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 4](#_Toc13881783)

[2. Проектирование 6](#_Toc13881784)

[2.1. Модульная структура 6](#_Toc13881785)

[2.2. Интерфейсы модулей 6](#_Toc13881786)

[3. Кодирование 9](#_Toc13881787)

[3.1. Структура текста программы 9](#_Toc13881788)

[3.2. Алгоритмы реализации модулей 9](#_Toc13881789)

[3.2.1. Блок транслитерации 10](#_Toc13881790)

[3.2.2. Лексический блок 10](#_Toc13881791)

[3.2.3. Синтаксический блок 17](#_Toc13881792)

[3.2.4. Блок идентификации ключевых слов 19](#_Toc13881793)

[3.3. Размер текста программы (в строках) 19](#_Toc13881794)

[4. Тестирование 20](#_Toc13881795)

[4.1. Автономное тестирование 20](#_Toc13881796)

[4.2. Комплексное тестирование 26](#_Toc13881797)

[Заключение 27](#_Toc13881798)

[Литература 28](#_Toc13881799)

# Спецификация

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

**<цепочка>::=<условный оператор>**

<условие>::=<идентификатор>

<идентификатор>::=<буква> | <подчеркивание> |<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>

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

**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**

Примеры входных данных и соответствующих им выходных данных представлены в таблице 1.1.

Таблица 1.1 – Примеры входных выходных данных

|  |  |
| --- | --- |
| **INPUT.TXT** | **OUTPUT.TXT** |
| if Cond then A1:=Func(id); | ACCEPT |
| if Cond then myvar:=foo(id1) else proc(2); | ACCEPT |
| if Cond then samp:=func3(id1) else proc(-32); | ACCEPT |
| if \_cond2 then v := prog(inp) ; | ACCEPT |
| if COND then abc := pr(inp) else pr(+8); | ACCEPT |
| if AbCd\_98 then plit := func3(id1) else proc(-322585) ; | ACCEPT |
| if Cond then var :=foo(id1) else proc(5); | REJECT |
| if 33ad then var:=f(id2); | REJECT |
| iff cond then input := foo(id1) ; | REJECT |
| /if cond then input := foo(id1) ; | REJECT |
| if cond input := foo(id1) ; | REJECT |
| if cond then input := foo(5) ; | REJECT |
| if cond then input := foo(id1) else | REJECT |
| if cond then input := foo(id1) else progr1(-32.5); | REJECT |

# Проектирование

На этапе проектирования выполняется проектирование модульной структуры программы и разработка набора тестов и соответствующих тестовых программ для проведения тестирования. Важной особенностью этапа проектирования является то, что все работы на данном этапе выполняются без использования системы программирования.

## Модульная структура

В результате проведённого анализа требований была спроектирована мо-дульная структура программы. Программа включает 4 основных блока, которые находятся на первом уровне иерархии (Рисунок 2.1). Все блоки вызываются непосредственно из головного модуля последовательно. В головном модуле программы реализованы также функции для считывания входных данных и записи результата в выходной файл. Описание основных блоков приведено ниже, модульная структура программы приведена ниже (Рисунок 2.1).

Рисунок 2.1. 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 (func\_Translit):*** разбивает полученную из файла строку на символы и присваивает каждому символу свой класс (*буква, цифра, пробел и т.д.*).

*Входные данные*:

buffer - строка с исходной символьной цепочкой.

*Выходные данные*:

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

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

*Входные данные:*

translit\_result– список кортежей, полученный в результате работы блока транслитерации.

*Выходные данные:*

leks\_result - список кортежей, где в каждом кортеже нулевой элемент является строкой с «символом(строкой) входного языка», а первый элемент является строкой с «классом символа входного языка».

***Блок идентификации ключевых слов (func\_KeyWordIdent):*** проверяет, является ли полученный идентификатор ключевым словом языка Pascal и возвращает его класс (*идентификатор, клслово\_const, …*).

*Входные данные:*

leks\_result – список кортежей, полученный от лексического анализатора.

*Выходные данные:*

id\_result – список кортежей, где в каждом кортеже нулевой элемент является строкой с «символом(строкой) входного языка», а первый элемент является строкой с «классом символа входного языка» (зарезервированным словом языка).

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

*Входные данные:*

id\_result – список кортежей, полученный из блока идентификации зарезервированных слов.

*Выходные данные:*

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

В перечисленных блоках будет использоваться тип «Token», определенный для удобства передачи информации между блоками и содержащий строку с входными символами и строку с соответствующим ей классом. Кроме того, при реализации будут использоваться типы из библиотеки STL языка C++ [3]. Например, vector для удобства работы с массивами данных, string для удобства работы со строками и пр. Для хранения состояний конечного автомата будут использоваться перечисления (enum).

# Кодирование

Для реализации распознавателя символьной цепочки была выбрана среда разработки Microsoft Visual Studio Professional 2017 и язык программирования C++11. Ниже приведено описание структуры текста программы и алгоритмы работы модулей.

## Структура текста программы

Ниже изображена структура текста программы (Рисунок 3.1). Все модули программы (блок транслитерации, лексический блок, блок определения зарезервированных слов (ЗС) языка Pascal и синтаксический блок) выделены в отдельные файлы. Основной модуль находится в файле main.cpp, к которому подключаются остальные модули.

main.cpp

1\_func\_Translit.cpp

Блок транслитерации

4\_func\_Syntax.cpp

Синтаксический блок

2\_func\_Leks.cpp

3\_func\_KeyWordIdent.cpp

Лексический блок

Блок определения ЗС

Рисунок 3.1 – Схема структуры текста программы

Модуль 1\_func\_Translit.cpp экспортирует подпрограмму «Блок транслитерации», преобразует исходную символьную цепочку в цепочку лексем вида ("символ цепочки", "класс символа цепочки "). Модуль 2\_func\_Leks.cpp экспортирует подпрограмму «Лексический блок», преобразует цепочку лексем, полученную от модуля 1\_func\_Translit.cpp, в цепочку лексем вида ("символ (подстрока) цепочки", "класс символа входного языка"). Модуль 3\_func\_KeyWordIdent.cpp экспортирует подпрограмму «Блок определения зарезервированных слов языка Pascal», проверяет цепочку лексем, полученную от модуля 2\_func\_Leks.cpp на наличие зарезервированных слов, возвращает цепочку вида ("символ (подстрока) цепочки", "класс символа входного языка"), где в "класс символа входного языка" так же войдут зарезервированные слова Pascal, если они имеются в исходной цепочке. Модуль 4\_func\_Syntax.cpp –экспортирует подпрограмму «Синтаксический блок»; устанавливает, соответствует ли цепочка лексем, полученная от модуля 3\_func\_KeyWordIdent.cpp, заданным формулам Бэкуса-Наура.

