МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное   
учреждение высшего образования

**"Южно-Уральский государственный университет**

**(национальный исследовательский университет)"**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

**ОТЧЕТ   
по учебной практике**

бакалавра направления 09.03.04 "Программная инженерия"

|  |  |
| --- | --- |
|  | Выполнил:\_\_\_\_\_\_\_\_\_\_\_\_ студент группы КЭ-143  Д.С. Чистяков  Проверил:\_\_\_\_\_\_\_\_\_\_\_\_\_  преподаватель кафедры СП  A.A. Сахарова  Дата:\_\_\_\_\_\_\_\_, Оценка:\_\_\_\_\_\_\_\_\_\_ |

Челябинск, 2023 г.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное   
учреждение высшего образования

**«Южно-Уральский государственный университет**

**(национальный исследовательский университет)»**

**Высшая школа электроники и компьютерных наук**

**Кафедра системного программирования**

УТВЕРЖДАЮ

Зав. кафедрой

системного программирования

\_\_\_\_\_\_\_\_\_\_\_\_\_ Л.Б. Соколинский

**ЗАДАНИЕ**

**по** **учебной практике**

1. **Цель работы**

Необходимо разработать распознаватель заданной символьной цепочки. Символьная цепочка задается с помощью формул Бэкуса-Наура.

1. **Исходные данные к работе**
2. *Йенсен К., Вирт Н.* Паскаль. Руководство пользователя и описание языка. М.: Компьютер, 1995.
3. *Льюис Ф., Розенкранц Д., Стирнз Р.* Теоретические основы проектирования компиляторов. М.: Мир, 1979.
4. **Перечень подлежащих разработке вопросов**
5. Выполнить анализ требований и разработать спецификации программы.
6. Провести проектирование программы.
7. Реализовать спроектированные модули.
8. Провести тестирование и отладку реализованных модулей.
9. **Сроки**

Дата выдачи задания: "26" июня 2023 г.

Срок сдачи законченной работы: "21" июля 2023 г.

**Руководитель:**

преподаватель кафедры СП \_\_\_\_\_\_\_\_\_\_\_\_\_ Сахарова A.A.

должность, ученая степень подпись ФИО преподавателя

**Задание принял к исполнению:**

\_\_\_\_\_\_\_\_\_\_\_\_\_ Чистяков Д.С.

подпись ФИО студента

**ОГЛАВЛЕНИЕ**

[1. СПЕЦИФИКАЦИЯ 4](#_Toc139283868)

[2. ПРОЕКТИРОВАНИЕ 6](#_Toc139283869)

[2.1. Модульная структура 6](#_Toc139283870)

[2.2. Интерфейсы модулей 7](#_Toc139283871)

[3. Кодирование 10](#_Toc139283872)

[3.1. Структура текста программы 10](#_Toc139283873)

# СПЕЦИФИКАЦИЯ

Необходимо разработать распознаватель заданной символьной цепочки. Символьная цепочка задается с помощью формул Бэкуса-Наура.

Например:

**<цепочка>::=<описание константы>;**

<значение>::=<целая константа> | <16-ричная константа> | <вещественная

константа>| <строковая константа>

<строковая константа>::='<цифра> | <идентификатор> |

<цифра><идентификатор>'

<целая константа>::=<целое со знаком> | <целое без знака>

<целое со знаком>::=<знак><целое без знака>

<целое без знака>::=<цифра> | <цифра><целое без знака>

<цифра>::=0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<знак>::=+ | -

<16-ричная константа>::= $<список 16-ричных букв и цифр>

<вещественная константа>::=<целое со знаком> | <вещественное со знаком> | <вещественное без знака>

<идентификатор>::= <буква> | <идентификатор><буква> | <идентификатор><цифра>

<буква>::=A | B | C | D | E | F | ... | Z

Помимо этого на цепочку накладывается следующее семантическое ограничение: идентификатор, входящий в цепочку, не должен совпадать с ключевыми словами языка Pascal [1].

**Описание входных данных**

Цепочка записана в текстовом файле INPUT.TXT, который состоит из одной строки. Длина цепочки не превышает 80 символов.

**Описание выходных данных**

Результат распознавания необходимо записать в текстовый файл OUTPUT.TXT в одно из следующих сообщений: ACCEPT, если цепочка допустима, и REJECT, если цепочка недопустима.

**Примеры входных и выходных данных**

В таблице 1 приведены примеры входных и соответствующих им выходных данных.

Таблица 1 – примеры входных и выходных данных

|  |  |
| --- | --- |
| **INPUT.TXT** | **OUTPUT.TXT** |
| const Error = $FFFF; | ACCEPT |
| const Eps = 1E-3; | ACCEPT |
| const OneHndNeg = -100; | ACCEPT |
| const Str = '0123456789abcdefgh'; | ACCEPT |
| var value = 01; | REJECT |
| const expression = -True; | REJECT |

# ПРОЕКТИРОВАНИЕ

В данном параграфе представлена информация о модульной структуре программы, содержащая описание интерфейсов между модулями. Кроме того, здесь представлена семантика модулей программы.

## Модульная структура

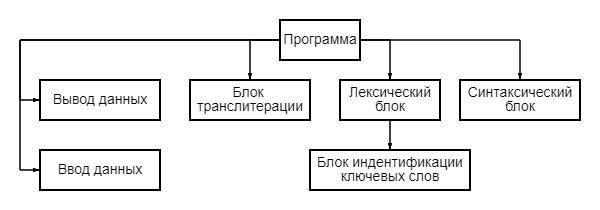


Рисунок 1 - Модульная структура

**Программа** – главный модуль, из которого происходит запуск программы и обработка входной цепочки.

**Блок работы с файлами** – содержит в себе модули для работы с файлами.

**Ввод данных** – считывает данные из файла INPUT.txt для дальнейшей работы с ними.

**Вывод данных** – записывает итоговый результат в файл OUTPUT.txt.

**Блок транслитерации** – подпрограмма, преобразующая исходную символьную цепочку в цепочку лексем вида ("символ цепочки", "класс символа цепочки ").

**Лексический блок** – подпрограмма, преобразующая цепочку лексем, полученную от транслитератора, в цепочку лексем вида ("символ входного языка", "класс символа входного языка").

**Блок идентификации ключевых слов** – подпрограмма, которая устанавливает, какое из ключевых слов языка Pascal соответствует заданному идентификатору, либо сообщает, что заданный идентификатор не является ключевым словом языка Pascal.

**Синтаксический блок** – подпрограмма, которая получает цепочку лексем вида ("символ входного языка", "класс символа входного языка") и устанавливает, соответствует ли она заданным формулам Бэкуса-Наура.

## Интерфейсы модулей

Программа:

Семантика: main.py:

Входные данные: -

Выходные данные: -

Модуль вызывает остальные подпрограммы.

Ввод данных:

Семантика: def data\_input()

Входные данные: -

Выходные данные: str – строка, считанная из файла INPUT.txt.

Вывод данных:

Семантика: def data\_output(result):

Входные данные: result – переменная, в зависимости от которой в файл output.txt выводится ACCEPT или REJECT.

Выходные данные: –

В зависимости от значения переменной result в файл OUTPUT.txt

записывается результат.

Блок транслитерации:

Семантика:def tansliteration(string):

Входные данные: string – строка, считанная из INPUT.txt.

Выходные данные: возвращает двумерный массив размера length \* 2 вида ["символ цепочки"] ["класс символа цепочки"], где length – длина исходной строки.

Например, символьную цепочку “const Error = $FFFF;” блок транслитерации должен преобразовать в двумерный массив лексем, пример представлен в таблице 2.

Таблица 2 – пример массива блока транслитерации

|  |  |
| --- | --- |
| c | буква |
| o | буква |
| n | буква |
| s | буква |
| t | буква |
|  | пробел |
| E | буква |
| r | буква |
| r | буква |
| o | буква |
| r | буква |
|  | пробел |
| = | равно |
|  | пробел |
| $ | доллар |
| F | буква |
| F | буква |
| F | буква |
| F | буква |
| ; | тчзпт |

Лексический блок:

Семантика: def Lexer(arrStr):

Входные данные: arrStr – двумерный массив, в котором находится строка, разбитая на символы с присвоенными им классами.

Выходные данные: resultWords – двумерный массив размера length \* 2

Блок отвечает за преобразование двумерного массива лексем, полученного от блока транслитерации, в двумерный массив лексем вида ["слово входного языка"] ["класс слова входного языка"]. Например, лексический блок должен выдать цепочку лексем, описанную в таблице 3.

Таблица 3 – пример массива лексического блока

|  |  |
| --- | --- |
| const | ИДЕНТИФИКАТОР |
| Error | ИДЕНТИФИКАТОР |
| = | равно |
| $ | доллар |
| FFFF | ЦЕЛОЕ |
| ; | тчкзпт |

Блок идентификации ключевых слов:

Семантика: def identification (string):

Входные данные: string – двумерный массив, идентификаторы которого просматриваются на совпадения с ключевыми словами языка Паскаль.

Выходные данные: arr – двумерный массив, идентификаторы, классы которых были изменены в случае совпадения с ключевыми словами языка Pascal. Блок устанавливает, присутствуют ли в двумерном массиве лексем ключевые слова языка Паскаль.

Синтаксический блок:

Семантика: def syntax(string):

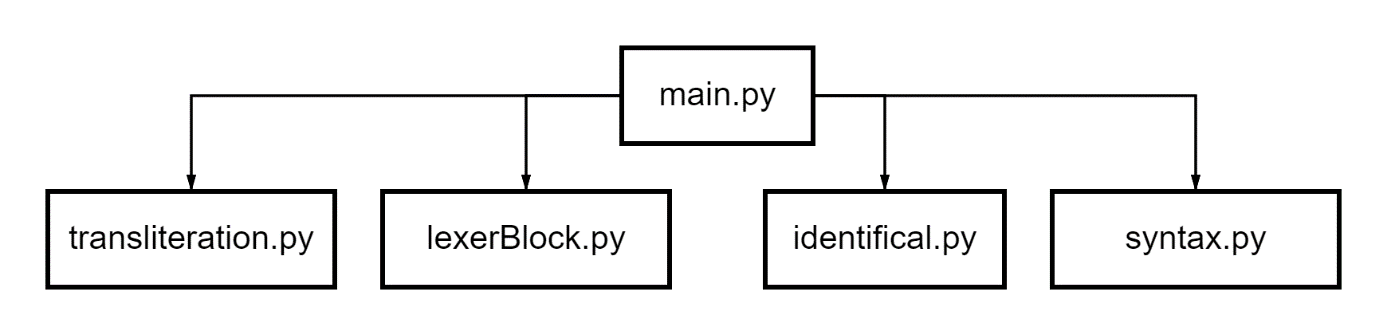
Входные данные: string – двумерный массив, содержащий лексемы и их классы.

Выходные данные: если данная цепочка не совпадает с приведенными в подпрограмме примерами будет возвращено REJECT, иначе – ACCEPT.

# Кодирование

На данном этапе описан процесс кодирования: структура текста программы, алгоритмы реализации модулей, размер текста программы. Алгоритмы реализации модулей поделены на четыре блока: блок транслитерации, лексический блок, блок идентификации и синтаксический блок. Программа написана на языке Python в программе PyCharm Community.

## Структура текста программы



main.py – головной модуль, содержащий функцию, по очереди вызывающую функции из каждого модуля.

transliteration.py – модуль транслитерации, содержащий основную функцию, которая принимает символьную цепочку string в качестве входных данных, считываемую из файла input.txt.

lexerBlock.py – лексический модуль, содержащий функцию, получающую на вход список классов «символ/тип символа», состоящий из исходной строки, разбитой на символы и возвращающий список классов «символ/тип символа», состоящий из слов входного языка и их типов.

identifical.py – модуль идентификации ключевых слов. Функция получает на вход список лексем, состоящих из слов входного языка и их типов, проверяет, есть ли среди слов входного языка ключевые слова языка Pascal, и если они присутствуют, меняет класс этих слов на само слово.

syntax.py – синтаксический модуль, содержащий функцию, получающую на вход список лексем, состоящих из слов входного языка и их типов, проверенный на наличие ключевых слов, и возвращающий значение 0 или 1 в зависимости от корректности цепочки.