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 02.03.02 "Фундаментальная информатика   
и информационные технологии"

|  |  |
| --- | --- |
|  | Выполнил:  студент группы: КЭ-102  Иконникова Анастасия Витальевна  Проверил:  Ст. преподаватель кафедры СП  Федянина Раиса Сулеймановна.  Дата:\_\_\_\_\_\_\_\_, Оценка:\_\_\_\_\_\_\_\_\_\_ |

Челябинск-2019

Министерство науки и высшего образования Российской Федерации

Южно-Уральский государственный университет

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: 1 июля 2019 г.

Срок сдачи законченной работы: 26 июля 2019 г.

**Руководитель:**

Ст. преподаватель кафедры СП \_\_\_\_\_\_\_\_\_\_\_\_\_ Р.С. Федянина

Подпись

**Задание принял к исполнению:**

\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Иконникова

Подпись

**ОГЛАВЛЕНИЕ**

[1. Спецификация 5](#_Toc327919175)

[2. Проектирование 5](#_Toc327919176)

[2.1. Модульная структура 5](#_Toc327919177)

[2.2. Интерфейсы модулей 5](#_Toc327919178)

[3. Кодирование 5](#_Toc327919179)

[3.1. Структура текста программы 5](#_Toc327919180)

[3.2. Алгоритмы реализации модулей 5](#_Toc327919181)

[3.2.1. Блок транслитерации 5](#_Toc327919182)

[3.2.2. Лексический блок 5](#_Toc327919183)

[3.2.3. Синтаксический блок 6](#_Toc327919184)

[3.2.4. Блок идентификации ключевых слов 6](#_Toc327919185)

[3.3. Размер текста программы (в строках) 6](#_Toc327919186)

[4. Тестирование 6](#_Toc327919187)

[4.1. Автономное тестирование 6](#_Toc327919188)

[4.2. Комплексное тестирование 6](#_Toc327919189)

[Заключение 6](#_Toc327919190)

[Литература 7](#_Toc327919191)

# Спецификация

В данном разделе следует поместить *формулировку задания*: определение символьной цепочки для распознавания, описание формата входных и выходных данных, примеры входных и соответствующих им выходных данных.

Вариант F

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

**<цепочка>::=<условный оператор>**

<условие>::=<идентификатор>

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

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

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

**<оператор1>::=<оператор присваивания>**

<оператор присваивания>::=<идентификатор>::=<выражение>

<выражение> ::= <вызов подпрограммы>

<вызов подпрограммы>::=<идентификатор>(<список параметров>);

<список параметров> ::= <идентификатор>

**<оператор2>::=<вызов подпрограммы>**

<список параметров> ::= <целая константа>

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

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

<знак>::=+ | -

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

<логическая константа> ::= TRUE | FALSE

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**

|  |  |
| --- | --- |
| **INPUT.TXT** | **OUTPUT.TXT** |
| const one=1; | ACCEPT |
| const Code=-23; | ACCEPT |
| const HEX=$10A; | ACCEPT |
| const N12=+5; | ACCEPT |
| const Code=-23 | REJECT |
| const Min=+-4; | REJECT |
| conct D5=$5F; | REJECT |
| const Max=4E; | REJECT |
| const n=$5H; | REJECT |

# Проектирование

Данный раздел следует разбить на следующие пункты:

На данном этапе проектирования необходимо выполнить проектирование модульной структуры программы и разработать набор тестов и соответствующие тестовые программы для проведения тестирования. Важной особенностью этапа проектирования является то, что все работы на данном этапе выполняются без использования системы программирования

## Модульная структура

В данном пункте следует поместить *рисунок со схемой модульной структуры* с кратким описанием назначения входящих в нее модулей.

В результате проведённого анализа требований была спроектирована мо-дульная структура программы. Программа включает 4 основных блока, 3 из ко-торых на первом уровне иерархии и 1 на втором уровне. Непосредственно из головного модуля вызывается блок транслитерации, лексический блок и син-таксический блок последовательно. Блок идентификации ключевых слов вызы-вается в процессе работы лексического блока. В головном модуле программы реализованы также функции для считывания входных данных и записи резуль-тата в выходной файл. Описание основных блоков приведено ниже, модульная структура программы приведена ниже на рисунке.