## Алгоритмы реализации модулей

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

### Блок транслитерации

Ниже (Таблица 3.1) приведены входные символы блока транслитерации и соответствующие им классы лексем. Блок транслитерации реализован в виде конечного автомата [2], который выполняет обработку и распознавание входной символьной цепочки.

Таблица 3.1. Транслитерация символьной цепочки

|  |  |
| --- | --- |
| **Символы** | **Класс лексемы** |
| A..Z, a..z | БУКВА |
| 0..9 | ЦИФРА |
| : | ДВОЕТОЧИЕ |
| = | РАВНО |
| **;** | ТЧКЗПТ |
| +, - | ЗНАК |
| ( | ОСКОБКА (открывающаяся скобка) |
| ) | ЗСКОБКА (закрывающаяся скобка) |
| \_ | ПОДЧЕРК (нижнее подчеркивание) |
| Пробел | ПРОБЕЛ |
| другие символы | ОШИБКА |

### Лексический блок

Построение конечного автомата лексического блока выполним следующим образом. В первую очередь, используя метод разметки символов [2], построим конечный распознаватель входной цепочки лексем: определим список состояний (название / номер и семантика состояния), выделим в данном списке начальное состояние и допускающие состояния, укажем функцию переходов.

В таблице 3.2 (с. 11) определены состояния конечного распознавателя лексического блока символьной цепочки, определенной в пункте 1. Для удобства оформления состояния будут записаны кратко.

Начальное состояние - НАЧ. Допустимые состояния - ТЧКЗПТ и ПРОБЕЛ9.

Таблица 3.2. Состояния конечного распознавателя лексического блока

|  |  |  |
| --- | --- | --- |
| **№**  **п/п** | **Состояние** | **Семантика** |
| 1. | НАЧ | Момент до начала обработки цепочки либо чтение пробелов в начале строки. |
| 2. | ЕСЛИ | Чтение ключевого слова if. |
| 3. | ПРОБЕЛ1 | Чтение пробелов, находящихся между ключевым словом if и идентификатором. |
| 4. | ИДЕНТ1 | Чтение идентификатора |
| 5. | ПРОБЕЛ2 | Чтение пробелов, находящихся между идентификатором и словом then. |
| 6. | ТО | Чтение ключевого слова then |
| 7. | ПРОБЕЛ3 | Чтение пробелов, находящихся между then и идентификатором. |
| 8. | ИДЕНТ2 | Чтение идентификатора. |
| 9. | ПРОБЕЛ4 | Чтение пробелов, находящихся между идентификатором и знаком присвоения. |
| 10. | ДВОЕТОЧИЕ | Чтение знака ‘:’ |
| 11. | РАВНО | Чтение знака ‘=’ |
| 12. | ПРОБЕЛ5 | Чтение пробелов, находящихся между знаком присвоения и вызовом функции. |
| 13. | ИМЯ1 | Чтение имени функции. |
| 14. | ОСКОБКА1 | Чтение открывающейся скобки. |
| 15. | ИДЕНТ3 | Чтение идентификатора. |
| 16. | ЗСКОБКА1 | Чтение закрывающейся скобки. |
| 17. | ПРОБЕЛ6 | Чтение пробелов после закрывающей скобки. |
| 18. | ИНАЧЕ | Чтение else |
| 19. | ПРОБЕЛ7 | Чтение пробелов после else |
| 20. | ИМЯ2 | Чтение имени функции. |
| 21. | ОСКОБКА2 | Чтение открывающей скобки. |
| 22. | ЗНАК | Чтение знака константы. |
| 23. | ЦЕЛОЕ | Чтение целого числа. |
| 24. | ЗСКОБКА2 | Чтение закрывающей скобки |
| 25. | ПРОБЕЛ8 | Чтение пробелов после закрывающей скобки в else. |
| 26. | ТЧКЗПТ | Чтение точки с запятой ‘;' |
| 27. | ПРОБЕЛ9 | Чтение пробелов после ';' |
| 28. | E | Ошибка. |

В таблице 3.3 приведена спецификация конечного распознавателя лексического блока. Для удобства построения таблицы введем следующие обозначения: ТЧКЗПТ будем обозначать как ‘;’, ДВОЕТОЧИЕ как ‘:’, РАВНО как ‘=’, ОСКОБКА как ‘(’, ЗСКОБКА как ‘)’, ПОДЧЕРКИВАНИЕ как ‘\_’, ЗНАК как ‘’.

