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

Студент группы М80-109Б-22 Серый Никита Олегович, № по списку 16

Контакты:
email — nikita.seryj@mail.ru
telegram - @hukumkas

Работа выполнена: «18» декабря 2022г.

Преподаватель: каф. 806 Сысоев Максим Алексеевич

Отчет сдан «19» _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:** Составить программу на языке Си, выполняющую анализ и обработку вводимого текста в соответствии с выданным преподавателем вариантом задания.

3. **Задание** (*вариант №* **35** **):**
   Выделить все десятичные числа от 17 до 77 по модулю и распечатать их числа в словесной форме по-французски.

4. **Оборудование** (студента):

   Процессор *AMD Ryzen 5 5500U @ 6x 2.1GH* с ОП *15360* Мб, НМД (?) *512* Гб. Монитор *1920x1080*

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

   Операционная система семейства: *linux*, наименование: *mint*, версия *21 Cinnamon*
   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
Программа считывает знаки, пока не встречает EOF – конец входного потока. Она проверяет, удовлетворяет ли строка следующим условиям: **десятичная** система счисления; модуль числа (игнорирует `+` и `-`перед числом), игнорирует пробел, наличие запятой, символ новой строки, символ табуляции. Программа уточняет, находится ли число в границах от 17 до 77 (лежит ли число десятков в границах от 1 до 7 включительно, а число единиц — в границах от 0 до 9 включительно). В случае корректности всех данных, она переходит между состояниями, что отражает работу конечного автомата. Это почти как в лифте: переходит на другой этаж только если двери закрыты. Также и у нас: сначала считываем строку, проверяем его на соблюдение первичных пунктов
(похожа ли строка на цифру), затем идём к условиям соблюдения границ.

Как работает конечный автомат:

s0:

Если выполняется одна из функций, идём к s1

s1:

1) Если 'c' != одной из границ, то число десятков = 'a', идём к s2 (из s1)

2) Если 'c' = одной из границ десятков, идём к s0

s2:

1) Если (a != 1 и c >= '7') и (a != 7 и c <= '7'), идём в s3

2) Иначе идём в s0

s3:

1) Если выполняется одна из функций, идём к s4, объявляем numbers

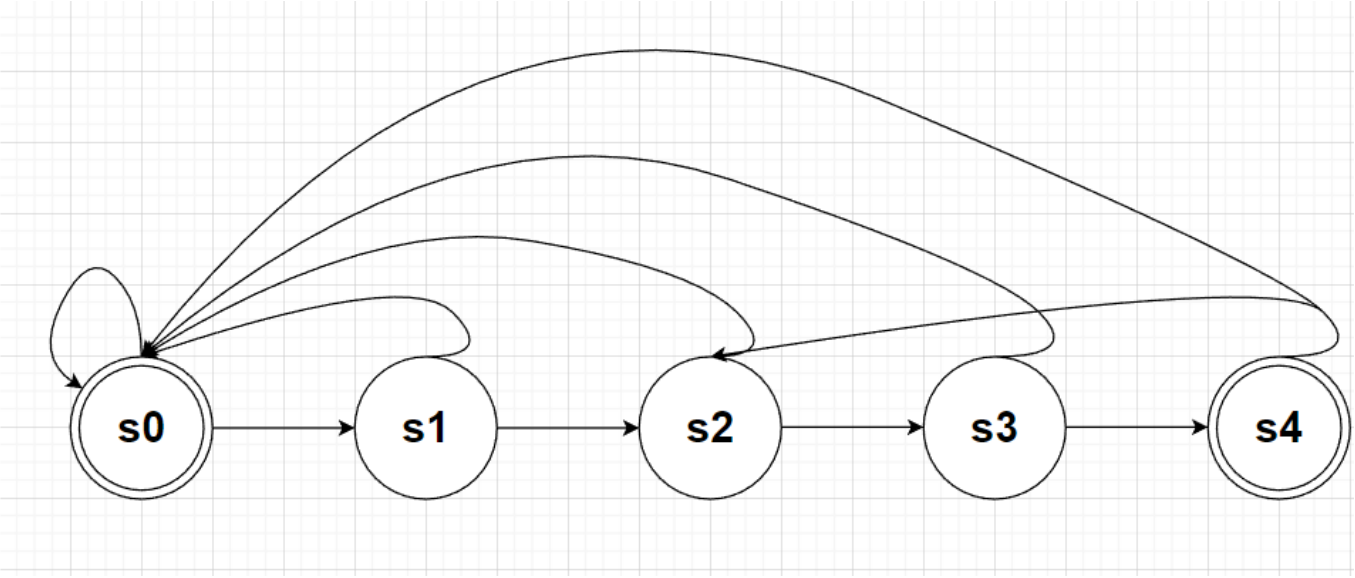
2) Иначе идём к s0

s4:

1) Если 'c' != одной из границ десятков, идём к s2

2) Если 'c' = одной из границ десятков, идём к s0

3) Если выполняется одна из функций, индикатор = '0', идём к s1. Иначе индикатор = 0, идём к s0



7. Сценарий выполнения работы

Входные данные	**Выходные данные**	**Описание тестируемого случая**
e77	нет	Проверка на (соблюдено)
+12	нет	Проверка вхождения в границы (соблюдено)
,17,	dix-sept	Проверка на влияние разделителя (всё верно)
-17	dix-sept	Проверка на вывод модуля числа (всё верно)

8. Распечатка протокола

```
#include <stdio.h>
#include <assert.h>
typedef enum{
 s0,
 s1,
 s2,
 s3,
 s4
}status;
int check_emptiness(char c){
  if (c == ' ')
    return 1;
  else
    return 0;
}
void test_check_emptiness(){
    char c=' ';
    assert(check_emptiness(c)==1);
    c='r';
    assert(check_emptiness(c)==0);
    c=EOF;
    assert(check_emptiness(c)==0);
}
int check_comma(char c){
   if ( c == ',')
    return 1;
   else
    return 0;
}
void test_check_comma(){
   char c=',';
    assert(check_comma(c)==1);
    c='r';
    assert(check_comma(c)==0);
    c=EOF;
    assert(check_comma(c)==0);
}
int check_tabulation(char c){
   if (c == '\t')
    return 1;
   else
    return 0;
}
void test_check_tabulation(){
    char c='\t';
    assert(check_tabulation(c)==1);
    c='r';
    assert(check_tabulation(c)==0);
    c=EOF;
    assert(check_tabulation(c)==0);
}
int check_new_string(char c){
   if (c == '\n')
    return 1;
```

```
   else
    return 0;
}
void test_check_new_string(){
    char c='\n';
    assert(check_new_string(c)==1);
    c='\t';
    assert(check_new_string(c)==0);
    c=EOF;
    assert(check_new_string(c)==0);
}
int check_num_10(char c ){
   if(c >= '0'&&c <= '9')
    return 1;
   else
    return 0;
}
int check_sign(char c){
    if(c == '+' || c == '-')
        return 1;
    else
        return 0;
}
void test_check_num_10(){
    char c='0';
    assert(check_num_10(c)==1);
     c='1';
    assert(check_num_10(c)==1);
     c='2';
    assert(check_num_10(c)==1);
     c='3';
    assert(check_num_10(c)==1);
     c='4';
    assert(check_num_10(c)==1);
     c='5';
    assert(check_num_10(c)==1);
     c='6';
    assert(check_num_10(c)==1);
     c='7';
    assert(check_num_10(c)==1);
     c='8';
    assert(check_num_10(c)==1);
    c='9';
    assert(check_num_10(c)==1);
    c='\t';
    assert(check_num_10(c)==0);
    c=EOF;
    assert(check_num_10(c)==0);
}
void test_check_sign(){
    char c='+';
    assert(check_sign(c)==1);
     c='-';
    assert(check_sign(c)==1);
     c='r';
    assert(check_sign(c)==0);
```

```
    c=EOF;
   assert(check_sign(c)==0);
}
int numbers (char a, char b){
   if( check_num_10(a) && check_num_10(b) ){
   switch(a){
      case '1':
         printf("dix");
      break;
      case '2':
         printf("vingt");
      break;
      case '3':
         printf("trente");
      break;
      case '4':
         printf("quarante");
      break;
      case '5':
         printf("cinquante");
      break;
      case '6':
         printf("soixante");
      break;
      case '7':
         printf("soixante-");
      break;
      default:
         return 0;
      break;
   }

   switch(b){
      case '0':
         if (a == '7')
            printf("dix\n");
         else
            printf("\n");
      break;
      case '1':
         if (a == '7')
            printf("et-onze\n");
         else
            printf(" et un\n");
      break;
      case '2':
         if (a == '7')
            printf("douze\n");
         else
            printf("-deux\n");
      break;
      case '3':
         if (a == '7')
            printf("treize'\n");
         else
            printf("-trois'\n");
```

17	dix-sept
18	dix-huit
19	dix-neuf
20	VINGT

21 vingt et un
22 vingt-deux
23 vingt-trois
24 vingt-quatre
25 vingt-cinq
26 vingt-six
27 vingt-sept
28 vingt-huit
29 vingt-neuf
30 TRENTE
31 trente et un
...
40 QUARANTE
41 quarante et un
...
50 CINQUANTE
51 cinquante et un
...
60 SOIXANTE
61 soixante et un
62 soixante-deux
63 soixante-trois
64 soixante-quatre
...
69 soixante-neuf

