Министерство образования и науки 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» Кафедра «Системы автоматизированного проектирования»

# Отчет

**по лабораторной работе №5,6 по дисциплине**

# «Программное обеспечение САПР» Вариант T5

Выполнил: студент группы РК6-32 Журавлев Н.В.

Проверил: старший преподаватель Родионов С.В.

Москва

2020

## Техническое задание

## Постановка задачи

Разработать транслятор для преобразования записи любой десятичной дроби с необязательным знаком в сумму ее целой и дробной части. Записи десятичных дробей в любом допустимом формате с фиксированной точкой должны передаваться транслятору строками потока стандартного ввода. Результаты трансляции должны отображаться строками потока стандартного вывода, где целая и дробная части разделены знаком '+'.

**Введение**

Настоящий документ определяет техническое задание на разработку программы транслятора десятичных дробей (далее по тексту - программа T5).

## Основания для разработки

Программа разрабатывается в рамках лабораторной работы по курсу "Лингвистическое обеспечение САПР" для практического изучения этапов лексического и синтаксического анализа в процедурах трансляции формальных языков.

## Назначение разработки

Программа предназначается для реализации лексического и грамматического разбора, заключающегося в распознавании необходимых десятичных наборов чисел, в системах и модулях обработки десятичных последовательностей.

## Требования к программе

Требования к функциональным характеристикам

1. Программа T5 должна в интерактивном режиме распознавать и преображать десятичные дроби в необходимый формат.
2. Десятичные дроби должны быть в любом допустимом формате с фиксированной точкой с необязательным знаком («-»).
3. Записи десятичных дробей должны передаваться транслятору в любом допустимом формате с фиксированной точкой строками потока стандартного ввода.
4. Результаты трансляции должны отображаться суммой целого числа и его дробной части в потоке стандартного вывода.

Требования к надежности

Программа T5 не должна иметь каких-либо ограничений по количеству цифр в транслируемой десятичной дроби, кроме внутренних ограничений инструментальных средств, использованных для ее реализации.

Требования к составу и параметрам технических средств

Программа T5 должна быть разработана исходя из возможности реализации на стандартном составе технических средств компьютеров любой архитектуры, после соответствующей трансляции исходного кода.

Условия эксплуатации

1. Программа T5 должна быть ориентирована на эксплуатацию в среде OS UNIX.
2. Программа T5 должна быть реализована в виде выполняемого файла с именем t5, по которому она должна вызываться средствами любого командного процессора OS UNIX.
3. Программа T5 должна эксплуатироваться в интерактивном режиме, читая строки из потока стандартного ввода и отображая результаты их обработки в потоке стандартного вывода.

Требования к информационной и программной совместимости

1. Чтобы обеспечить выполнение требуемых технических характеристик, программа T5 должна реализовывать лексический и синтаксический анализ входных строк из потока стандартного ввода.
2. Лексический анализатор программы T5 должен обеспечивать распознавание лексем, соответствующих формату структурных элементов десятичных дробей.
3. Лексический анализатор программы T5 должен обеспечивать выделение следующей лексемы: INTEGER – отрицательное число до знака “.”

DROP – положительное число и число после знака “.”

END – символы разделители

Макроопределения значений типов лексем следует сосредоточить в заголовочном файле y.tab.h,

который должен формироваться при разработке синтаксичекого анализатора программы T5.

1. Для сохранения значений лексем следует использовать внешнюю (extern) целочисленную (типа int)

переменную yylval.

1. Для разработки лексического анализатора программы T5, необходимо использовать генератор лексических анализаторов (далее по тексту - LEX) OS UNIX, инструментальные средства которого должны быть ориентированы на обработку файла спецификаций проектируемого лексического анализатора (далее по тексту - Lex-файл).
2. При разработке лексического анализатора программы T5 необходимо составить Lex-файл, отражающий специфику лексического анализа десятичных дробей, и сохранить его под именем t5.l в выбранном рабочем каталоге файловой системы OS UNIX.
3. Проектируемый Lex-файл t5.l должен состоять из 2-х разделов: раздел деклараций и раздел правил.

Разделы Lex-файла должны отделяться символической парой %%.

1. В разделе деклараций Lex-файла t5.l необходимо специфицировать блок внешних описаний, ограничив его директивами %{ и %}, в котором нужно объявить внешнюю целочисленную переменную yylval для хранения значений лексем, и включить заголовочные файл y.tab.h для макроопределений терминалов и stdlib.h для функции преобразования строки в число (atoi), и

с помощью директивы "#include" препроцессора системы программирования C.