Таблица 3.3. Конечный распознаватель лексического блока

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ПРОБЕЛ** | **;** | **:** | **=** | **(** | **)** | **БУКВА** | **ЦИФРА** |  | **\_** |  |
| **НАЧ** | НАЧ | Е | Е | Е | Е | Е | ЕСЛИ | Е | Е | Е | 0 |
| **ЕСЛИ** | ПРОБЕЛ1 | Е | Е | Е | Е | Е | ЕСЛИ | Е | Е | Е | 0 |
| **ПРОБЕЛ1** | ПРОБЕЛ1 | Е | Е | Е | Е | Е | ИДЕНТ1 | Е | Е | ИДЕНТ1 | 0 |
| **ИДЕНТ1** | ПРОБЕЛ2 | Е | Е | Е | Е | Е | ИДЕНТ1 | ИДЕНТ1 | Е | ИДЕНТ1 | 0 |
| **ПРОБЕЛ2** | ПРОБЕЛ2 | Е | Е | Е | Е | Е | ТО | Е | Е | E | 0 |
| **ТО** | ПРОБЕЛ3 | Е | Е | Е | Е | Е | ТО | Е | Е | E | 0 |
| **ПРОБЕЛ3** | ПРОБЕЛ3 | Е | Е | Е | Е | Е | ИДЕНТ2 | Е | Е | ИДЕНТ2 | 0 |
| **ИДЕНТ2** | ПРОБЕЛ4 | Е | **:** | Е | Е | Е | ИДЕНТ2 | ИДЕНТ2 | Е | ИДЕНТ2 | 0 |
| **ПРОБЕЛ4** | ПРОБЕЛ4 | Е | **:** | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ДВОЕТОЧИЕ** | Е | Е | Е | = | Е | Е | Е | Е | Е | Е | 0 |
| **РАВНО** | ПРОБЕЛ5 | Е | Е | Е | Е | Е | ИМЯ1 | Е | Е | ИМЯ1 | 0 |
| **ПРОБЕЛ5** | ПРОБЕЛ5 | Е | Е | Е | Е | Е | ИМЯ1 | Е | Е | ИМЯ1 | 0 |
| **ИМЯ1** | Е | Е | Е | Е | ОСКОБКА1 | Е | ИМЯ1 | ИМЯ1 | Е | ИМЯ1 | 0 |
| **ОСКОБКА1** | Е | Е | Е | Е | Е | Е | ИДЕНТ3 | Е | Е | ИДЕНТ3 | 0 |
| **ИДЕНТ3** | Е | Е | Е | Е | Е | ЗСКОБКА1 | ИДЕНТ3 | ИДЕНТ3 | Е | ИДЕНТ3 | 0 |
| **ЗСКОБКА1** | ПРОБЕЛ6 | **;** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ6** | ПРОБЕЛ6 | **;** | Е | Е | Е | Е | ИНАЧЕ | Е | Е | Е | 0 |
| **ИНАЧЕ** | ПРОБЕЛ7 | Е | Е | Е | Е | Е | ИНАЧЕ | Е | Е | Е | 0 |
| **ПРОБЕЛ7** | ПРОБЕЛ7 | Е | Е | Е | Е | Е | ИМЯ2 | Е | Е | ИМЯ2 | 0 |
| **ИМЯ2** | Е | Е | Е | Е | ОСКОБКА2 | Е | ИМЯ2 | ИМЯ2 | Е | ИМЯ2 | 0 |
| **ОСКОБКА2** | Е | Е | Е | Е | Е | Е | Е | ЦЕЛОЕ |  | Е | 0 |
| **ЗНАК** | Е | Е | Е | Е | Е | Е | Е | ЦЕЛОЕ | Е | Е | 0 |
| **ЦЕЛОЕ** | Е | Е | Е | Е | Е | ЗСКОБКА2 | Е | ЦЕЛОЕ | Е | Е | 0 |
| **ЗСКОБКА2** | ПРОБЕЛ8 | **;** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ8** | ПРОБЕЛ8 | **;** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ТЧКЗПТ** | ПРОБЕЛ9 | Е | Е | Е | Е | Е | Е | Е | Е | Е | 1 |
| **ПРОБЕЛ9** | ПРОБЕЛ9 | Е | Е | Е | Е | Е | Е | Е | Е | Е | 1 |
| **E** | Е | Е | Е | Е | Е | Е | Е | Е | Е | Е | 0 |

Далее выполним редукцию построенного автомата [2]. Недостижимых состояний нет. Состояния РАВНО и ПРОБЕЛ5, ЗСКОБКА2 и ПРОБЕЛ8, ТЧКЗПТ и ПРОБЕЛ9 попарно идентичны, а, следовательно, эквивалентны. Дальнейший процесс редукции конечного распознавателя лексического блока приведен в таблице 3.4.