**Модульная структура программы:**

***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

***Блок транслитерации:*** разбивает полученную строку на символы и присваивает каждому символу свой класс (*буква, цифра, пробел и т.д.*).

*Входные данные*:

symbols - строка с исходной символьной цепочкой.

*Выходные данные*:

new\_list - список кортежей, где в каждом кортеже нулевой элемент является строкой с «символом цепочки», а первый элемент является строкой с «классом символа цепочки».

***Лексический блок:*** группирует символы из полученного списка кортежей в строки и присваивает каждой строке свой класс (*идентификатор, целое, равно и т.д.*).

*Входные данные:*

old\_list – список кортежей, полученный в результате работы блока транслитерации.

*Выходные данные:*

new\_list - список списков, где в каждом внутреннем списке нулевой элемент является строкой с «символом входного языка», а первый элемент является строкой с «классом символа входного языка».

***Блок идентификации ключевых слов:*** проверяет, является ли полученный идентификатор ключевым словом языка паскаль и возвращает его класс (*идентификатор, клслово\_const, др\_клслово*).

*Входные данные:*

identifier – строка, в который записан проверяемый идентификатор.

*Выходные данные:*

new\_class – строка, в которой записан класс проверяемого идентификатора.

***Синтаксический блок:*** проверяет, соответствует ли входной язык заданным формулам Бэкуса-Наура и возвращает строку, в которой записан ответ (*ACCEPT, REJECT*).

*Входные данные:*

final\_list – список списков, полученный в результате работы лексического блока

*Выходные данные:*

answer – строка “ACCEPT”, если удовлетворяет; строка “REJECT”, если не удовлетворяет.

# Кодирование

Для реализации распознавателя символьной цепочки была выбрана среда разработки PyCharm и язык программирования Python. Ниже приведено описание структуры текста программы и алгоритмы работы модулей.

## Структура текста программы

Ниже изображена структура текста программы. Модули первого уровня иерархии выделены в отдельные файлы. Блок идентификации ключевых слов (модуль второго уровня иерархии) реализован как функция лексического блока. Основной модуль находится в файле Main.py к которому подключаются остальные модули.

**Структура текста программы:**

Main.py

unit1.py

Блок

транслитерации

unit2.py

Блок

идентификации

ключевых слов

unit3.py

Синтаксический блок

Лексический блок

Модуль unit1.py – экспортирует подпрограмму «Блок транслитерации»; преобразует исходную символьную цепочку в цепочку лексем вида ("символ цепочки", "класс символа цепочки "). Модуль unit2.py –экспортирует подпрограмму «Лексический блок»; преобразует цепочку лексем, полученную от модуля unit1.py, в цепочку лексем вида ("символ входного языка", "класс символа входного языка"). Модуль unit3.py –экспортирует подпрограмму «Синтаксический блок»; устанавливает, соответствует ли цепочка лексем, полученная от модуля unit2.py, заданным формулам Бэкуса-Наура.

## Алгоритмы реализации модулей

Данный пункт следует разбить на следующие подпункты:

В данном пункте приведены алгоритмы реализации основных модулей программы: блок транслитерации, лексический блок, блок идентификации ключевых слов, синтаксический блок.

### Блок транслитерации

В данном пункте необходимо поместить *таблицу транслитерации* символьной цепочки.

Ниже приведены входные символы блока транслитерации и соответствующие им классы лексем. В таблице приведена транслитерация символьной цепочки.

**Транслитерация символьной цепочки**

|  |  |
| --- | --- |
| **Символы** | **Класс лексемы** |
| A..F | *шестрчбуква* |
| G..Z | *буква* |
| 0..9 | *цифра* |
| = | *равно* |
| **;** | *Тчкзпт* |
| +, - | *Знак* |
| $ | *Знкдоллар* |
| Пробел | *Пробел* |
| другие символы | *Ошибка* |

### Лексический блок

В данном пункте необходимо описать построение *обрабатывающего автомата лексического блока*: построение и редукция конечного распознавателя лексического блока, затем примитивные процедуры и преобразование распознавателя в обрабатывающий автомат. При использовании стандартных методов поиска недостижимых и эквивалентных состояний (например, методом разбиения) следует поместить соответствующую библиографическую ссылку.