1. В разделе правил Lex-файла t5.l должны быть введены правила, которые обеспечивают распознавание лексем, перечисленных в п. 3, с помощью шаблонов регулярных выражений и необходимую функциональную обработку переменных в блоках действий правил.
2. Блоки действий правил Lex-файла t5.l должны обеспечивать:

сохранение значений внешней переменной yylval, которые возвращаются функцией atoi,

преобразовывающей yytext в число;

возврат типа распознанной лексемы оператором return, аргумент которого должен соответствовать макроопределению лексемы из заголовочного файла y.tab.h;

Каждый блок действий должен быть ограничен парой фигурных скобок, внутри которых допустимо использовать любые конструкции и вызовы библиотечных функций системы программирования.C.   
 Вызов лексического анализатора должен осуществляться путем обращения к функции yylex() в синтаксическом анализаторе программы T9. Связь между синтаксическим и лексическим анализаторами должна осуществляться по типу лексемы, определяемому кодом возврата функции yylex(), и по значению лексемы, который передается через внешнюю переменную yylval.

1. Синтаксический анализатор программы T5 должен обеспечивать трансляцию потока лексем от лексического анализатора в необходимый формат.
2. Для выполнения трансляции разбора синтаксический анализатор программы T5 должен реализовывать однозначную контекстно-свободную грамматику, которая ориентирована на построчную обработку десятичных дробей из потока стандартного ввода и далее по тексту называется грамматикой дробей.
3. Грамматику дробей синтаксического анализатора программы T5 должны составлять следующие элементы: нетерминалы, терминалы (лексемы, выделенные лексическим анализатором), система продукций (правил вывода), которые должны обеспечивать вывод входной строки стандартного ввода путем приведения терминалов и нетерминалов к аксиоме грамматики.
4. Приведение потока лексем к начальному нетерминалу с помощью продукций грамматики дробей синтаксического анализатора программы T5 должно означать успешное завершение грамматического разбора, т.е. распознавание корректной целочисленной дроби. Альтернативный результат должен рассматриваться как синтаксическая ошибка формата введенной дроби.
5. Нетерминалы должны определяться продукциями грамматики дробей синтаксического анализатора программы T5 следующим образом:

input: начальный символ грамматики;  
exp: состав дроби

1. Начальный нетерминал input должен определяться продукциями грамматики дробей синтаксического анализатора программы T5 как входная строка, которая может соответствовать терминалу exp, терминалу error или быть пустой. Приведение по 1-й альтернативе (exp) означает успех грамматического разбора.
2. Для разработки синтаксического анализатора программы T5, 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.
3. При разработке синтаксического анализатора программы T5 необходимо составить Yacc-файл, отражающий специфику грамматического разбора целочисленных дробей, и сохранить его под именем t5.y в выбранном рабочем каталоге файловой системы OS UNIX.
4. Проектируемый Yacc-файл t5.y должен состоять из 3-х секций: секция деклараций, секция правил и секция функций. Разделителем секций должна быть символическая пара %%.
5. Секция деклараций Yacc-файла t5.y должна включать:

перечисление терминалов грамматики дробей с помощью директив(ы) %token;

cпецификацию блока внешних описаний, ограниченную директивами %{ и %}, в котором необходимо определить функции yylex для лексического анализа и yyerror для вывода ошибки, а также подключить библиотеку ввода-вывода stdio.h.

1. В секции правил Yacc-файла t5.y должны быть приведены описания продукций приведения нетерминалов грамматики дробей.
2. Каждая продукция секции правил Yacc-файла t5.y должна быть задана в нотации, близкой к форме Бэкуса-Наура, где в левой части указывается приводимый нетерминал, а в правой - последовательность терминалов и нетерминалов грамматики дробей, которые перечисляются через пробел. В частном случае, правая часть может быть пустой, если необходимо построить пустую продукцию. 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.