Таблица 3.4. Редукция конечного распознавателя лексического блока

|  |  |  |
| --- | --- | --- |
| Шаг | Результат (блоки состояний) | Действия |
| 0. | P0={НАЧ, ЕСЛИ, ПРОБЕЛ1, ИДЕНТ1, ПРОБЕЛ2, ТО, ПРОБЕЛ3, ИДЕНТ2, ПРОБЕЛ4, ДВОЕТОЧИЕ, РАВНО, ИМЯ1, ОСКОБКА1, ИДЕНТ3, ЗСКОБКА1, ПРОБЕЛ6, ИНАЧЕ, ПРОБЕЛ7, ИМЯ2, ОСКОБКА2, ЗНАК, ЦЕЛОЕ, ЗСКОБКА2, ТЧКЗПТ, E} | Разбиваем P0 на 2 блока: допустимые и отвергающие состояния. |
| 1. | P11={НАЧ, ЕСЛИ, ПРОБЕЛ1, ИДЕНТ1, ПРОБЕЛ2, ТО, ПРОБЕЛ3, ИДЕНТ2, ПРОБЕЛ4, ДВОЕТОЧИЕ, РАВНО, ИМЯ1, ОСКОБКА1, ИДЕНТ3, ЗСКОБКА1, ПРОБЕЛ6, ИНАЧЕ, ПРОБЕЛ7, ИМЯ2, ОСКОБКА2, ЗНАК, ЦЕЛОЕ, ЗСКОБКА2, E},  P12={ТЧКЗПТ} | Разбиваем P11 по входу БУКВА. |
| 2. | P21={НАЧ, ЕСЛИ},  P22={ПРОБЕЛ1, ИДЕНТ1},  P23={ПРОБЕЛ2, ТО},  P24={ПРОБЕЛ3, ИДЕНТ2},  P25={ПРОБЕЛ4, ДВОЕТОЧИЕ, ЗСКОБКА1, ОСКОБКА2, ЗНАК, ЦЕЛОЕ, ЗСКОБКА2, E},  P26={РАВНО, ИМЯ1},  P27={ОСКОБКА1, ИДЕНТ3},  P28={ПРОБЕЛ6, ИНАЧЕ},  P29={ПРОБЕЛ7, ИМЯ2},  P210={ТЧКЗПТ} | 1. Разбиваем P21 по входу ПРОБЕЛ 2. Разбиваем P22 по входу ЦИФРА 3. Разбиваем P23 по входу ПРОБЕЛ 4. Разбиваем P24 по входу ЦИФРА 5. Разбиваем P25 по входу ТЧКЗПТ 6. Разбиваем P26 по входу ЦИФРА 7. Разбиваем P27 по входу ЦИФРА 8. Разбиваем P28 по входу ТЧКЗПТ 9. Разбиваем P29 по входу ЦИФРА |
| 3. | P31={НАЧ}, P32={ЕСЛИ},  P33={ПРОБЕЛ1}, P34={ИДЕНТ1},  P35={ПРОБЕЛ2}, P36={ТО}, P37={ПРОБЕЛ3},  P38={ИДЕНТ2}, P39={РАВНО}, P310={ИМЯ1},  P311={ОСКОБКА1}, P312={ИДЕНТ3},  P313={ПРОБЕЛ6}, P314={ИНАЧЕ},  P315={ПРОБЕЛ7}, P316={ИМЯ2},  P317={ЗСКОБКА1, ЗСКОБКА2},  P318={ПРОБЕЛ4, ДВОЕТОЧИЕ, ОСКОБКА2, ЗНАК, ЦЕЛОЕ, E}, P319={ТЧКЗПТ} | 1. Разбиваем P317 по входу ПРОБЕЛ 2. Разбиваем P318 по входу ЦИФРА |
| 4. | P41={НАЧ}, P42={ЕСЛИ}, P43={ПРОБЕЛ1},  P44={ИДЕНТ1}, P45={ПРОБЕЛ2}, P46={ТО},  P47={ПРОБЕЛ3}, P48={ИДЕНТ2},  P49={РАВНО}, P410={ИМЯ1},  P411={ОСКОБКА1}, P412={ИДЕНТ3},  P413={ПРОБЕЛ6}, P414={ИНАЧЕ},  P415={ПРОБЕЛ7}, P416={ИМЯ2},  P417={ЗСКОБКА1}, P418={ЗСКОБКА2},  P419={ПРОБЕЛ4, ДВОЕТОЧИЕ, E},  P420={ОСКОБКА2, ЗНАК, ЦЕЛОЕ},  P421={ТЧКЗПТ} | 1. Разбиваем P419 по входу РАВНО 2. Разбиваем P420 по входу ЗСКОБКА |
| 5. | P51={НАЧ}, P52={ЕСЛИ}, P53={ПРОБЕЛ1},  P54={ИДЕНТ1}, P55={ПРОБЕЛ2}, P56={ТО},  P57={ПРОБЕЛ3}, P58={ИДЕНТ2},  P59={РАВНО}, P510={ИМЯ1},  P511={ОСКОБКА1}, P512={ИДЕНТ3},  P513={ПРОБЕЛ6}, P514={ИНАЧЕ},  P515={ПРОБЕЛ7}, P516={ИМЯ2},  P517={ЗСКОБКА1}, P518={ЗСКОБКА2},  P519={ТЧКЗПТ}, P520={ДВОЕТОЧИЕ},  P521={ПРОБЕЛ4, E}, P522={ЦЕЛОЕ},  P523={ОСКОБКА2, ЗНАК} | 1. Разбиваем P521 по входу ДВОЕТОЧИЕ 2. Разбиваем P523 по входу ЗНАК |
| 6. | P61={НАЧ}, P62={ЕСЛИ}, P63={ПРОБЕЛ1},  P64={ИДЕНТ1}, P65={ПРОБЕЛ2}, P66={ТО},  P67={ПРОБЕЛ3}, P68={ИДЕНТ2},  P69={РАВНО}, P610={ИМЯ1},  P611={ОСКОБКА1}, P612={ИДЕНТ3},  P613={ПРОБЕЛ6}, P614={ИНАЧЕ},  P615={ПРОБЕЛ7}, P616={ИМЯ2},  P617={ЗСКОБКА1}, P618={ЗСКОБКА2},  P619={ТЧКЗПТ}, P620={ДВОЕТОЧИЕ},  P621={ОСКОБКА2}, P622={ЗНАК},  P623={ЦЕЛОЕ}, P624={ПРОБЕЛ4}, P625={E} | Блоки содержат по одному состоянию и дальнейшему разбиению не подлежат. Дополнительно эквивалентных состояний обнаружено не было. |

В таблице 3.5 приведен результат редукции конечного распознавателя лексического блока. После редукции были удалены состояния «ПРОБЕЛ5», «ПРОБЕЛ8», «ПРОБЕЛ9», остальные состояния «ПРОБЕЛ» были перенумерованы.

