	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 Робототехники и комплексной автоматизации

КАФЕДРА Системы автоматизированного проектирования (РК-6)

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

Студент Платонова Елена Павловна

Группа РК6-31Б

Тип задания лабораторная работа

Вариант лабораторной работы А15(В)

Студент _____ **Платонова Е.П.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Волосатова Т.М.**
подпись, дата *фамилия, и.о.*

Оценка _____

Москва, 2019 г.

Оглавление

Задание к лабораторной работе

Постановка задачи

Разработать конечный автомат для распознавания во входном потоке цифровой информации записей двоичных наборов, где суммарное количество нулей и единиц четно. Записи всех двоичных наборов должны передаваться конечному автомату строками потока стандартного ввода. Результаты их распознавания должны отображаться через поток стандартного вывода. Программная реализация конечного автомата должна формироваться генератором YACC по правилам регулярной грамматики.

Основания для разработки

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

Назначение разработки

Программа BINSET предназначается для реализации конечного автомата, распознающего необходимые записи двоичных наборов, в системах и модулях обработки бинарных файлов.

Требования к программе

1. Требования к функциональным характеристикам

1.1. Программа BINSET должна в интерактивном режиме распознавать двоичные наборы, соответствующие ТЗ

1.2. Двоичные наборы должны содержать произвольное натуральное количество разрядов.

1.3. Суммарное количество 0 и 1 четно.

1.4. Двоичные наборы должны передаваться строками стандартного ввода через знаки пробелов и табуляций.

1.5. Программа BINSET обнаруживает соответствие или несоответствие двоичной последовательности заданным правилам.

1.6. При нарушении ввода или несоответствии последовательности заданным правилам, программа BINSET должна игнорировать эти данные и не выводить их в поток стандартного вывода.

1.7. Результат распознавания программой BINSET двоичной последовательности, соответствующей заданным правилам, должен отображаться в колонку в поток стандартного вывода

2. Требования к надежности

Программа BINSET не должна иметь каких-либо ограничений по числу символов в анализируемой двоичной последовательности, кроме внутренних ограничений инструментальных средств, использованных для ее реализации.

3. Требования к составу и параметрам технических средств

Программа BINSET 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 BINSET должна быть ориентирована на эксплуатацию в среде OS UNIX.

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

4.3. Программа BINSET 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

5.1. Чтобы обеспечить выполнение требуемых технических характеристик, программы BINSET должны реализовывать синтаксический анализ любой входной строки бинарного выражения из потока стандартного ввода.

5.2. Синтаксический анализатор программ BINSET должен обеспечивать грамматический разбор двоичных последовательностей с целью установить

соответствие или несоответствие содержащей их строки потока стандартного ввода заданным правилам.

5.3. Для выполнения грамматического разбора синтаксический анализатор программ BINSET должен реализовывать однозначную контекстно-свободную грамматику, которая ориентирована на обработку строки заданной бинарной последовательности из потока стандартного ввода, и далее по тексту называется грамматикой БП.

5.4. Грамматику БП синтаксического анализатора программ BINSET должны составлять следующие элементы:

- терминальные символы (терминалы), соответствующие структурным единицам (лексемам) входного выражения;
- начальный нетерминальный символ (начальный нетерминал), к которому приводится входное выражение;
- нетерминальные символы (нетерминалы), обозначающие допустимые варианты комбинации лексем во входном скобочном выражении;
- система продукций (правил вывода), обеспечивающая грамматический разбор входного скобочного выражения.

5.5. Терминальные символы грамматики БП синтаксического анализатора программ BINSET должны представляться лексемами, специфицированными следующими литералами:

- ‘0’, ’1’, ’\n’

  которые обозначают коды ASCII нуля, единицы и перевода строки.

5.6. Нетерминальные символы грамматики БП должны обозначаться следующими именами

s, p, q.

Они должны определяться следующим образом:

- p : сумма нулей и единиц четно;
- q : сумма нулей и единиц нечетно;

- s – начальный нетерминал;

5.7. Начальный нетерминал грамматики БП синтаксического анализатора программ BINSET должен обозначаться именем s. Он должен выводиться из любой корректной последовательности.

