Отчет по лабораторной работе № 11 по курсу по курсу “Фундаментальная информатика”

Студент группы **М8О-107Б-20** Чекменев Вячеслав Алексеевич, № по списку 25

Контакты e-mail: chekmenev031@gmail.com, telegram: @suraba03

Работа выполнена: «13» декабря 2020 г.

Преподаватель: каф. 806 Найденов Иван Евгеньевич

Отчет сдан « » _____20 ___ г., итоговая оценка _____

Подпись преподавателя _____

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

2. **Цель работы:** Составить программу на языке С, используя конечные автоматы

3. **Задание** ()**:** вариант 13

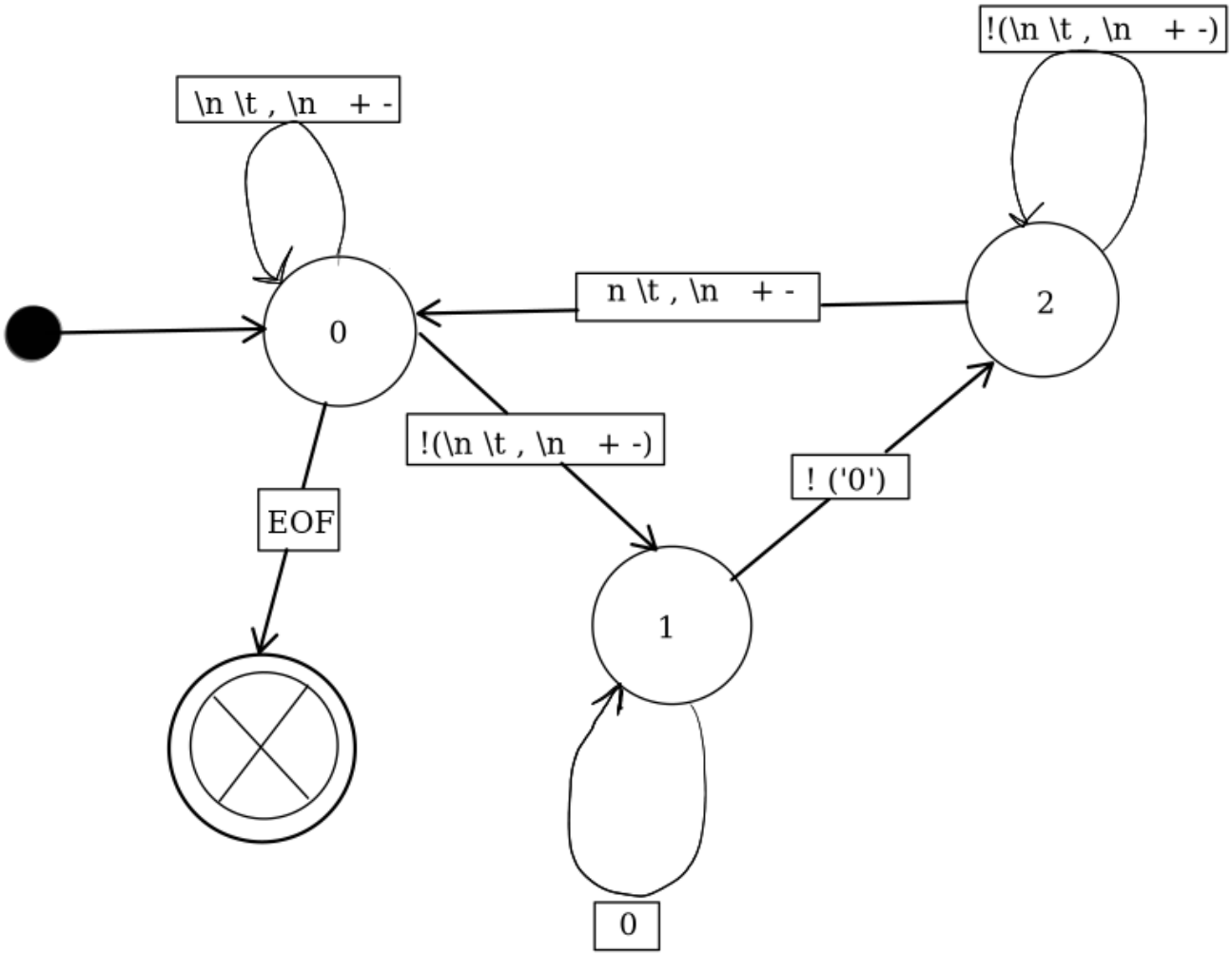
4. **Оборудование** (студента):