Таблица 3.5. Минимальный конечный распознаватель лексического блока

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ПРОБЕЛ** | **;** | **:** | **=** | **(** | **)** | **БУКВА** | **ЦИФРА** |  | **\_** |  |
| **НАЧ** | НАЧ | Е | Е | Е | Е | Е | ЕСЛИ | Е | Е | Е | 0 |
| **ЕСЛИ** | ПРОБЕЛ1 | Е | Е | Е | Е | Е | ЕСЛИ | Е | Е | Е | 0 |
| **ПРОБЕЛ1** | ПРОБЕЛ1 | Е | Е | Е | Е | Е | ИДЕНТ1 | Е | Е | ИДЕНТ1 | 0 |
| **ИДЕНТ1** | ПРОБЕЛ2 | Е | Е | Е | Е | Е | ИДЕНТ1 | ИДЕНТ1 | Е | ИДЕНТ1 | 0 |
| **ПРОБЕЛ2** | ПРОБЕЛ2 | Е | Е | Е | Е | Е | ТО | Е | Е | E | 0 |
| **ТО** | ПРОБЕЛ3 | Е | Е | Е | Е | Е | ТО | Е | Е | E | 0 |
| **ПРОБЕЛ3** | ПРОБЕЛ3 | Е | Е | Е | Е | Е | ИДЕНТ2 | Е | Е | ИДЕНТ2 | 0 |
| **ИДЕНТ2** | ПРОБЕЛ4 | Е | **:** | Е | Е | Е | ИДЕНТ2 | ИДЕНТ2 | Е | ИДЕНТ2 | 0 |
| **ПРОБЕЛ4** | ПРОБЕЛ4 | Е | **:** | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ДВОЕТОЧИЕ** | Е | Е | Е | = | Е | Е | Е | Е | Е | Е | 0 |
| **РАВНО** | РАВНО | Е | Е | Е | Е | Е | ИМЯ1 | Е | Е | ИМЯ1 | 0 |
| **ИМЯ1** | Е | Е | Е | Е | ОСКОБКА1 | Е | ИМЯ1 | ИМЯ1 | Е | ИМЯ1 | 0 |
| **ОСКОБКА1** | Е | Е | Е | Е | Е | Е | ИДЕНТ3 | Е | Е | ИДЕНТ3 | 0 |
| **ИДЕНТ3** | Е | Е | Е | Е | Е | ЗСКОБКА1 | ИДЕНТ3 | ИДЕНТ3 | Е | ИДЕНТ3 | 0 |
| **ЗСКОБКА1** | ПРОБЕЛ5 | **;** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ПРОБЕЛ5** | ПРОБЕЛ5 | **;** | Е | Е | Е | Е | ИНАЧЕ | Е | Е | Е | 0 |
| **ИНАЧЕ** | ПРОБЕЛ6 | Е | Е | Е | Е | Е | ИНАЧЕ | Е | Е | Е | 0 |
| **ПРОБЕЛ6** | ПРОБЕЛ6 | Е | Е | Е | Е | Е | ИМЯ2 | Е | Е | ИМЯ2 | 0 |
| **ИМЯ2** | Е | Е | Е | Е | ОСКОБКА2 | Е | ИМЯ2 | ИМЯ2 | Е | ИМЯ2 | 0 |
| **ОСКОБКА2** | Е | Е | Е | Е | Е | Е | Е | ЦЕЛОЕ |  | Е | 0 |
| **ЗНАК** | Е | Е | Е | Е | Е | Е | Е | ЦЕЛОЕ | Е | Е | 0 |
| **ЦЕЛОЕ** | Е | Е | Е | Е | Е | ЗСКОБКА2 | Е | ЦЕЛОЕ | Е | Е | 0 |
| **ЗСКОБКА2** | ТЧКЗПТ | **;** | Е | Е | Е | Е | Е | Е | Е | Е | 0 |
| **ТЧКЗПТ** | ТЧКЗПТ | Е | Е | Е | Е | Е | Е | Е | Е | Е | 1 |
| **E** | Е | Е | Е | Е | Е | Е | Е | Е | Е | Е | 0 |

Полученный минимальный конечный распознаватель преобразуем в обрабатывающий автомат. В таблице 3.6 указаны примитивные процедуры, необходимые для преобразования минимального конечного распознавателя в обрабатывающий автомат лексического блока.

Таблица 3.6. 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. |

Итак, представим обрабатывающий автомат лексического блока нашей программы в таблице 3.7.

Таблица 3.7. Обрабатывающий автомат лексического блока

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **ПРОБЕЛ** | **;** | **:** | **=** | **(** | **)** | **БУКВА** | **ЦИФРА** |  | **\_** | **-|** |
| **НАЧ** | НАЧ |  |  |  |  |  | **(1)** ЕСЛИ |  |  |  |  |
| **ЕСЛИ** | ПРОБЕЛ1 |  |  |  |  |  | **(2)**  ЕСЛИ |  |  |  |  |
| **ПРОБЕЛ1** | ПРОБЕЛ1 |  |  |  |  |  | **(3)**  ИДЕНТ1 |  |  | **(3)**  ИДЕНТ1 |  |
| **ИДЕНТ1** | ПРОБЕЛ2 |  |  |  |  |  | **(2)**  ИДЕНТ1 | **(2)**  ИДЕНТ1 |  | **(2)**  ИДЕНТ1 |  |
| **ПРОБЕЛ2** | ПРОБЕЛ2 |  |  |  |  |  | **(4)**  ТО |  |  |  |  |
| **ТО** | ПРОБЕЛ3 |  |  |  |  |  | **(2)**  ТО |  |  |  |  |
| **ПРОБЕЛ3** | ПРОБЕЛ3 |  |  |  |  |  | **(5)**  ИДЕНТ2 |  |  | **(5)**  ИДЕНТ2 |  |
| **ИДЕНТ2** | ПРОБЕЛ4 |  | **(6)**  **:** |  |  |  | **(2)**  ИДЕНТ2 | **(2)**  ИДЕНТ2 |  | **(2)**  ИДЕНТ2 |  |
| **ПРОБЕЛ4** | ПРОБЕЛ4 |  | **(6)**  **:** |  |  |  |  |  |  |  |  |
| **ДВОЕТОЧИЕ** |  |  |  | **(7)**  = |  |  |  |  |  |  |  |
| **РАВНО** | РАВНО |  |  |  |  |  | **(8)**  ИМЯ1 |  |  | **(8)**  ИМЯ1 |  |
| **ИМЯ1** |  |  |  |  | **(9)**  ОСКОБКА1 |  | **(2)**  ИМЯ1 | **(2)**  ИМЯ1 |  | **(2)**  ИМЯ1 |  |
| **ОСКОБКА1** |  |  |  |  |  |  | **(10)**  ИДЕНТ3 |  |  | **(10)**  ИДЕНТ3 |  |
| **ИДЕНТ3** |  |  |  |  |  | **(11)**  ЗСКОБКА1 | **(2)**  ИДЕНТ3 | **(2)**  ИДЕНТ3 |  | **(2)**  ИДЕНТ3 |  |
| **ЗСКОБКА1** | ПРОБЕЛ5 | **(12)**  **;** |  |  |  |  |  |  |  |  |  |
| **ПРОБЕЛ5** | ПРОБЕЛ5 | **(12)**  **;** |  |  |  |  | **(13)**  ИНАЧЕ |  |  |  |  |
| **ИНАЧЕ** | ПРОБЕЛ6 |  |  |  |  |  | **(2)**  ИНАЧЕ |  |  |  |  |
| **ПРОБЕЛ6** | ПРОБЕЛ6 |  |  |  |  |  | **(14)**  ИМЯ2 |  |  | **(14)**  ИМЯ2 |  |
| **ИМЯ2** |  |  |  |  | **(15)**  ОСКОБКА2 |  | **(2)**  ИМЯ2 | **(2)**  ИМЯ2 |  | **(2)**  ИМЯ2 |  |
| **ОСКОБКА2** |  |  |  |  |  |  |  | **(16)**  ЦЕЛОЕ | **(17)** |  |  |
| **ЗНАК** |  |  |  |  |  |  |  | **(16)**  ЦЕЛОЕ |  |  |  |
| **ЦЕЛОЕ** |  |  |  |  |  | **(18)**  ЗСКОБКА2 |  | **(2)**  ЦЕЛОЕ |  |  |  |
| **ЗСКОБКА2** | ТЧКЗПТ | **(12)**  **;** |  |  |  |  |  |  |  |  |  |
| **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 3.8.

Таблица 3.8. Процедуры переходов обрабатывающего автомата лексического блока

|  |  |
| --- | --- |
| **Действие** | **Семантика** |
| **(1)** | Лексема(ЕСЛИ);  *Обработать;* |
| **(2)** | *Обработать;* |
| **(3)** | Лексема(ИДЕНТ1);  *Обработать;* |
| **(4)** | Лексема(ТО);  *Обработать;* |
| **(5)** | Лексема(ИДЕНТ2);  *Обработать;* |
| **(6)** | Лексема(ДВОЕТОЧИЕ);  *Обработать;* |
| **(7)** | Лексема(РАВНО);  *Обработать;* |
| **(8)** | Лексема(ИМЯ1);  *Обработать;* |
| **(9)** | Лексема(ОСКОБКА1);  *Обработать;* |
| **(10)** | Лексема(ИДЕНТ3);  *Обработать;* |
| **(11)** | Лексема(ЗСКОБКА1);  *Обработать;* |
| **(12)** | Лексема(ТЧКЗПТ);  *Обработать;* |
| **(13)** | Лексема(ИНАЧЕ);  *Обработать;* |
| **(14)** | Лексема(ИМЯ2);  *Обработать;* |
| **(15)** | Лексема(ОСКОБКА2);  *Обработать;* |
| **(16)** | Лексема(ЦЕЛОЕ);  *Обработать;* |
| **(17)** | Лексема(ЗНАК);  *Обработать;* |
| **(18)** | Лексема(ЗСКОБКА2);  *Обработать;* |

### Синтаксический блок

В таблице 3.9 представлен конечный распознаватель синтаксического блока. Пустым клеткам соответствует вызов примитивной процедуры *НЕТ*, которая отвергает цепочку.

Для удобства построения таблицы введем следующие обозначения: ТЧКЗПТ будем обозначать как ‘;’, ДВОЕТОЧИЕ как ‘:’, РАВНО как ‘=’, ОСКОБКА как ‘(’, ЗСКОБКА как ‘)’, ПОДЧЕРКИВАНИЕ как ‘\_’, ЗНАК как ‘’.

Таблица 3.9. Конечный распознаватель синтаксического блока

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ЕСЛИ | ИДЕНТ1 | ТО | ИДЕНТ2 | : | = | ИМЯ1 | ОСКОБКА1 | ИДЕНТ3 | ЗСКОБКА1 | ; | ИНАЧЕ | ИМЯ2 | ОСКОБКА2 | ЦЕЛОЕ |  | ЗСКОБКА2 |
| **НАЧ** | ЕСЛИ |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ЕСЛИ** |  | ИДЕНТ1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ИДЕНТ1** |  |  | ТО |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ТО** |  |  |  | ИДЕНТ2 |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ИДЕНТ2** |  |  |  |  | : |  |  |  |  |  |  |  |  |  |  |  |  |
| **ДВОЕТОЧИЕ** |  |  |  |  |  | = |  |  |  |  |  |  |  |  |  |  |  |
| **РАВНО** |  |  |  |  |  |  | ИМЯ1 |  |  |  |  |  |  |  |  |  |  |
| **ИМЯ1** |  |  |  |  |  |  |  | ОСКОБКА1 |  |  |  |  |  |  |  |  |  |
| **ОСКОБКА1** |  |  |  |  |  |  |  |  | ИДЕНТ3 |  |  |  |  |  |  |  |  |
| **ИДЕНТ3** |  |  |  |  |  |  |  |  |  | ЗСКОБКА1 |  |  |  |  |  |  |  |
| **ЗСКОБКА1** |  |  |  |  |  |  |  |  |  |  |  | ИНАЧЕ |  |  |  |  |  |
| **ИНАЧЕ** |  |  |  |  |  |  |  |  |  |  |  |  | ИМЯ2 |  |  |  |  |
| **ИМЯ2** |  |  |  |  |  |  |  |  |  |  |  |  |  | ОСКОБКА2 |  |  |  |
| **ОСКОБКА2** |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ЦЕЛОЕ |  |  |
| **ЗНАК** |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ЦЕЛОЕ |  |  |
| **ЦЕЛОЕ** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ЗСКОБКА2 |
| **ЗСКОБКА2** |  |  |  |  |  |  |  |  |  |  | ; |  |  |  |  |  |  |
| **ТЧКЗПТ** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