5.8. Система продукций грамматики БП синтаксического анализатора программ BINSET должна обеспечивать грамматический разбор произвольной входной строки потока стандартного ввода путем приведения терминалов и нетерминалов к начальному нетерминалу с помощью левостороннего вывода. Также отдельно должна быть создана программа с аналогичными функциями, реализующая разбор с помощью левостороннего вывода.

5.9. Для разработки синтаксического анализатора программ BINSET, 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.

5.10. При разработке синтаксического анализатора программ BINSET необходимо составить два Yacc-файла, отражающих специфику грамматического разбора с помощью правостороннего и левостороннего выводов, и сохранить их под именем binsetL.y и binsetR.y соответственно в выбранном доступном рабочем каталоге файловой системы OS UNIX.

5.11. Проектируемые Yacc-файлы должны состоять из 3-х секций: деклараций, правил и функций. Разделителем секций должны быть символические пары %%, расположенные в начальных позициях содержащих их строк Yacc-файла.

5.12. Секция деклараций Yacc-файлов должна включать:

спецификацию блока внешних описаний, ограниченную директивами %{ и %}, в которой необходимо подключить библиотеку ввода/вывода stdio.h.

5.13. В секции правил Yacc-файлов должны быть приведены описания продукций приведения нетерминалов грамматики БП.

5.14. Каждая продукция секции правил Yacc-файлов 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.Например:

```
s: p '\n';
```

5.15. Продукции секции правил Yacc-файлов, приведение нетерминалов которых необходимо сопровождать функциональной обработкой, должны содержать блоки действий. Блоки действий должны располагаться в правых частях продукций и ограничиваться парой фигурных скобок. Внутри блоков действий можно использовать любые конструкции и вызовы функций системы программирования C. Например, правило определения начального нетерминала s целесообразно дополнить блоком действий, который возвращает успешное выполнение разбора:

```
s: p  '\n' { return 0; };
```

5.16. Продукции секции правил Yacc-файлов, 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 (|).

5.17. Секция функций Yacc-файлов должна содержать исходный код, оформленный по правилам системы программирования C, для функции yylex() которая должна иметь целочисленный код возврата.

5.18. Для достижения целей лексического анализа, указанных в п. 5.17, исходный код функции yylex() должен предусматривать чтение и возврат очередного символа;

5.19. Исходный код секции функций Yacc-файлов образует исходный код программ BINSET, который должен формироваться генератором

синтаксических анализаторов YACC в файлах с предопределенными именами binsetL.c и binsetR.c в текущем рабочем каталоге файловой системы OS UNIX. Выполняемые модули программ BINSET должны быть созданы по файлам исходного кода в выполняемых файлах binsetL и binsetR средствами компилирующей системы программирования C.

Стадии и этапы разработки

Процесс разработки программы BINSET целесообразно разделяться на следующие 3 стадии:

- составить YACC-файлы binsetL.y и binsetR.y в выбранном рабочем каталоге файловой системы OS UNIX, используя любой текстовый редактор;
- получить исходный код синтаксического анализатора в файлах binsetL.y и binsetR.y текущего каталога файловой системы OS UNIX, обработав YACC-файл binset.y командой yacc, следующим образом:

```
$ yacc binsetL.y > binsetL.c
$ yacc binsetR.y > binsetR.c
```

- сформировать выполняемый модуль в файлах binsetL.y и binsetR.y текущего каталога файловой системы OS UNIX, компилируя исходный код синтаксического анализатора следующей командой:

```
$ cc –o binsetL binsetL.c –lfl
$ cc –o binsetR binsetR.c –lfl
```

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

Порядок контроля и приемки

1. Для проверки функционирования программы BINSET должны быть предложены контрольные примеры, предусматривающие стандартный ввод корректных и некорректных.

2. Для приемки программы BINSET должен быть организован вызов выполняемых файлов binsetL.y и binsetR.y в консольном режиме работы OS UNIX или режиме эмуляции терминала операционной среды X Window System.

