|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК-6)

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

**по курсу «****Программное обеспечение систем автоматизированного проектирования»**

Студент Камалов Антон Павлович

Группа РК6-35Б

Тип задания Лабораторная работа №4

Вариант Y31

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Камалов А.П.**

*подпись, дата*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ Князева С.В.**

*подпись, дата*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Москва, 2023 г.*

**Оглавление**

[**Введение** 3](#_Toc149595148)

[**Основания для разработки** 3](#_Toc149595149)

[**Назначение разработки** 3](#_Toc149595150)

[**Требования к программе** 3](#_Toc149595151)

[**1. Требования к функциональным характеристикам** 3](#_Toc149595152)

[**2. Требования к надежности** 4](#_Toc149595153)

[**3. Требования к составу и параметрам технических средств** 4](#_Toc149595154)

[**4. Условия эксплуатации** 4](#_Toc149595155)

[**5. Требования к информационной и программной совместимости** 4](#_Toc149595156)

[**Порядок контроля и приема** 7](#_Toc149595157)

[**Список используемой литературы** 7](#_Toc149595158)

[**Приложение №1 Результаты разработки** 7](#_Toc149595159)

[**Приложение №2** **Содержимое файла спецификации** 9](#_Toc149595160)

**Задание**

Разработать синтаксический анализатор для грамматического разбора входного потока цифровых данных с целью распознавания записей двоичных наборов, которые соответствуют формальному языку **L={1 | n>0}**. Записи двоичных наборов должны передаваться синтаксическому анализатору строками потока стандартного ввода. Результаты грамматического разбора входного потока должны отображать диагностические сообщения в потоке стандартного вывода.

# **Введение**

Настоящий документ определяет техническое задание на разработку программы синтаксического анализатора (далее по тексту - программа Y31) для распознания заданных двоичных наборов. Программа Y31 разрабатывается в рамках лабораторной работы по курсу «Программное обеспечение систем автоматизированного проектирования» для практического изучения этапов лексического и синтаксического анализа в процедурах трансляции формальных языков.

# **Основания для разработки**

Программа Y31 разрабатывается в рамках лабораторной работы по курсу «Программное обеспечение систем автоматизированного проектирования» для практического изучения этапов лексического и синтаксического анализа в процедурах трансляции формальных языков.

# **Назначение разработки**

Программа Y31 предназначается для реализации грамматического разбора, заключающегося в распознавании необходимых двоичных наборов чисел, в системах и модулях обработки двоичных последовательностей и бинарных файлов.

# **Требования к программе**

## **1. Требования к функциональным характеристикам**

1.1. Программа Y31 должна в интерактивном режиме распознавать двоичные наборы, соответствующие формальному языка.

1.2. Двоичные наборы должны передаваться строками стандартного ввода без разделителей и иных символов, кроме символа нуля, единицы и перевода строки.

1.3. Программа Y31 обнаруживает соответствие или несоответствие бинарной последовательности заданным правилам и ошибку ввода.

1.4. При нарушении ввода или несоответствии последовательности заданным правилам программа Y31 должна отображать диагностическое сообщение «INCORRECT» в потоке стандартного вывода.

1.5. Результат распознавания программой Y31 бинарной последовательности, соответствующей заданным правилам, должен сопровождаться отображением в потоке стандартного вывода информационным сообщением «CORRECT».

## **2. Требования к надежности**

Программа Y31 не должна иметь каких-либо ограничений по числу символов в анализируемой бинарной последовательности, кроме внутренних ограничений инструментальных средств, использованных для её реализации.

## **3. Требования к составу и параметрам технических средств**

Программа Y31 должна быть разработана исходя из возможности реализации на стандартном составе технических средств компьютеров любой архитектуры после соответствующей трансляции исходного кода.

## **4. Условия эксплуатации**

4.1. Программа Y31 должна быть ориентирована на эксплуатацию в среде OS UNIX.

