# **Отчет по лабораторной работе № 13 по** курсу “Фундаментальная информатика”

Студент группы М80-109Б-22 Степанов Алексей Николаевич, № по списку 18

Контакты aleksey.stepanov2004@mail.ru, telegram @Alex1stepa

Работа выполнена: «23» ноября 2022г.

Преподаватель: каф. 806 Сысоев Максим Алексеевич

Отчет сдан «  » _____20__ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема: Множества**

2. **Цель работы:** В соответствии с вариантом задания составить программу проверки характеристик введенных последовательностей слов и печати развернутого ответа.
3. **Задание** (*вариант 7): Есть ли слова, начинающиеся и заканчивающиеся гласными?(буквы латинского алфавита)*
4. **Оборудование** (студента):
   Процессор *Intel Core i5-8265U @ 8x 3.9GH* с ОП *7851* Мб, НМД *1024* Гб. Монитор *1920x1080*

5. **Программное обеспечение (**студента**):**
   Операционная система семейства: *linux*, наименование: *ubuntu*, версия *18.10 cosmic*
   интерпретатор команд: *bash* версия *4.4.19*.
   Система программирования -- версия --**,** редактор текстов *emacs* версия *25.2.2*
   Утилиты операционной системы --
   Прикладные системы и программы: VTM(QT )
   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):
1. Составим множество, полностью состоящего из гласных букв латинского алфавита, а именно: 01000100000100000100010001, что соответствует десятичному числу 17842449. При конъюнкции этого числа с другим, полученный переводом его из char в множество, засунутое в int, результат отличный от 0( то есть не Ложь, ибо для лжи все биты должны быть нулевыми) будет означать, что данный символ был гласным.
2. Сымитируем конечный автомат, подобный тому, что был в 11 лабораторной, у которого будет 3 состояния:
2.1)0ое состояние, в котором он находится изначально и переходит после “переднего” разделителя. В нем он, при введении следующей литеры проверяет ее принадлежность к множеству гласных букв латинского алфавит( да и букв в целом), после чего переходит или в 1ое состояние, если это гласная буква, 2ое – если нет или не буква и остается в нулевом, если это разделитель.
2.2) 1ое состояние – в нем он запоминает предшествующий символ, пока не наткнется на 1 из разделителей. Если он считывает разделитель, то берет запомненный символ и проверяет, является ли тот гласной буквой, как и в 0 состоянии. Если он является, то мы нашли нужное слово=> PROFIT, иначе мы устанавливаем 0 состояние, чтобы сызнова перейти к поиску подходящего слова.
2.3) 2ое состояние – проходим любые символы, кроме разделителей. Достигнув разделитель, переходим в 0 состояние, чтобы сызнова перейти к поиску подходящего слова.

3. Достигнув EOF и проверив последнее слово при необходимости(1 состояние),выведем наш ответ, сохраненный в переменную solution как int, имитирующий булевский тип, в виде предложения на английском, информирующее о наличии или отсутствии требуемого слова в введенных строках.

7. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию]:
***"Проводилось unit-тестирование, исходный код тестов приложен в пункте №8"***
8. ***Распечатка протокола*** (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).
Код программы:

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <assert.h>
int64_t letter_or_not(char a){

  if(( (int64_t)a >='a' && (int64_t)a<='z') || ( (int64_t)a >='A' && (int64_t)a<='Z')){
    return 1;
  }
```

```
    else
    {
        return 0;
    }

}
uint64_t is_it_vowel(uint64_t set_input){   //маске со всеми гласными соответствует число 17842449( сама маска -
01000100000100000100010001)
    uint64_t set_mask=17842449;
    if((set_input&set_mask)){
        return 1;
    }
    else
    {
        return 0;
    }

}
uint64_t letter_to_set(char inp_a){

    return 1u<<(inp_a - 'a');

}
void test_to_set(){
    char c='a';
    assert(letter_to_set(c)==1);
    c='b';
    assert(letter_to_set(c)==2);
    c='c';
    assert(letter_to_set(c)==4);
    c='z'; //как можно заметить set(a(i)) = 2^(i-1), где i - номер буквы. z=2^25;
    assert(letter_to_set(c)==33554432);