Ниже приведён алгоритм разработки конечного автомата лексического блока, а также его редукция и преобразование к обрабатывающему автомату. В таблице ниже приведены состояния конечного распознавателя и их семантика. Начальное состояние - НАЧ. Допустимые состояния - ТЧКЗПТ и ПРОБЕЛ6.

**Состояния конечного распознавателя лексического блока**

|  |  |  |
| --- | --- | --- |
| **№**  **п/п** | **Состояние** | **Семантика** |
| 1. | НАЧ | Момент до начала обработки цепочки либо чтение пробелов в начале строки. |
| 2. | КЛСЛОВО | Чтение ключевого слова const. |
| 3. | ПРОБЕЛ1 | Чтение пробелов, находящихся между ключевым словом const и именем константы. |
| 4. | ИМЯ | Чтение имени константы. |
| 5. | ПРОБЕЛ2 | Чтение пробелов, находящихся между именем константы и знаком =. |
| 6. | РАВНО | Прочитан знак =. |
| 7. | ПРОБЕЛ3 | Чтение пробелов, находящихся между знаком = и значением константы. |
| 8. | ЗНАК | Прочитан знак целой константы. |
| 9. | ПРОБЕЛ41 | Чтение пробелов, находящихся между знаком целой константы и значением целой константы. |
| 10. | ЦЕЛОЕ | Чтение значения целой константы. |
| 11. | ЗНДОЛЛАР | Прочитан знак $. |
| 12. | ПРОБЕЛ42 | Чтение пробелов, находящихся между знаком $ и значением шестнадцатеричной константы. |
| 13. | ШЕСТРЧ | Чтение значения шестнадцатеричной константы. |
| 14. | ПРОБЕЛ5 | Чтение пробелов, находящихся между значение константы и знаком ; |
| 15. | ТЧКЗПТ | Прочитан знак ; |
| 16. | ПРОБЕЛ6 | Чтение пробелов, находящихся после знака ; |
| 17. | E | Ошибка. |

В таблице ниже приведена спецификация конечного распознавателя лексического блока.

**Конечный распознаватель лексического блока**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***шестрч-буква*** | ***буква*** | ***цифра*** | ***Пробел*** | ***равно*** | ***знак*** | ***знк-доллар*** | ***тчкзпт*** |  |
| **НАЧ** | КЛСЛОВО | КЛСЛОВО | Е | НАЧ | Е | Е | Е | Е | 0 |
| **КЛСЛОВО** | КЛСЛОВО | КЛСЛОВО | КЛСЛОВО | ПРОБЕЛ1 | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ1** | ИМЯ | ИМЯ | Е | ПРОБЕЛ1 | Е | Е | Е | Е | 0 |
| **ИМЯ** | ИМЯ | ИМЯ | ИМЯ | ПРОБЕЛ2 | РАВНО | Е | Е | Е | 0 |
| **ПРОБЕЛ2** | Е | Е | Е | ПРОБЕЛ2 | РАВНО | Е | Е | Е | 0 |
| **РАВНО** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ3 | Е | ЗНАК | ЗНДОЛЛАР | Е | 0 |
| **ПРОБЕЛ3** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ3 | Е | ЗНАК | ЗНДОЛЛАР | Е | 0 |
| **ЗНАК** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ41 | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ41** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ41 | Е | Е | Е | Е | 0 |
| **ЦЕЛОЕ** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ЗНДОЛЛАР** | ШЕСТРЧ | Е | ШЕСТРЧ | ПРОБЕЛ42 | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ42** | ШЕСТРЧ | Е | ШЕСТРЧ | ПРОБЕЛ42 | Е | Е | Е | Е | 0 |
| **ШЕСТРЧ** | ШЕСТРЧ | Е | ШЕСТРЧ | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ПРОБЕЛ5** | Е | Е | Е | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ТЧКЗПТ** | Е | Е | Е | ПРОБЕЛ6 | Е | Е | Е | Е | 1 |
| **ПРОБЕЛ6** | Е | Е | Е | ПРОБЕЛ6 | Е | Е | Е | Е | 1 |
| **E** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |

У конечного распознавателя лексического блока нет недостижимых состояний. Состояния РАВНО и ПРОБЕЛ3, ЗНАК и ПРОБЕЛ41, ЗНДОЛЛАР и ПРОБЕЛ42, ТЧКЗПТ и ПРОБЕЛ6 попарно эквивалентны. Дальнейший процесс редукции конечного распознавателя лексического блока методом разбиения [2] приведён в таблице ниже.