70 SOIXANTE-DIX
71 soixante-et-onze
72 soixante-douze
73 soixante-treize
74 soixante-quatorze
75 soixante-quinze
76 soixante-seize
77 soixante-dix-sept

```
        break;
        case '4':
            if (a == '7')
                printf("quatorze\n");
            else
                printf("-quatre\n");
        break;
        case '5':
            if (a == '7')
                printf("quinze\n");
            else
                printf("-cinq\n");
        break;
        case '6':
            if (a == '7')
                printf("seize\n");
            else
                printf("-six\n");
        break;
        case '7':
            if (a == '7')
                printf("dix-sept\n");
            else
                printf("-sept\n");
        break;
        case '8':
                printf("-huit\n");
        break;
        case '9':
                printf("-neuf\n");
        break;
        default:
            return 0;
        break;
    }

    return 1;
    }

    else{
        return 0;
    }
}

void test_numbers(){
    char a='4',b='0';
    assert(numbers(a,b)==1);
     a='r',b='8';
    assert(numbers(a,b)==0);
    a='7',b='3';
    assert(numbers(a,b)==1);
}

int main()
{   status var_status = s0;
    char c=' ', a=' ', b=' ';
```

```
    int indicator = 0;

    test_check_emptiness();
    test_check_comma();
    test_check_tabulation();
    test_check_num_10();
    test_check_new_string();
    test_check_sign();

    while( (c=getchar() ) != EOF ){ // пока не кончится входной поток, будем считывать знаки
        switch(var_status){
////////////////////////////////////////
        case s0:
         if ( check_emptiness(c) || check_new_string(c) || check_tabulation(c) || check_comma(c) ){
              var_status = s1;    // если выполняется одна из функций, идём к s1
         }
         break;
////////////////////////////////////////
        case s1:
         if ( check_num_10(c) && c != '0' && c != '8' ){
            a = c;   // если 'c' != одной из границ, то число десятков = 'a', идём к s2
            var_status = s2;
            indicator = 0;

         }
         else if ( check_num_10(c) && (c=='0' || c=='8') ){ // если 'c' = одной из границ десятков, идём к s0
            var_status = s0;
            indicator = 0;
         }
         else if ( check_sign(c) && indicator == 0 ){ // если проверяем знак и индикатор = '0", то флаг = '1'
            flag = 1;
         }
         else if ( check_sign(c) && indicator == 1 ){ // если проверяем знак и индикатор = '1', то идём к s0
            var_status = s0;
            indicator = 0;
         }
         else if ( check_emptiness(c) || check_new_string(c) || check_tabulation(c) || check_comma(c) ){
            indicator = 0; // если выполняется одна из функций, индикатор = '0'
         }
         else {
            var_status = s0; // иначе состояние s0, индикатор = '0'
            indicator = 0;
         }
         break;
////////////////////////////////////////
        case s2:
         if ( check_num_10(c) && !(a == '1' && c < '7') && !(a == '7' && c > '7') ){
              b = c; // если (a != 1 и c >= '7') и (a != 7 и c <= '7'), идём в s3
              var_status = s3;
         }
         else
            var_status = s0;
         break;
////////////////////////////////////////
        case s3:
         if ( check_emptiness(c) || check_new_string(c) || check_tabulation(c) || check_comma(c) ){
```

```
                var_status = s4;   // если выполняется одна из функций, идём к s4, объявляем numbers
                numbers(a,b);
                a=' ';
                b=' ';
            }
            else
                var_status = s0; // иначе s0
            break;
////////////////////////////////////////
           case s4:
            if( check_num_10(c) && c != '0' && c !='8' ){ //  если 'c' != одной из границ десятков, идём к s2
                a = c;
                var_status = s2;
                indicator = 0;

            }
            else if ( check_num_10(c) && (c == '0' || c == '8') ){
                var_status = s0; // если 'c' = одной из границ десятков, идём к s0
                indicator = 0;
            }
            else if ( check_sign(c) && flag == 0 ){
                indicator = 1;
                var_status = s1;
            }
            else if( check_sign(c) && flag == 1 ){
                var_status = s0;
                indicator = 0;
            }
            else if ( check_emptiness(c) || check_new_string(c) || check_tabulation(c) || check_comma(c) ){
                indicator = 0;  // если выполняется одна из функций, индикатор = '0', идём к s1
                var_status = s1;
            }
            else{
                indicator = 0; // иначе индикатор = 0, идём к s0
                var_status = s0;
            }
            break;
////////////////////////////////////////
           }
    }

    if (var_status == s3){ // если состояние s3, идём в s4
        numbers(a,b);
        var_status = s4;
    }
    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
1	дом	9.12.2022	04:20	Заметил, что есть исключение в ряде 70-77	Придумал, как его внедрить в код	Я доволен

10. Замечания автора по существу работы

Замечаний нет.

11. Выводы

В ходе работы я научился работе с конструкцией `switch-case`, познакомился с unit-тестированием, выполняемым при помощи библиотеки assert.h, понятием конца файла `EOF`, изучил конечные автоматы, и что самое главное — узнал, как пишутся числа на французском. Материал данной работы очень насыщенный и даёт прочный фундамент для выполнения будущих заданий.

Подпись студента

_____