1. Альтернативные продукции (с одинаковыми нетерминалами в левых частях правил), необходимые для приведения нетерминалов input, exp, должны быть объединены в секции правил Yacc-файла t5.y с помощью оператора ИЛИ, который обозначается символом вертикальной черты (|).
2. Продукции секции правил Yacc-файла t5.y, приведение нетерминалов которых необходимо сопровождать функциональной обработкой, должны содержать блоки действий. Блоки действий должны располагаться в правых частях продукций и ограничиваться парой фигурных скобок.
3. Внутри блока действий продукции приведения десятичной дроби к нужному формату необходимо использовать функцию форматного вывода printf и псевдопеременные $1..$n, содержащие семантические значения терминалов, для вывода строки в требуемом формате.
4. Блок действия альтернативной продукции приведения начального нетерминала input через терминал error может содержать макрос yyerrok для сброса состояния ошибки.
5. Вызов синтаксического анализатора должен осуществляться путем обращения к функции yyparse() в основной функции main() программы T5.
6. Функция yyparse() должна возврашать в основную функцию main() программы T5 код 0 при успехе грамматического разбора каждой входной строки потока стандартного ввода и 1 - при обнаружении синтаксических ошибок.

## Стадии и этапы разработки

Процесс разработки программы T5 целесообразно разделяться на следующие 3 стадии:

1. составить Yacc-файл t5.y в выбранном рабочем каталоге файловой системы OS UNIX, используя любой текстовый редактор;
2. получить исходный код синтаксического анализатора в файле y.tab.cи исходный код необходимых для лексического анализатора директив препроцессора в заголовочном файле y.tab.h текущего каталога файловой системы OS UNIX, обработав Yacc-файл t9.y командой yacc, следующим образом:

*$ yacc -d t5.y*

1. сформировать выполняемый модуль в файле t5 текущего каталога файловой системы OS UNIX,

компилируя исходный код синтаксического анализатора в файле y.tab.c следующей командой:

*$ cc -o t5 y.tab.c lex.yy.cc –ly*

Результаты разработки программы T5 должны содержать описание грамматики и файл спецификаций для генератора синтаксических анализаторов YACC.

## Порядок контроля и приемки

1. Для проверки функционирования программы T5 должны быть предложены контрольные примеры, предусматривающие стандартный ввод корректных и некорректных скобочных выражений с учетом вложенности и конкатенации блоков скобок различных типов.
2. Для приемки программы T5 должен быть организован вызов выполняемого файла t5 в консольном режиме работы OS UNIX или режиме эмуляции терминала операционной среды X Window System.

## Результаты разработки

## Описание грамматики

Терминальный алфавит: Σ = {INTEGER DROP END} Нетерминальный алфавит: {input exp};  
 Правила:

1. input → exp (Свертка в начальный нетерминал для вывода целого числа)
2. exp → INTEGER '.' DROP END (получение отрицательной десятичной дроби из потока стандартного ввода)
3. exp → DROP '.' DROP END (получение положительной десятичной дроби из потока стандартного ввода)

Начальный нетерминал: input

## Результаты трансляции

|  |  |
| --- | --- |
| **Входные данные** | **Выходные данные** |
| 1.1 | 1 + 0.1 |
| -1.1 | -1 + (-0.1) |
| 13.5 | 13+ 0.5 |
| -13.5 | -13 + (-0.5) |

## Содержимое файла спецификации lex

%{

#include "y.tab.h"

#include <stdlib.h>

extern int yylval;

%}

%%

-[1-9][0-9]\* |

-0 {yylval = atoi(yytext); return INTEGER; }

[^0-9\n] {return yytext[0];}

[\n ] {return END;}

[0-9]+ {yylval = atoi(yytext); return DROP; }

%%

int yywrap(){

return 1;}

## Содержимое файла спецификации yacc

## %{

## #include<stdio.h>

## void yyerror(char const \*);

## int yylex();

## %}

## %token INTEGER DROP END;

## %start input

## %%

## input: {printf("Enter the line:\n");}

## | input exp {printf("\nEnter the line:\n");}

## ;

## 

## exp: INTEGER '.' DROP END {printf("Result: %d + (-0,%d)\n", $1, $3);}

## | DROP '.' DROP END {printf("Result: %d + 0,%d\n", $1, $3);}

## | error END {yyerrok;}

## %%

## void yyerror(char const \*s) {

## fprintf(stderr, "%s\n", s);

## }

## int main() {

## yyparse();

## }Список литературы

1. База и Генератор Образовательных Ресурсов (bigor.bmstu.ru)
2. Генератор программ синтаксического анализа yacc. Производственно-внедренческий кооператив"  
   И Н Т Е Р Ф Е Й С": Демос/P 2.1, 1988
3. Т.М. Волосатова, С.В. Родионов «Автоматизация проектирования лексических анализаторов»: Издательство МГТУ им. Н.Э. Баумана, 2005