**Редукция конечного распознавателя лексического блока**

|  |  |  |
| --- | --- | --- |
| **Шаг** | **Результат (блоки состояний)** | **Действия** |
| 0. | P0={НАЧ,КЛСЛОВО,ПРОБЕЛ1,ИМЯ,ПРОБЕЛ2,РАВНО, ЗНАК,ЦЕЛОЕ,ЗНДОЛЛАР,ШЕСТРЧ,ПРОБЕЛ5,ТЧКЗПТ, Е} | Разбиваем P0 на два блока: допустимые и отвергающие состояния. |
| 1. | P11={НАЧ,КЛСЛОВО,ПРОБЕЛ1,ИМЯ,ПРОБЕЛ2,РАВНО, ЗНАК,ЦЕЛОЕ,ЗНДОЛЛАР,ШЕСТРЧ,ПРОБЕЛ5,Е}  P12={ТЧКЗПТ} | Разбиваем P11 по входу шестрчбуква. |
| 2. | P21={НАЧ,КЛСЛОВО}  P22={ПРОБЕЛ1,ИМЯ}  P23={ПРОБЕЛ2,РАВНО,ЗНАК,ЦЕЛОЕ,ПРОБЕЛ5,Е}  P24={ЗНДОЛЛАР,ШЕСТРЧ}  P25={ТЧКЗПТ} | 1.Разбиваем P21 по входу цифра.  2.Разбиваем P22 по входу цифра.  3.Разбиваем P23 по входу пробел.  4.Разбиваем P24 по входу тчкзпт |
| 3. | P31={НАЧ}  P32={КЛСЛОВО}  P33={ПРОБЕЛ1}  P34={ИМЯ}  P35={ПРОБЕЛ2}  P36={РАВНО}  P37={ЗНАК}  P38={ЦЕЛОЕ,ПРОБЕЛ5}  P39={ЗНДОЛЛАР}  P310={ШЕСТРЧ}  P311={ТЧКЗПТ}  P312={Е} | Разбиваем P38 по входу цифра |
| 4. | P41={НАЧ}  P42={КЛСЛОВО}  P43={ПРОБЕЛ1}  P44={ИМЯ}  P45={ПРОБЕЛ2}  P46={РАВНО}  P47={ЗНАК}  P48={ЦЕЛОЕ}  P49={ЗНДОЛЛАР}  P410={ШЕСТРЧ}  P411={ПРОБЕЛ5}  P412={ТЧКЗПТ}  P413={Е} |  |

В таблице ниже приведён результат редукции конечного распознавателя лексического блока.

**Минимальный конечный распознаватель лексического блока**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***шестрч-буква*** | ***буква*** | ***цифра*** | ***пробел*** | ***равно*** | ***знак*** | ***знк-доллар*** | ***тчкзпт*** |  |
| **НАЧ** | КЛСЛОВО | КЛСЛОВО | Е | НАЧ | Е | Е | Е | Е | 0 |
| **КЛСЛОВО** | КЛСЛОВО | КЛСЛОВО | КЛСЛОВО | ПРОБЕЛ1 | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ1** | ИМЯ | ИМЯ | Е | ПРОБЕЛ1 | Е | Е | Е | Е | 0 |
| **ИМЯ** | ИМЯ | ИМЯ | ИМЯ | ПРОБЕЛ2 | РАВНО | Е | Е | Е | 0 |
| **ПРОБЕЛ2** | Е | Е | Е | ПРОБЕЛ2 | РАВНО | Е | Е | Е | 0 |
| **РАВНО** | Е | Е | ЦЕЛОЕ | РАВНО | Е | ЗНАК | ЗНДОЛЛАР | Е | 0 |
| **ЗНАК** | Е | Е | ЦЕЛОЕ | ЗНАК | Е | Е | Е | Е | 0 |
| **ЦЕЛОЕ** | Е | Е | ЦЕЛОЕ | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ЗНДОЛЛАР** | ШЕСТРЧ | Е | ШЕСТРЧ | ЗНДОЛЛАР | Е | Е | Е | Е | 0 |
| **ШЕСТРЧ** | ШЕСТРЧ | Е | ШЕСТРЧ | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ПРОБЕЛ5** | Е | Е | Е | ПРОБЕЛ5 | Е | Е | Е | ТЧКЗПТ | 0 |
| **ТЧКЗПТ** | Е | Е | Е | ТЧКЗПТ | Е | Е | Е | Е | 1 |
| **E** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |

В таблице ниже указаны примитивные процедуры, необходимые для преобразования минимального конечного распознавателя в обрабатывающий автомат лексического блока.

**Примитивные процедуры обрабатывающего автомата лексического блока**

|  |  |  |
| --- | --- | --- |
| **№**  **п/п** | **Процедура** | **Семантика** |
| 1. | *ДА* | Остановить обработку и допустить цепочку. |
| 2. | *НЕТ* | Остановить обработку и отвергнуть цепочку. |
| 3. | *Обработать* | Добавить входной символ к значению текущей лексемы |
| 4. | *Лексема(класс)* | Увеличить счетчик лексем на 1, установить заданный класс текущей лексемы. |

В таблице ниже приведён обрабатывающий автомат лексического блока

**Обрабатывающий автомат лексического блока.**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ***шестрч-буква*** | ***буква*** | ***цифра*** | ***пробел*** | ***равно*** | ***знак*** | ***знк-доллар*** | ***тчкзпт*** | **-|** |
| **НАЧ** | **(1)**  КЛСЛОВО | **(1)**  КЛСЛОВО |  | НАЧ |  |  |  |  |  |
| **КЛСЛОВО** | **(2)**  КЛСЛОВО | **(2)**  КЛСЛОВО |  | ПРОБЕЛ1 |  |  |  |  |  |
| **ПРОБЕЛ1** | **(1)**  ИМЯ | **(1)**  ИМЯ |  | ПРОБЕЛ1 |  |  |  |  |  |
| **ИМЯ** | **(2)**  ИМЯ | **(2)**  ИМЯ | **(2)**  ИМЯ | ПРОБЕЛ2 | **(3)**  РАВНО |  |  |  |  |
| **ПРОБЕЛ2** |  |  |  | ПРОБЕЛ2 | **(3)**  РАВНО |  |  |  |  |
| **РАВНО** |  |  | **(4)**  ЦЕЛОЕ | РАВНО |  | **(4)**  ЗНАК | **(4)**  ЗНДОЛЛАР |  |  |
| **ЗНАК** |  |  | **(2)**  ЦЕЛОЕ | ЗНАК |  |  |  |  |  |
| **ЦЕЛОЕ** |  |  | **(2)**  ЦЕЛОЕ | ПРОБЕЛ5 |  |  |  | **(5)**  ТЧКЗПТ |  |
| **ЗНДОЛЛАР** | **(2)**  ШЕСТРЧ |  | **(2)**  ШЕСТРЧ | ЗНДОЛЛАР |  |  |  |  |  |
| **ШЕСТРЧ** | **(2)**  ШЕСТРЧ |  | **(2)**  ШЕСТРЧ | ПРОБЕЛ5 |  |  |  | **(5)**  ТЧКЗПТ |  |
| **ПРОБЕЛ5** |  |  |  | ПРОБЕЛ5 |  |  |  | **(5)**  ТЧКЗПТ |  |
| **ТЧКЗПТ** |  |  |  | ТЧКЗПТ |  |  |  |  | *ДА* |

Пустым клеткам соответствует вызов примитивной процедуры *НЕТ*. Для удобства чтения действия обрабатывающего автомата, выполняемые перед переходом в новое состояние, обозначены цифрами. Семантика действий дана в таблице.

**Процедуры переходов обрабатывающего автомата лексического блока**

|  |  |
| --- | --- |
| **Действие** | **Семантика** |
| **(1)** | Лексема(ИДЕНТ);  *Обработать;* |
| **(2)** | *Обработать;* |
| **(3)** | Лексема(РАВНО);  *Обработать;* |
| **(4)** | Лексема(ЗНАЧ);  *Обработать;* |
| **(5)** | Лексема(ТЧКЗПТ);  *Обработать;* |

### Синтаксический блок

Данный пункт должен иметь содержание, аналогичное пункту 3.2.1 – применительно к *конечному автомату синтаксического блока*.

В таблице представлен конечный распознаватель синтаксического блока. Пустым клеткам соответствует вызов примитивной процедуры *НЕТ*, которая отвергает цепочку.

**Конечный распознаватель синтаксического блока**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | ***КЛСЛОВО\_***  ***CONST*** | ***ИДЕНТ*** | ***РАВНО*** | ***ЗНАЧ*** | ***ТЧКЗПТ*** |  |
| **НАЧ** | CONST |  |  |  |  | 0 |
| **CONST** |  | ИМЯ |  |  |  | 0 |
| **ИМЯ** |  |  | РАВНО |  |  | 0 |
| **РАВНО** |  |  |  | ЗНАЧЕНИЕ |  | 0 |
| **ЗНАЧЕНИЕ** |  |  |  |  | ТЧКЗПТ | 0 |
| **ТЧКЗПТ** |  |  |  |  |  | 1 |

Данный автомат не подлежит редукции, а, следовательно, является минимальным.

### Блок идентификации ключевых слов

В данном пункте необходимо поместить *описание используемого метода идентификации ключевых слов* и поместить соответствующую библиографическую ссылку.

Блок идентификации ключевых слов будет содержать в себе список всех ключевых слов языка Pascal (кроме слова *const*). Если полученный идентификатор является ключевым словом *const*, то подпрограмма вернёт строку *клслово\_const*, иначе методом линейного поиска [3] подпрограмма сверит идентификатор со всеми словами из списка и при обнаружении совпадения вернёт строку *ошибка*. В таблице указаны все ключевые слова, при которых блок идентификации ключевых слов будет возвращать строку *ошибка.*

**Ключевые слова языка Pascal**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Absolute | downto | If | Nil | set | var |
| And | destructor | implementation | Not | shl | while |
| Array | else | In | Or | shr | with |
| Asm | end | inline | Of | string | xor |
| Begin | exports | interrupt | object | then |  |
| Case | external | interface | packed | to |  |
| Constructor | file | inherited | procedure | type |  |
| Div | for | label | program | unit |  |
| Goto | forward | library | record | until |  |
| Do | function | mod | repeat | uses |  |

## Размер текста программы (в строках)

Данный пункт отчета должен присутствовать, если выполнены работы по кодированию программной системы. В данном пункте следует поместить округленный до сотен *общий размер созданных исходных текстов в строках* (включая пустые строки, комментарии, спецификации программных файлов и др.).

Общий размер программы составляет 137 строк: размер головного модуля – 27 строк, размер модуля Unit1 – 30 строк, размер модуля Unit2 – 68 строк, размер модуля Unit3 – 12 строк. Реализация программы указан в приложении.

# Тестирование

В данном пункте приведены протоколы автономного и комплексного тестирования программы.

## Автономное тестирование

Данный пункт следует разбить на подпункты в соответствии с модульной структурой программы и в каждом пункте поместить *протокол тестирования* *соответствующего* *модуля*.

В таблицах ниже представлены протоколы тестирования модулей распознавателя лексической цепочки: блок транслитерации, лексический блок, блок идентификации ключевых слов, синтаксический блок.

**Протокол тестирования блока транслитерации**

|  |  |  |
| --- | --- | --- |
| **Модуль** | **Тесты** | |
| **Входные данные** | **Выходные данные** |
| Блок транслитерации | const N3=$4F; | (c, шестрчбуква),  (о, буква),  (n, буква),  (s, буква),  (t, буква),  ( , пробел),  (n, буква),  (3, цифра),  (=, равно),  ($, зндоллар),  (4, цифра),  (f, шестрчбуква),  (;, тчкзпт) |

**Протокол тестирования лексического блока**

|  |  |  |
| --- | --- | --- |
| **Модуль** | **Тесты** | |
| **Входные данные** | **Выходные данные** |
| Лексический блок | (c, шестрчбуква),  (о, буква),  (n, буква),  (s, буква),  (t, буква),  ( , пробел),  (n, буква),  (3, цифра),  (=, равно),  ($, зндоллар),  (4, цифра),  (f, шестрчбуква),  (;, тчкзпт) | (const, ИДЕНТ),  (n3, ИДЕНТ),  (=, РАВНО),  ($4f, ЗНАЧ),  (;, ТЧКЗПТ) |

**Протокол тестирования блока идентификации ключевых слов**

|  |  |  |
| --- | --- | --- |
| **Модуль** | **Тесты** | |
| **Входные данные** | **Выходные данные** |
| Блок идентификации ключевых слов | const | КЛСЛОВО\_CONST |
| n3 | ИДЕНТ |
| fo | ИДЕНТ |
| for | *Ошибка* |
| repeat | *Ошибка* |

**Протокол тестирования синтаксического блока**

|  |  |  |
| --- | --- | --- |
| **Модуль** | **Тесты** | |
| **Входные данные** | **Выходные данные** |
| Синтаксический блок | (const, КЛСЛОВО\_CONST),  (n3, ИДЕНТ),  (=, РАВНО),  ($4f, ЗНАЧ),  (;, ТЧКЗПТ) | ACCEPT |
| (conct, ИДЕНТ),  (n3, ИДЕНТ),  (=, РАВНО),  ($4f, ЗНАЧ),  (;, ТЧКЗПТ) | REJECT |

Тесты изображены *символически*, без указания точного формата входных и выходных данных.

## Комплексное тестирование

В данном пункте следует поместить *протокол тестирования головного модуля* программы.

В таблице представлен протокол тестирования головного модуля. Всего было проведено 9 тестов, в ходе которых ошибок обнаружено не было.

**Протокол тестирования головного модуля**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные**  **Данные** | **Выходные**  **данные** | **Действительный**  **Результат** | **Тест**  **пройден?** |
| 1. | const one=1; | ACCEPT | ACCEPT | Да |
| 2. | const Code=-23; | ACCEPT | ACCEPT | Да |
| 3. | const HEX=$10A; | ACCEPT | ACCEPT | Да |
| 4. | const N12=+5; | ACCEPT | ACCEPT | Да |
| 5. | const Code=-23 | REJECT | REJECT | Да |
| 6. | const Min=+-4; | REJECT | REJECT | Да |
| 7. | conct D5=$5F; | REJECT | REJECT | Да |
| 8. | const Max=4E; | REJECT | REJECT | Да |
| 9. | const n=$5H; | REJECT | REJECT | Да |

# Заключение

В данном разделе следует поместить *краткую сводку всех полученных результатов*: перечислить пройденные этапы технологического цикла разработки и соответствующие выполненные работы и указать не пройденные этапы (не выполненные работы) и работы, выполненные лишь частично.

В ходе работы был разработан конечный распознаватель заданной символьной цепочки. В результате анализа требований составлена спецификация, спроектирована модульная структура программы, описаны алгоритмы реализации всех модулей, выполнена реализация, проведено автономное тестирование основных модулей и комплексное тестирование программы. В результате тестирования ошибок выявлено не было.

# Литература

В данном разделе следует поместить все *использованные библиографические источники*, ссылки на которые имеются в остальных разделах отчета. Список литературы должен содержать не менее двух наименований. Каждый элемент списка литературы должен быть оформлен в соответствии с оформлением библиографических ссылок в списке литературы методических указаний к практике.

1. *Йенсен К., Вирт Н.* Паскаль. Руководство пользователя и описание языка. - М.: Компьютер, 1995.
2. *Льюис Ф., Розенкранц Д., Стирнз Р.* Теоретические основы проектирования компиляторов. - М.: Мир, 1979.
3. *Дональд Кнут.* Искусство программирования, том 3. Сортировка и поиск - 2-е изд. - М.: «Вильямс», 2007.