Результаты разработки

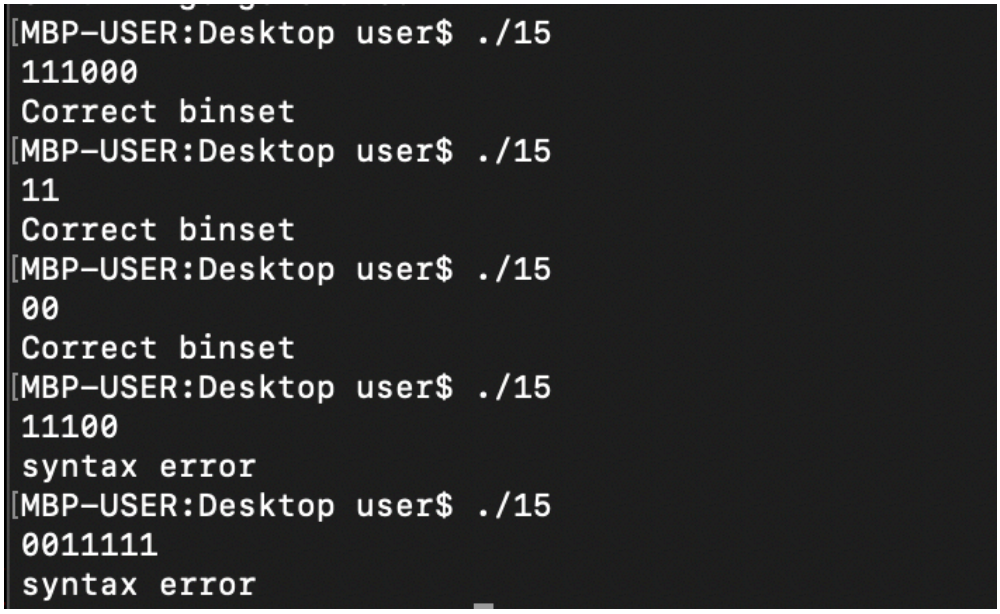
Алгоритм синтаксического анализатора

Программа BINSET выполняет построение конечного автомата на основе правил регулярной грамматики. Далее при запуске программы, в течение её выполнения, происходит процесс считывания информации из входного потока и последующий вывод требуемых двоичных наборов в случае их обнаружения в строке.

Описание грамматики

- Терминальный алфавит: Σ = {‘0’, ’1’, ’\n’}
- Нетерминальный алфавит: {p, q, s};
- Правила для левостороннего вывода:
    1. s → p '\n'
    2. p → q '1'
    3. p → q '0'
    4. q → p '1'
    5. q → p '0'
    6. q → '1'
    7. q → '0'
- Правила для правостороннего вывода:
    1. s → p '\n'
    2. p → '1' q
    3. p → '0' q
    4. q → '1' p
    5. q → '0' p
    6. q → '1'
    7. q → '0'

Результаты синтаксического анализа



Содержимое файла спецификации с левосторонним выводом

```
%{
#include<stdio.h>
%}
%%
S:p'\n' {return (0);};

p: q '1'
 |q '0'
 ;
q:p '1'
 |p '0'
 |'0'
 |'1'
 ;

%%
int yylex()
{
return(getchar());
}
int yyerror(char *s)
{
printf("%s\n",s);
return(1);
}
int main(int argc, char *argv[])
{
int ret;
if((ret=yyparse())==0)
puts("Correct binset");
return (ret);
}
```

Содержимое файла спецификации с правосторонним выводом

```
%{
#include<stdio.h>
%}
%%
S:p'\n' {return (0);};

p: '1'q
 |'0' q
 ;
q:'1' p
```

```
 |'0' p
 |'0'
 |'1'
 ;

%%
int yylex()
{
return(getchar());
}
int yyerror(char *s)
{
printf("%s\n",s);
return(1);
}
int main(int argc, char *argv[])
{
int ret;
if((ret=yyparse())==0)
puts("Correct binset");
return (ret);
}
```

Список использованных источников

1. Родионов С.В., Волосатова Т.М. “Автоматизация проектирования синтаксических анализаторов”
2. Тихомиров В.П., Давидов М.И. Операционная система ДЕМОС: инструментальные средства программирования, М.: Финансы и статистика, 1988
3. База и Генератор Образовательных Ресурсов (bigor.bmstu.ru)
4. Генератор программ синтаксического анализа YACC. Производственно-внедренческий кооператив "И Н Т Е Р Ф Е Й С": Демос/Р 2.1, 1988