Отчёт по лабораторной работе № 11 по курсу «Архитектура компьютера и информационных систем»

Студент группы М8О-112Б-22

Епифанов Евгений Валерьевич, № по списку: 9

# E-mail: [epi\_abitur](mailto:epi_abitur)@mail.ru

# Телефон: 8(916)783-95-69

# Работа выполнена: 14 ноября 2022 г.

Преподаватель: Никулин Сергей Петрович, каф.806

**1. Тема:** Обработка последовательности литер входного текстового файла. Простейшие приёмы лексического анализа. Диаграммы состояний и переходов.

**2. Цель работы:** Составить программу на языке Си, выполняющую анализ и обработку вводимого текста в соответствии с выданным преподавателем вариантом задания.

Изображение выглядит как текст

Автоматически созданное описание

**3. Задание:** составить программу на Си для решения полученной задачи (Вариант 39: “Подсчитать количество шестнадцатеричных слов в строке, изображающих отрицательные числа в 64-битной архитектуре.”).

Изображение выглядит как текст

Автоматически созданное описание

**4. Оборудование (ПЭВМ студента):**

Ноутбук с процессором Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz, 1992 МГц, ядер: 4, логических процессоров: 8, ОП 8 ГБ. SSD 256 ГБ.

**5. Программное обеспечение ЭВМ студента:**

* Операционная система семейства UNIX, наименование: Linux Ubuntu (22.04) 5.15.0-48-generic #54-Ubuntu SMP Fri Aug 26 13:26:29 UTC 2022 x86\_64 x86\_64 x86\_64 GNU/Linux.
* Интерпретатор команд: GNU bash, version 5.1.16(1)-release (x86\_64-pc-linux-gnu).
* Компилятор gcc Linux.
* Редактор текстов: GNU Emacs 28.2
* Утилиты и прочие программы.

**6. Идея, метод, алгоритм решения задачи:**

1. Взаимодействие с программой.
   1. emacs \_\_\_.c – запуск emacs с последующим вводом кода.
   2. cc \_\_.c – компиляции программы.
   3. ./a.out – тестирование программы.
2. Программа:

#include <stdio.h>

#include <ctype.h>

int main()

{

char c;

int sost = 0, otv = 0, kol = 0;

while (sost != 3)

{

c = getchar();

switch (sost)

{

case 0:

if (c == EOF) {sost = 3; break;}

if (c != ' ' && c != ',' && c != '\t' && c != '\n' && c != EOF) sost = 1;

case 1:

if (c == EOF) {sost = 3; break;}

if (c == ' ' || c == ',' || c == '\t' || c == '\n') sost = 0;

if (c == '8' || c == '9' || c == 'A' || c == 'B' || c == 'C' || c == 'D' || c == 'E' || c == 'F') sost = 2; kol = 1;

case 2:

if (c == EOF) {sost = 3; break;}

if (!(isdigit(c)) && c != 'A' && c != 'B' && c != 'C' && c != 'D' && c != 'E' && c != 'F' && kol == 17) {sost = 1; kol = 0; otv += 1;}

if (!(isdigit(c)) && c != 'A' && c != 'B' && c != 'C' && c != 'D' && c != 'E' && c != 'F') {sost = 1; kol = 0;}

else kol += 1;

}

}

printf("\n%s%i\n", "V texte podhodyashih chisel: ", otv);

return 0;

}

Данная программа получает на вход некоторый текст. Программа посимвольно проверяет вводимые данные и определяет, вводятся ли сейчас данные, необходимые для получения ответа. В программе присутствует разделение на состояния (switch case). Программа прекратит работу, когда будет нажата комбинация клавиш: Ctrl + D.

1. Тесты для программы:
2. Ввод:

i like unix!

01100101100001

Вывод:

V texte podhodyashih chisel: 0

1. Ввод:

chislo 123456789ABCDE12 dast noll

Вывод:

V texte podhodyashih chisel: 0

1. Ввод:

odno iz chisel vernoe: A962537BCA66011A - 4FABC87878700AA4D

Вывод:

V texte podhodyashih chisel: 1

1. Ввод:

one imposter among us...

