = Число.

Для удобства лексического анализа создадим функцию void getExpr(FILE \*fp, char \*pStr), которая убирает из прочтенной строки пробельные символы и заменяет буквы на прописные.

За синтаксический анализ отвечает остальная часть рБНФ конструкции:

Выражение = Слагаемое {Сложение Слагаемое}.

Слагаемое = Множитель {Умножение множитель}.

Множитель = Число | Выражение | (Инверсия Выражение) |

Побит\_оп (Выражение, Выражение).

Синтаксический анализ будет основан на методе рекурсивного спуска. За каждый терминал, стоящий слева от “=” будет отвечать отдельная функция.

Код лексического и синтаксического анализа будет реализован в отдельном модуле analysis.h