   Процессор *Intel Core i5-8265U* с ОП 7851 Мб, НМД 256 Гб. Монитор *1920x1080*

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

   Операционная система семейства UNIX: linux, наименование: manjaro, версия: 20.1 Mikah
   интерпретатор команд: bash, версия: 5.0.18.
   текствый редактор: code - oss
   Утилиты операционной системы: –
   Прикладные системы и программы: GNOME terminal
   Местонахождение и имена файлов программ и данных –

6. Идея, метод, алгоритм



7. Сценарий выполнения работы

1) написать функции `is_split_simb,` is_hex_symb, hex_simb_to_int

2) в функции main: находиться в STATE0 пока не закончатся ведущие нули
3) перейти в STATE1, там сосчитать нули
4) В STATE2 вывести все в нужной последовательности

Тесты:

Входные данные	**Выходные данные**	**Описание тестируемого случая**
00123ab 00AB123	123ab00 AB12300	Проверка с ведущими нулями, 16ричное число
00123afb 00AFB123	00123afb 00AFB123	Проверка с ведущими нулями, не 16ричное число
+00123ab -00AB123	+123ab00 - AB12300	Проверка с ведущими нулями, 16ричное число, со знаком
+0gdhdhsj -kwodbb	+0gdhdhsj -kwodbb	Проверка с ведущими нулями, не 16ричное число, со знаком

8. Распечатка протокола

```
#include <stdio.h>

typedef enum {
   STATE0,
   STATE1,
   STATE2
} State;

int is_split_simb(int c)
{
   return c == ' ' || c == '\t' || c == ',' || c == '\n' || c == '+' || c == '-';
}

int is_hex_symb(int c)
{
   return (c <= 'F' && c >= 'A') || (c <= 'f' && c >= 'a') || (c <= '9' && c >= '0');
}


int hex_simb_to_int(char x) // дыоаптдып
{
   int n_10 = 0;
   if (x <= 'F' && x >= 'A') {
      n_10 = x - 55;
      return n_10;
   } else if (x <= 'f' && x >= 'a') {
      n_10 = x - 87;
      return n_10;
   } else {
      n_10 = x - 48;
      return n_10;
   }
}

int main(void)
{
   State st = STATE0;
   char c;
   int cnt_0 = 0, num_decim = 0, check_size = 0;
   while ((c = getchar()) != EOF) {
      switch (st) {
         case STATE0:
            if (is_split_simb(c)) {
               printf("%c", c);
            } else {
               if (is_hex_symb(c) && c != '0') {
                  if (c <= 'f' && c >= 'a') {
                     check_size = 0;
                  } else if (c <= 'F' && c >= 'A') {
                     check_size = 1;
                  }
```

```
            num_decim *= 16;
            num_decim += hex_simb_to_int(c);
            st = STATE2;
            break;
        } else if (c == '0') {
            cnt_0++;
        } else {
            printf("%c", c);
        }
        st = STATE1;
    }
    break;
case STATE1:
    if (c == '0') {
        cnt_0++;
    } else {
        st = STATE2;
    }
case STATE2:
    if (is_hex_symb(c)) {
        if (c <= 'f' && c >= 'a') {
            check_size = 0;
        } else if (c <= 'F' && c >= 'A') {
            check_size = 1;
        }
        num_decim *= 16;
        num_decim += hex_simb_to_int(c);
    } else if (is_split_simb(c)) {
        if (num_decim != 0) {
            if (check_size) {
                printf("%X", num_decim);
            } else {
                printf("%x", num_decim);
            }
        }
        if (cnt_0 != 0) {
            for (int i = 0; i < cnt_0; i++) {
                printf("0");
            }
        }
        printf("%c", c);
        cnt_0 = 0;
        num_decim = 0;
        st = STATE0;
    } else {
        if (cnt_0 != 0) {
            for (int i = 0; i < cnt_0; i++) {
                printf("0");
            }
        }
        if (num_decim != 0) {
            if (check_size) {
                printf("%X", num_decim);
```

```
                } else {
                    printf("%x", num_decim);
                }
            }
            printf("%c", c);
            cnt_0 = 0;
            num_decim = 0;
        }
    }
  }
  return 0;
}
```

9. Дневник отладки должен содержать дату 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	Примечание
11	дом	13 декабря	19:00	Перепробовал все тесты, на 199 тесте ошибка	сдать лабу, там узнать ошибку	

10. Замечания автора по существу работы

Вроде все хорошо, но не могу никак пройти все тесты на чекере

11. Выводы

Работа оказалась интресной, много пришлось исправлять код, чтобы пройти все тесты.

Подпись студента: