**Отчет по лабораторной работе № 11 по курсу «Фундаментальная информатика»**

Студент группы М8О-109Б-22 Концебалов Олег Сергеевич

Контакты: telegram @baronpipistron

Работа выполнена: 05.12.2022

Преподаватель: каф.806 Сысоев Максим Алексеевич

Отчет сдан «12» декабря 2022 г., итоговая оценка \_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

**2. Цель работы:** Написать программу на языке С, которая будет выполнять действия, указанные в задании

**3. Задание (вариант № 17):** Выделить все числа, записанные в троичной системе счисления и кратные 3

**4. Оборудование (студента):**

Процессор AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHz, ОП 16,0 Гб, SSD 512 Гб. Монитор 1920x1080 144 Hz

**5. Программное обеспечение (студента):**

Операционная система семейства Linux, наименование Ubuntu, версия 18.10

Интерпретатор команд: bash, версия 4.4.19

Система программирования – версия --, редактор текстов Emacs, версия 25.2.2

Утилиты операционной системы –

Прикладные системы и программы –

Местонахождение и имена файлов программ и данных на домашнем компьютере –

**6. Идея, метод, алгоритм решения задачи** *(в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)*

Идея заключается в том, что считываемые из входного файла с помощью метода getchar символы преобразуются в их ASCII код и дальнейшая работа с ними будет осуществляться как с числом. Выполняем действия согласно разработанному конечному автомату (прикреплен в п.8). Вывод чисел реализован с помощью двух функций, число, равное ASCII-коду определенного символа, преобразуется в этот символ и записывается в массив, элементы которого позже выводятся.

Идея проверки делимости на 3 в троичной системе – число должно содержать только цифры 0 1 2 и оканчиваться на 0

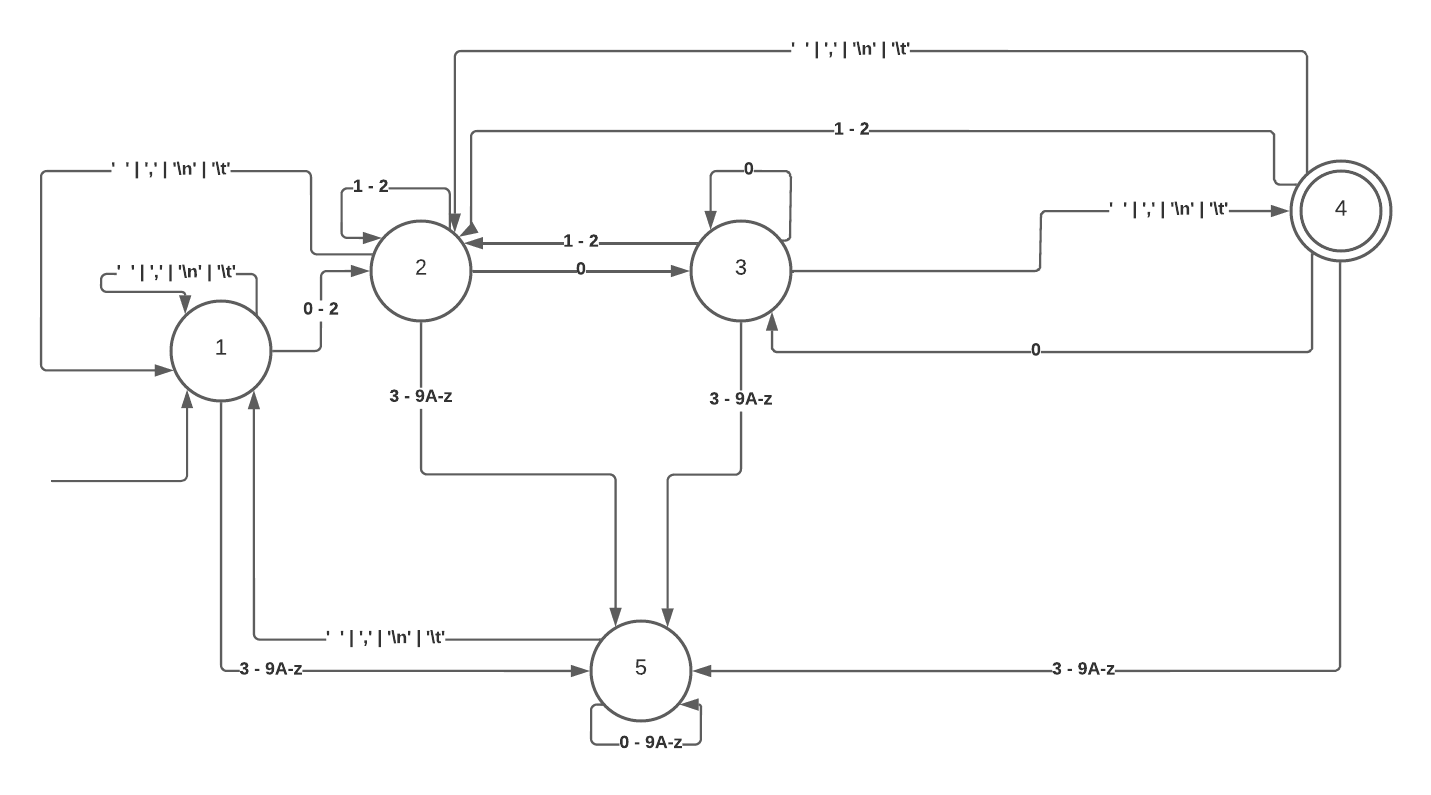
**7. Сценарий выполнения работы** *(план работы, первоначальный текст программы в черновике [можно на отдельном листе] и тесты, либо соображения по тестированию)*