4.2. Программа Y31 должна быть реализована в виде выполняемого файла с именем Y31, по которому она должна вызываться средствами любого командного процессора OS UNIX.

4.3. Программа Y31 должна эксплуатироваться в интерактивном режиме, читая строки из потока стандартного ввода и отображая результаты их обработки в потоке стандартного вывода.

## **5. Требования к информационной и программной совместимости**

* 1. Чтобы обеспечить выполнение требуемых технических характеристик, программа Y31 должна реализовывать синтаксический анализ любой входной строки бинарного выражения из потока стандартного ввода.
  2. Синтаксический анализатор программы Y31 должен обеспечивать грамматический разбор двоичных наборов с целью установить соответствие или несоответствие содержащей их строки потока стандартного ввода заданным правилам.
  3. Для выполнения грамматического разбора синтаксический анализатор программы Y31 должен реализовывать однозначную контекстно-свободную грамматику, которая ориентирована на обработку строки заданной бинарной последовательности из потока стандартного ввода (далее по тексту - БП).
  4. Грамматику БП синтаксического анализатора программы Y31 должны составлять следующие элементы:
* терминальные символы (терминалы), соответствующие структурным единицам (лексемам) входного выражения;
* нетерминальный символ (нетерминал), к которому приводится входное выражение;
* нетерминальные символы (нетерминалы), обозначающие допустимые варианты комбинации лексем во входном бинарном выражении;
* система продукций (правил вывода), обеспечивающая грамматический разбор входного бинарного выражения.
  1. Терминальные символы грамматики БП синтаксического анализатора программы Y31 должны представляться лексемами, специфицированными следующими литералами: ‘1’ и ‘0’, обозначающие код ASCII единицы и нуля соответственно.
  2. Нетерминальные символы грамматики БП должны обозначаться следующими именами: input, expr, expr\_n. Они должны определяться следующим образом:

input: пустая строка

expr: двоичная последовательность

expr\_n: двоичная последовательность

* 1. Нетерминал грамматики БП синтаксического анализатора программы Y31 должен обозначаться именем line. Он должен выводиться из любой корректной или ошибочной последовательности.
  2. Система продукций грамматики БП синтаксического анализатора программы Y31 должна обеспечивать грамматический разбор произвольной входной строки потока стандартного ввода путем приведения терминалов и нетерминалов к начальному нетерминалу методом GLR(1) анализа. Приведение входной строки к начальному нетерминалу означает успех грамматического разбора, альтернативный результат - ошибку.
  3. Для разработки синтаксического анализатора программы Y31 необходимо использовать генератор синтаксических анализаторов (далее по тексту - YACC) из состава OS UNIX, инструментальные средства которого ориентированы на обработку файла спецификаций (далее по тексту - Yacc-файл) проектируемого синтаксического анализатора.
  4. При разработке синтаксического анализатора программы Y31 необходимо составить Yacc-файл, отражающий специфику грамматического разбора бинарного выражения, и сохранить его под именем Y31.y в выбранном доступном рабочем каталоге файловой системы OS UNIX.
  5. Проектируемый Yacc-файл Y31.y должен состоять из 3-х секций: деклараций, правил и функций. разделителем секций должны быть символические пары %%, расположенные в начальных позициях содержащих их строк Yacc-файла.
  6. Секция деклараций Yacc-файла Y31.y должна включать:
* спецификацию блока внешних описаний, ограниченную директивами %{ и %}, в которой необходимо подключить библиотеку ввода/вывода stdio.h, а также явно объявить функции yyerror и yylex;
* объявить с помощью директивы %token терминальные символы ZERO и ONE, соответствующие нулю и единице.
  1. В секции правил Yacc-файла Y31.y должны быть приведены описания продукций приведения нетерминалов грамматики БП.
  2. Каждая продукция секции правил Yacc-файла Y31.y должна быть задана в нотации, близкой к форме Бэкуса-Наура, где в левой части указывается приводимый нетерминал, а в правой - последовательность терминалов и/или нетерминалов грамматики БП, которые перечисляются через пробел. Для разделения частей продукции должен использоваться символ двоеточия (:). Каждую продукцию нужно начинать с новой строки и завершать либо символом точки с запятой (;), либо блоком действий в фигурных скобках.
  3. Продукции секции правил Yacc-файла Y31.y, приведение нетерминалов которых необходимо сопровождать функциональной обработкой, должны содержать блоки действий. Блоки действий должны располагаться в правых частях продукций и ограничиваться парой фигурных БП. Внутри блоков действий можно использовать любые конструкции и вызовы функций программирования С. Например, правило определения пустой строки:

'\n' {printf("EMPTY LINE\n");}

* 1. Продукции секции правил Yacc-файла Y31.y, необходимые для приведения нетерминалов, должны быть специфицированы с помощью альтернативных правил. В каждом из них альтернативы свертки различных нетерминалов правой части должны быть объединены с помощью оператора ИЛИ, который обозначается символом вертикальной черты (|).

Например, альтернативную продукцию приведения нетерминала expr\_n следует реализовать путем сочетания 2-х правил:

expr\_n:

expr

| /\* epsilon \*/

;

* 1. Секции функций Yacc-файла Y31.y должна содержать исходный код, оформленный по правилам системы программирования C, для 3-х функций с предопределенными именами: yylex(), yyerror() и main(), которые должны иметь целочисленный (типа int) код возврата. Исходный код перечисленных функций используется непосредственно для формирования кода программы Y31.
  2. Для достижения целей лексического анализа, указанных в п. 5.17., исходный код функции yylex() должен предусматривать:
* побайтное чтение любой заданной строки входного бинарного набора из потока стандартного ввода с помощью библиотечной функции getchar();
* распознавание единиц во входной строке с помощью оператора switch;
* возврат кодов ASCII обнаруженных символов либо лексемы END в случае конца строки.

5.19. Исходный код секции функций Yacc-файла Y31.y образует исходный код программы Y31, который должен формироваться генератором синтаксических анализаторов YACC в файле с предопределенным именем y.tab.c в текущем рабочем каталоге файловой системы OS UNIX. Выполняемый модуль программы Y31 должен быть создан по файлу исходного кода y.tab.c в выполняемом файле Y31 средствами компилирующей системы программирования C.

**Стадии и этапы разработки**

Процесс разработки программы Y31 целесообразно разделить на следующие 3 стадии:

* составить Yacc-файл в выбранном рабочем каталоге файловой системы OS

UNIX, используя любой текстовый редактор;

* получить исходный код синтаксического анализатора в файле y.tab.c текущего каталога файловой системы OS UNIX, обработав Yacc-файл командой yacc следующим образом:

**$ yacc Y31.y**

* сформировать выполняемый модуль в файле текущего каталога файловой системы OS UNIX, компилируя исходный код синтаксического анализатора следующей командой:

**$ cc y.tab.c -o Y31**

Результаты разработки программы Y31 должны содержать описание грамматики и файл спецификаций для генератора синтаксических анализаторов YACC.

# **Порядок контроля и приема**

1. Для проверки функционирования программы Y31 должны быть предложены контрольные примеры, предусматривающие стандартный ввод корректных и некорректных бинарных выражений.
2. Для приёмки программы Y31 должен быть организован вызов выполняемого файла Y31 в консольном режиме работы OS UNIX.

# **Список используемой литературы**

1. Родионов C.В., Волосатова Т.М. Автоматизация проектирования лексических анализаторов. Москва: Издательство МГТУ им. Н. Э. Баумана, 2005.
2. Волосатова Т.М., Родионов С.В.Лингвистическое обеспечение САПР. Режим доступа: http://bigor.bmstu.ru/?cnt/?doc=LO-SAPR/base.cou

## **Приложение №1 Результаты разработки**

**Описание структуры программы**

Секция объявлений:

* stdio.h - для работы с функциями стандартного ввода/вывода;
* %token ZERO ONE- терминалы, обозначающий цифры 0 и 1.

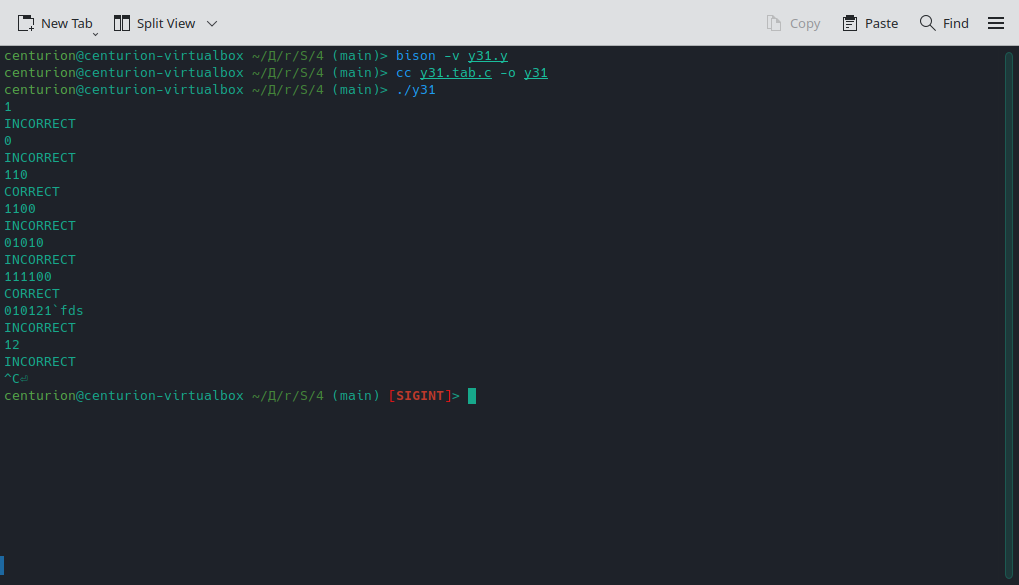
Секция правил:

* input - описывает входную строку;
* line - содержит различные варианты входной строки и действия при этих вариантах. Для пустой строки будет выведено сообщение о том, что строка пустая. Для строки, не соответствующей правилу, будет сброшен флаг ошибки, чтобы программа предложила повторный ввод. Для верной строки будет выведено соответствующее сообщение;
* expr, expr\_n - правила, рекурсивно описывающие требуемые двоичные наборы.

Секция программ:

* int yylex() - функция, реализующая лексический анализатор. Возвращает номер нетерминала, если он является нужной буквой, или символ в другом случае;
* int yyerror() - функция, вызываемая при неправильном вводе. Выводит на экран сообщение об ошибке;
* int main() - основная функция программы. Вызывает функцию yyparse(), которая реализует синтаксический анализатор.

**Результаты синтаксического анализа**



**Приложение №2**

**Содержимое файла спецификации:**

%{

#include <stdio.h>

int yyerror(char\*);

int yylex();

%}

%start input

%token ONE ZERO

%%

input:

input line

| /\* epsilon \*/

;

line:

'\n' {printf("EMPTY LINE\n");}

| error '\n'{yyerrok;}

| expr {printf("CORRECT\n");}

;

expr:

ONE ONE expr\_n ZERO;

expr\_n:

expr

| /\* epsilon \*/

;

%%

int yyerror(char \*s){

printf("INCORRECT\n");

return 1;

}

int yylex(){

int c;

c = getchar();

switch(c){

case '1' : return ONE;

case '0' : return ZERO;

case EOF : return 0;

default : return c;

}

}

int main(){

yyparse();

}