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

### Блок идентификации ключевых слов

Блок идентификации ключевых слов будет содержать в себе список всех ключевых слов языка Pascal, кроме слов “if”, “then”, “else”. Если полученный идентификатор является ключевым словом if, then или else, то подпрограмма вернет КЛСЛОВО\_if, КЛСЛОВО\_then или КЛСЛОВО\_else соответственно. В противном случае методом линейного поиска [2] программа сверит идентификатор со всеми словами из списка (Таблица 3.9) и при обнаружении совпадения вернет строку ОШИБКА. В таблице 3.10 указаны все ключевые слова, при которых блок идентификации ключевых слов будет возвращать строку *ОШИБКА.*

Таблица 3.10. Ключевые слова языка Pascal

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Absolute | downto | Xor | Nil | Set | Var |
| And | destructor | implementation | Not | Shl | While |
| Array | Const | In | Or | Shr |  |
| Asm | end | inline | Of | string |  |
| Begin | exports | interrupt | object | With |  |
| 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)

Общий размер программы составляет 1048 строк: размер головного модуля – 89 строк, размер модуля 1\_func\_Translit – 39 строк, размер модуля 2\_func\_Leks – 568 строк, размер модуля 3\_func\_KeyWordIdent – 69 строк, 4\_func\_Syntax – 283 строк. Реализация программы указан в приложении.

# Тестирование

В данном разделе приведены протоколы автономного и комплексного тестирования программы.

## Автономное тестирование

Данный пункт будет разбит на подпункты в соответствии с модульной структурой программы, каждый подпункт будет содержать протокол тестирования соответствующего модуля. Тесты изображены *символически*, без указания точного формата входных и выходных данных.

**Блок транслитерации**

В виду громоздкости представления результатов тестирования в таблице 4.1 представлены только 3 теста для модульного тестирования блока транслитерации. Выходные данные модуля совпали с ожидаемыми.

Таблица 4.1. Протокол тестирования блока транслитерации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные данные** | **Выходные данные** | **Действительный результат** | **Тест пройден?** |
| 1. | if Cond then A1:=Func(id); | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (C, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  (, ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (A, БУКВА),  (1, ЦИФРА),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  (F, БУКВА),  (u, БУКВА),  (n, БУКВА),  (c, БУКВА),  ((,ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (C, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  (, ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (A, БУКВА),  (1, ЦИФРА),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  (F, БУКВА),  (u, БУКВА),  (n, БУКВА),  (c, БУКВА),  ((,ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 2. | /if cond then input := foo(id1); | (/, ОШИБКА),  (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (c, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (i, БУКВА),  (n, БУКВА),  (p, БУКВА),  (u, БУКВА),  (t, БУКВА),  ( , ПРОБЕЛ),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  ( , ПРОБЕЛ),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (/, ОШИБКА),  (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (c, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (i, БУКВА),  (n, БУКВА),  (p, БУКВА),  (u, БУКВА),  (t, БУКВА),  ( , ПРОБЕЛ),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  ( , ПРОБЕЛ),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 3. | if cond then input := foo(id1) else progr1(-32.5); | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (c, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (i, БУКВА),  (n, БУКВА),  (p, БУКВА),  (u, БУКВА),  (t, БУКВА),  ( , ПРОБЕЛ),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  ( , ПРОБЕЛ),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  ( , ПРОБЕЛ),  (e, БУКВА),  (l, БУКВА),  (s, БУКВА),  (e, БУКВА),  ( , ПРОБЕЛ),  (p, БУКВА),  (r, БУКВА),  (o, БУКВА),  (g, БУКВА),  (r, БУКВА),  (1, ЦИФРА),  ((, ОСКОБКА),  (-, ЗНАК),  (3, ЦИФРА),  (2, ЦИФРА),  (., ОШИБКА),  (5, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (c, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (i, БУКВА),  (n, БУКВА),  (p, БУКВА),  (u, БУКВА),  (t, БУКВА),  ( , ПРОБЕЛ),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  ( , ПРОБЕЛ),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  ( , ПРОБЕЛ),  (e, БУКВА),  (l, БУКВА),  (s, БУКВА),  (e, БУКВА),  ( , ПРОБЕЛ),  (p, БУКВА),  (r, БУКВА),  (o, БУКВА),  (g, БУКВА),  (r, БУКВА),  (1, ЦИФРА),  ((, ОСКОБКА),  (-, ЗНАК),  (3, ЦИФРА),  (2, ЦИФРА),  (., ОШИБКА),  (5, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |

**Лексический блок**

В виду громоздкости представления результатов тестирования в таблице 4.2 представлены только 3 теста для модульного тестирования лексического блока. Выходные данные модуля совпали с ожидаемыми. Тесты с ошибкой, возникшей в блоке транслитерации, не приведены, поскольку при возникновении ошибки не произойдет вызова лексического блока.

Таблица 4.2. Протокол тестирования лексического блока

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные данные** | **Выходные данные** | **Действительный результат** | **Тест пройден?** |
| 1. | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (C, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  (, ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (A, БУКВА),  (1, ЦИФРА),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  (F, БУКВА),  (u, БУКВА),  (n, БУКВА),  (c, БУКВА),  ((,ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, ИДЕНТИФИКАТОР),  (Cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (A1, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (Func, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, ИДЕНТИФИКАТОР),  (Cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (A1, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (Func, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 2. | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (c, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (i, БУКВА),  (n, БУКВА),  (p, БУКВА),  (u, БУКВА),  (t, БУКВА),  ( , ПРОБЕЛ),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  ( , ПРОБЕЛ),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  ( , ПРОБЕЛ),  (e, БУКВА),  (l, БУКВА),  (s, БУКВА),  (e, БУКВА),  ( , ПРОБЕЛ),  (p, БУКВА),  (r, БУКВА),  (o, БУКВА),  (g, БУКВА),  (r, БУКВА),  (1, ЦИФРА),  ((, ОСКОБКА),  (-, ЗНАК),  (3, ЦИФРА),  (2, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, ИДЕНТИФИКАТОР),  (cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (input, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (foo, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id1, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (else, ИДЕНТИФИКАТОР),  (progr1, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (-32, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, ИДЕНТИФИКАТОР),  (cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (input, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (foo, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id1, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (else, ИДЕНТИФИКАТОР),  (progr1, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (-32, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 3. | (i, БУКВА),  (f, БУКВА),  ( , ПРОБЕЛ),  (-, ЗНАК),  (C, БУКВА),  (o, БУКВА),  (n, БУКВА),  (d, БУКВА),  ( , ПРОБЕЛ),  (t, БУКВА),  (h, БУКВА),  (e, БУКВА),  (n, БУКВА),  ( , ПРОБЕЛ),  (m, БУКВА),  (y, БУКВА),  (v, БУКВА),  (a, БУКВА),  (r, БУКВА),  (:, ДВОЕТОЧИЕ),  (=, РАВНО),  (f, БУКВА),  (o, БУКВА),  (o, БУКВА),  ((, ОСКОБКА),  (i, БУКВА),  (d, БУКВА),  (1, ЦИФРА),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, ИДЕНТИФИКАТОР),  (-, ОШИБКА) | (if, ИДЕНТИФИКАТОР),  (-, ОШИБКА) | Да |

**Блок идентификации ключевых слов**

В виду громоздкости представления результатов тестирования в таблице 4.3 представлены только 3 теста для модульного тестирования блока идентификации ключевых слов. Выходные данные модуля совпали с ожидаемыми. Тесты с ошибкой, возникшей в лексическом блоке, не приведены, поскольку при возникновении ошибки не произойдет вызова блока идентификации ключевых слов.

Таблица 4.3. Протокол тестирования блока идентификации ключевых слов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные данные** | **Выходные данные** | **Действительный результат** | **Тест пройден?** |
| 1. | (if, ИДЕНТИФИКАТОР),  (Cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (A1, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (Func, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (Cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (A1, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (Func, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (Cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (A1, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (Func, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 2. | (if, ИДЕНТИФИКАТОР),  (cond, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (input, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (foo, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id1, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (else, ИДЕНТИФИКАТОР),  (progr1, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (-32, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (input, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (foo, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (else, КЛСЛОВО\_else),  (progr1, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (-32, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (input, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (foo, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (else, КЛСЛОВО\_else),  (progr1, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (-32, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |
| 3. | (if, ИДЕНТИФИКАТОР),  (var, ИДЕНТИФИКАТОР),  (then, ИДЕНТИФИКАТОР),  (myvar, ИДЕНТИФИКАТОР),  (:=, ПРИСВОЕНИЕ),  (foo, ИДЕНТИФИКАТОР),  ((, ОСКОБКА),  (id1, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (var, ОШИБКА),  (then, КЛСЛОВО\_then),  (myvar, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (foo, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | (if, КЛСЛОВО\_if),  (var, ОШИБКА),  (then, КЛСЛОВО\_then),  (myvar, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (foo, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | Да |

**Синтаксический блок**

В виду громоздкости представления результатов тестирования в таблице 4.4 представлены только 3 теста для модульного тестирования синтаксического блока. Выходные данные модуля совпали с ожидаемыми. Тесты с ошибкой, возникшей в блоке идентификации ключевых слов, не приведены, поскольку при возникновении ошибки не произойдет вызова синтаксического блока, цепочка уже отвергнута.

Таблица 4.4. Протокол тестирования синтаксического блока

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные данные** | **Выходные данные** | **Действительный результат** | **Тест пройден?** |
| 1. | (if, КЛСЛОВО\_if))  (Cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (samp, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (func3, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (else, КЛСЛОВО\_else),  (prog, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (-322, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | ACCEPT | ACCEPT | Да |
| 2. | (if, КЛСЛОВО\_if))  (Cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (samp, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  ((, ОСКОБКА),  (id1, ИДЕНТИФИКАТОР),  (), ЗСКОБКА),  (else, КЛСЛОВО\_else),  (prog, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (-322, ЦЕЛОЕ),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | REJECT | REJECT | Да |
| 3. | (if, КЛСЛОВО\_if))  (Cond, НЕ\_КЛСЛОВО),  (then, КЛСЛОВО\_then),  (samp, НЕ\_КЛСЛОВО),  (:=, ПРИСВОЕНИЕ),  (func3, НЕ\_КЛСЛОВО),  ((, ОСКОБКА),  (id1, НЕ\_КЛСЛОВО),  (), ЗСКОБКА),  (;, ТЧКЗПТ) | ACCEPT | ACCEPT | Да |

## Комплексное тестирование

В таблице 4.5 представлен протокол тестирования головного модуля. Всего было проведено 9 тестов, в ходе которых ошибок обнаружено не было.

Таблица 4.5. Протокол тестирования головного модуля

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **№**  **п/п** | **Входные**  **данные** | **Выходные**  **данные** | **Действительный**  **Результат** | **Тест**  **пройден?** |
| 1. | if Cond then A1:=Func(id); | ACCEPT | ACCEPT | Да |
| 2. | if Cond then myvar:=foo(id1) else proc(2); | ACCEPT | ACCEPT | Да |
| 3. | if Cond then samp:=func3(id1) else proc(-32); | ACCEPT | ACCEPT | Да |
| 4. | if \_cond2 then v := prog(inp) ; | ACCEPT | ACCEPT | Да |
| 5. | if Cond then var :=foo(id1) else proc(5); | REJECT | REJECT | Да |
| 6. | if 33ad then var:=f(id2); | REJECT | REJECT | Да |
| 7. | iff cond then input := foo(id1) ; | REJECT | REJECT | Да |
| 8. | /if cond then input := foo(id1) ; | REJECT | REJECT | Да |
| 9. | if cond input := foo(id1) ; | REJECT | REJECT | Да |

# Заключение

В результате проделанной работы был разработан конечный распознаватель заданной символьной цепочки – готовый рабочий программный продукт. В нём были реализованы все требования, установленные в постановке задачи. В результате анализа требований была составлена спецификация, спроектирована модульная структура программы, описаны алгоритмы реализации всех модулей. Выполнена реализация программного кода, проведено автономное тестирование основных модулей и комплексное тестирование программы, её отладка.

В результате тестирования ошибок выявлено не было.

# Литература

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.