1. Составляю конечный автомат
2. Реализую его на С
3. Пишу функции и алгоритм для вывода чисел
4. Тестирую конечную программу

|  |  |  |
| --- | --- | --- |
| Входные данные | Выходные данные | Описание тестируемого случая |
| \* | Suitable numbers: 0 | На вход подается пустая строка |
| 1452 120 15545154 154542 20 11851 1121210 1515 \* | 120 20 1121210  Suitable numbers: 3 | Начало с неподходящего числа |
| 120 1515422254 8415484 5152451 111 11511 515844 5151120 121512 1841 5120 15151515 20 115151 12121212 10 12 \* | 120 20 10  Suitable numbers: 3 | Начало с подходящего числа |
| 1 215 5 545454 415252154845 51544 45454844 664545 4845152 151545 15154545 4545 x | Suitable numbers: 0 | Нет подходящих чисел |
| 120 2210 112120 1201120 111210 120020 210 \* | 120 2210 112120 1201120 111210 120020 210  Suitable numbers: 7 | Все числа подходящие |
| 12afc10 dqf 10 \* | 10  Suitable numbers: 1 | Вперемешку с буквами |

**8. Распечатка протокола** *(подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем)*

*Конечный автомат:*



*Код программы:*

#include <stdio.h>

#include <assert.h>

char ascii\_to\_char(int symbol){

if (symbol == 48){

return '0';

}

else if (symbol == 49){

return '1';

}

else if (symbol == 50){

return '2';

}

}

void print\_char(int count, char array[50]){

for (int k = 0; k < count; ++k){

printf("%c", array[k]);

}

}

void test\_ascii\_to\_char(){

assert(ascii\_to\_char(48) == '0');

assert(ascii\_to\_char(49) == '1');

assert(ascii\_to\_char(50) == '2');

}

int main(){

test\_ascii\_to\_char();

int state = 1, count = 0, k = 0;

int symbol;

while((symbol = getchar()) != '\*'){

char array[50];

switch (state) {

case 1:

if (symbol >= '0' && symbol <= '2'){

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 2;

}

else if ((symbol >= '3' && symbol <= '9') || (symbol >= 'A' && symbol <= 'Z') || (symbol >= 'a' && symbol <= 'z')){

state = 5;