A962537BCA66011B A962537BCA66011C A962537BCA66011F

A962537BCA66011Y A962537BCA660114 A962537BCA660110

Вывод:

V texte podhodyashih chisel: 5

1. Ввод:

DAD1ADD1DEF2FEDA

Вывод:

V texte podhodyashih chisel: 1

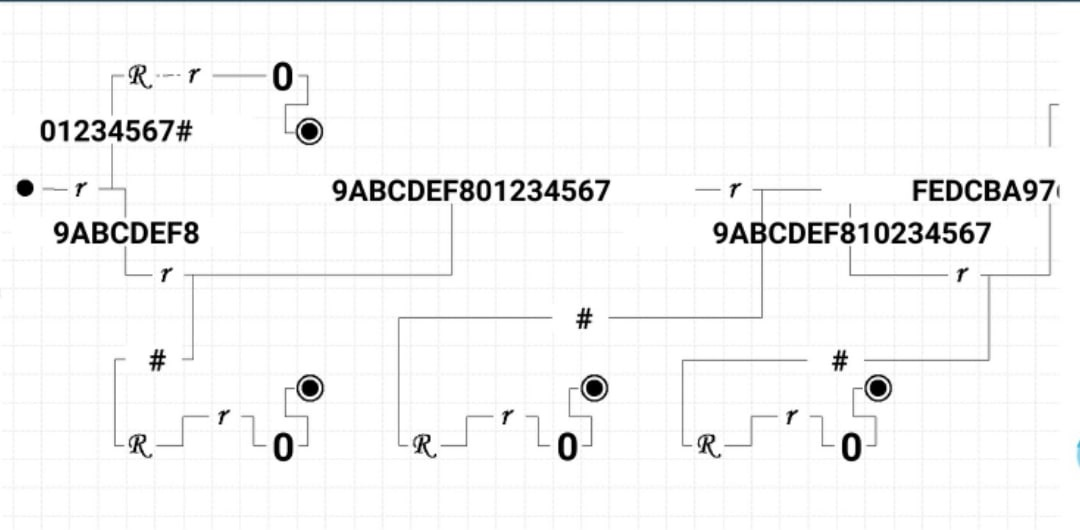
1. Ввод:

C68363627AA12345 - 1 = C68363627AA12344

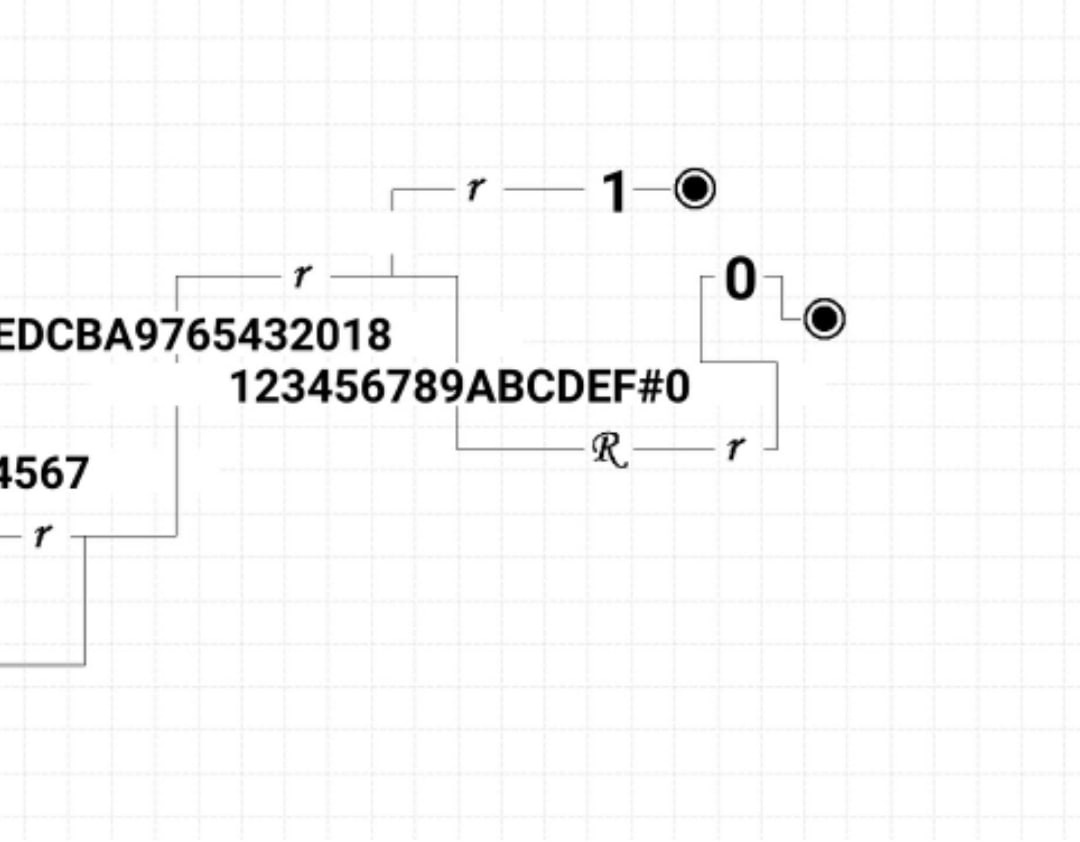
Вывод:

V texte podhodyashih chisel: 2

1. Схема:



# • • •



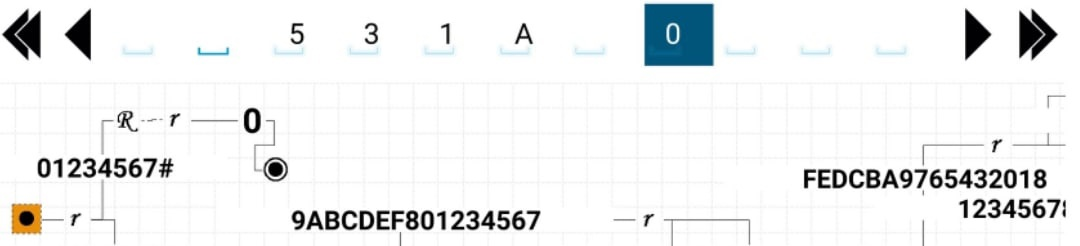
На данной схеме представлен алгоритм определения, является ли наше число частью ответа. На вход подается какое-то шестнадцатеричное число, алгоритм проверяет, что первая цифра числа принадлежит необходимому диапазону (8 – F (в двоичной записи только эти символы начинаются с 1)) и продолжает проверку, иначе программа ставит в конце 0, символизирующий то, что число нам не подходит. Далее идет 15 обычных проверок на то, что каждый из 15 символов также принадлежит азбуке 16-ричной СС (на фотографиях выше представлены начало и конец схемы, так как средняя часть полностью повторяет то, что изображено в правой части 1 фотографии).

Последняя проверка отслеживает длину числа: если в числе больше 16 знаков – ставим 0 в конце. Если все проверки будут успешно пройдены, то в конце будет напечатана 1. Символ ‘#’ обозначает любой символ не из алфавита 16-ричной СС, с помощью него отслеживаем, что используются только нужные нам символы.

Примеры:

1. Число не прошло проверку.





1. Число прошло проверку.



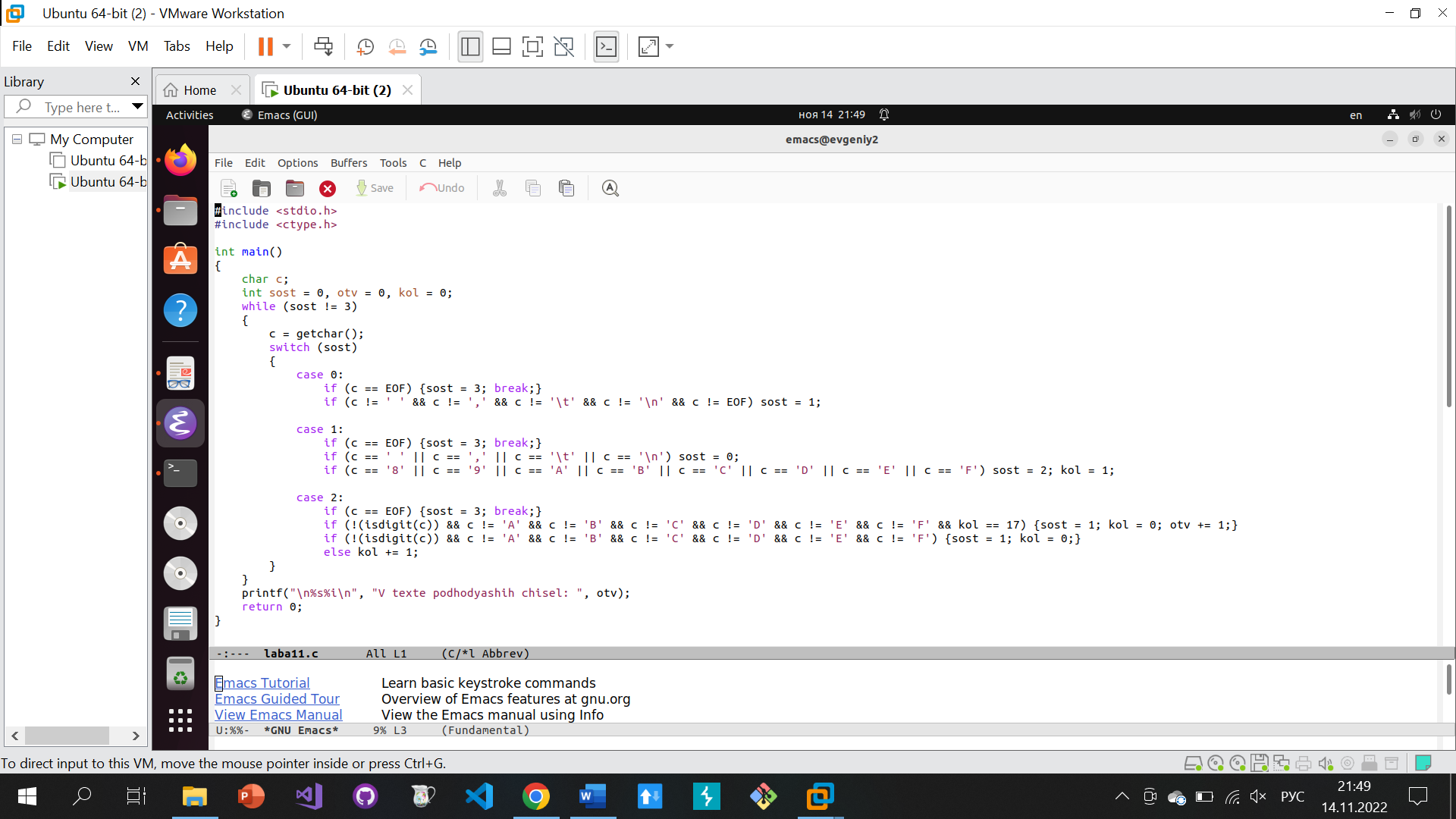


**7. Сценарий выполнения работы:**

1. Проанализировать полученную задачу и разработать для её решения корректно функционирующую программу.
2. Написать программу в текстовом редакторе Emacs.
3. Скомпилировать программу и проверить её работоспособность на заранее заготовленных тестах.

**8. Распечатка протокола:**

evgeniy2@evgeniy2:~$ emacs laba11.c



evgeniy2@evgeniy2:~$ cc laba11.c

evgeniy2@evgeniy2:~$ ./a.out

i like unix!

01100101100001

V texte podhodyashih chisel: 0

evgeniy2@evgeniy2:~$ ./a.out

chislo 123456789ABCDE12 dast noll

V texte podhodyashih chisel: 0

evgeniy2@evgeniy2:~$ ./a.out

odno iz chisel vernoe: A962537BCA66011A - 4FABC87878700AA4D

V texte podhodyashih chisel: 1

evgeniy2@evgeniy2:~$ ./a.out

one imposter among us...

A962537BCA66011B A962537BCA66011C A962537BCA66011F

A962537BCA66011Y A962537BCA660114 A962537BCA660110

V texte podhodyashih chisel: 5

evgeniy2@evgeniy2:~$ ./a.out

DAD1ADD1DEF2FEDA

V texte podhodyashih chisel: 1

evgeniy2@evgeniy2:~$ ./a.out

C68363627AA12345 - 1 = C68363627AA12344

V texte podhodyashih chisel: 2

evgeniy2@evgeniy2:~$

9. Дневник отладки:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
|  |  |  |  |  |  |  |

10. Замечания по существу работы: -

11. Выводы: Я научился решать задачи, связанные с обработкой текста, на языке программирования Си. Я усовершенствовал свои навыки программирования на Си в Linux.

Недочёты при выполнении задания могут быть устранены следующим образом: -