}

else if (symbol == ' ' || symbol == ',' || symbol == '\n' || symbol == '\t'){

count = 0;

state = 1;

}

break;

case 2:

if (symbol == '0'){

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 3;

}

else if (symbol >= '1' && symbol <= '2'){

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 2;

}

else if ((symbol >= '3' && symbol <= '9') || (symbol >= 'A' && symbol <= 'Z') || (symbol >= 'a' && symbol <= 'z')){

state = 5;

}

else if (symbol == ' ' || symbol == ',' || symbol == '\n' || symbol == '\t'){

if (count == 1 && array[0] == '0'){

printf("%c ", array[0]);

k += 1;

}

count = 0;

state = 1;

}

break;

case 3:

if (symbol == '0'){

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 3;

}

else if (symbol >= '1' && symbol <= '2'){

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 2;

}

else if ((symbol >= '3' && symbol <= '9') || (symbol >= 'A' && symbol <= 'Z') || (symbol >= 'a' && symbol <= 'z')){

state = 5;

}

else if (symbol == ' ' || symbol == ',' || symbol == '\n' || symbol == '\t'){

k += 1;

print\_char(count, array);

printf("%c", ' ');

state = 4;

}

break;

case 4:

if (symbol == '0'){

count = 0;

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 3;

}

else if (symbol >= '1' && symbol <= '2'){

count = 0;

array[count] = ascii\_to\_char(symbol);

count += 1;

state = 2;

}

else if ((symbol >= '3' && symbol <= '9') || (symbol >= 'A' && symbol <= 'Z') || (symbol >= 'a' && symbol <= 'z')){

count = 0;

state = 5;

}

else if (symbol == ' ' || symbol == ',' || symbol == '\n' || symbol == '\t'){

count = 0;

state = 1;

}

break;

case 5:

if ((symbol >= '0' && symbol <= '9') || (symbol >= 'A' && symbol <= 'Z') || (symbol >= 'a' && symbol <= 'z')){

count = 0;

state = 5;

}

else if (symbol == ' ' || symbol == ',' || symbol == '\n' || symbol == '\t'){

count = 0;

state = 1;

}

break;

default:

break;

}

}

printf("\nSuitable numbers: %d", k);

return 0;

}

**9. Дневник отладки** *(дата и время сеансов отладки и основные события [ошибки в сценарии и программе, нестандартные ситуации] и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы)*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *№* | *Лаб. или дом* | *Дата* | *Время* | *Событие* | *Действие по исправлению* | *Примечания* |
| 1 | дом | 4.12.22 | 21:34 | Выводит только первое подходящее число | Ошибся в конечном автомате, пришлось переделать |  |
| 2 | дом | 4.12.22 | 22:15 | Нет идей как реализовать вывод | Никаких |  |
| 3 | дом | 5.12.22 | 19:32 | Выводит подходящие числа только до первого неподходящего | Прописал функции вывода не в том case, быстро исправилось |  |

*Примечание: для завершения ввода необходимо напечатать символ \*. Данный костыль связан с тем, что стандартное сочетание для EOF ctrl+D у меня не работает. Посовещавшись с Иваном Фоминым, пришли к выводу, что это скорее всего проблема CLion’а*

**10. Замечания автора** *(по существу работы)*

Замечания отсутствуют

**11. Вывод**

Данная лабораторная работа очень понравилась, выполнять ее было в разы приятнее, чем 10. Составить конечный автомат, как и написать основной код программы, оказалось не очень сложно. Трудности возникли с реализацией вывода подходящих чисел – сначала вообще не было никаких идей. Пробовал реализовать через суммы, но так как работал с ASCII кодом, то ничего хорошего не получилось. В результате придумал сделать две функции: одна преобразует ASCII код в символ, а другая выводит конечный результат – подходящее число. Понравился момент, что в конце пришлось подумать над выводом.

В целом нахожу эту лабораторную довольно полезной, потому что она учит работать с входящим текстом и регулярными выражениями, что пригодится в будущей работе

Работа на 10/10

Подпись студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_