}
void test_letter_or_not(){
    char c='a';
    assert(letter_or_not(c)==1);
     c='A';
    assert(letter_or_not(c)==1);
    c='o';
    assert(letter_or_not(c)==1);
    c='U';
    assert(letter_or_not(c)==1);
    c='0';
    assert(letter_or_not(c)==0);
    c=EOF;
    assert(letter_or_not(c)==0);
    c='\n';
    assert(letter_or_not(c)==0);

}

void test_is_it_vowel(){
    uint64_t c =1; //a
    assert(is_it_vowel(c)==1);
    c=16777216;//y
    assert(is_it_vowel(c)==1);
    c=4;
    assert(is_it_vowel(c)==0);
    c=33554432;
    assert(is_it_vowel(c)==0);
```

```
}
int64_t is_it_lda(char c){
  if ( c == ' ')
    return 1;
  else
    return 0;

}

void test_lda(){
    char c=' '; //1 test - correct +
    assert(is_it_lda(c)==1);
    c='r'; // 2 test - incorrect +
    assert(is_it_lda(c)==0);
    c=EOF; // 3 test - EOF +
    assert(is_it_lda(c)==0);


}


int64_t  is_it_cmma(char c){
   if ( c == ',')
    return 1;
   else
    return 0;

}

void test_cmma(){
   char c=','; //1 test - correct +
    assert(is_it_cmma(c)==1);
    c='r'; // 2 test - incorrect +
    assert(is_it_cmma(c)==0);
    c=EOF; // 3 test - EOF +
    assert(is_it_cmma(c)==0);

}

int64_t  is_it_tab(char c){
   if ( c == '\t')
    return 1;
   else
    return 0;

}

void test_tab(){
    char c='\t'; //1 test - correct +
    assert(is_it_tab(c)==1);
    c='r'; // 2 test - incorrect +
    assert(is_it_tab(c)==0);
    c=EOF; // 3 test - EOF +
    assert(is_it_tab(c)==0);


}
int64_t  is_it_n(char c){
   if ( c == '\n')
    return 1;
```

```
  else
   return 0;

}

void test_n(){
   char c='\n'; //1 test - correct +
   assert(is_it_n(c)==1);
   c='\t'; // 2 test - incorrect +
   assert(is_it_n(c)==0);
   c=EOF; // 3 test - EOF +
   assert(is_it_n(c)==0);


}
char A_to_a(char a){
   int64_t y=(int64_t)a;
   if(y >='A' && y<='Z'){
      return (char)(y-'A'+'a');
   }
   else return a;
}
void test_A_to_a(){
   char c='A';
   assert(A_to_a(c)=='a');
   c='g';
   assert(A_to_a(c)=='g');
   c='P';
    assert(A_to_a(c)=='p');




}
int main()
{

   test_lda();
   test_cmma();
   test_tab();
   test_n();
   test_to_set();
   test_letter_or_not();
   test_is_it_vowel();
   test_A_to_a();

   int64_t solution=0,flag=0;
   char a=' ',b=' ';

   while((a=getchar())!=EOF){
         if(flag==0){
            if (letter_or_not(a)){
               if (is_it_vowel(letter_to_set(a))){
                     flag=1;
               }
               else flag=2;
            }
            else if (is_it_cmma(a)||is_it_lda(a)||is_it_n(a)||is_it_tab(a)){
               flag=0;
            }
            else{
               flag=2;
```

```
            }

        }
        else if (flag==1){
            if (is_it_cmma(a)||is_it_lda(a)||is_it_n(a)||is_it_tab(a)){
                 if (letter_or_not(b)){
                    if (is_it_vowel(letter_to_set(b))){
                          solution=1;
                    }
                    else
                     flag =0;

                 }
                 else
                  flag=0;

        }
        else
            flag=1;

      }
      else if(flag==2){
        if (is_it_cmma(a)||is_it_lda(a)||is_it_n(a)||is_it_tab(a)){
              flag=0;
      }
      else
        flag=2;
      }
    if(flag==1){
      b=a;
    }
  }
  if(flag==1){
     if (letter_or_not(b)){
        if (is_it_vowel(letter_to_set(b))){
                          solution=1;
        }
        else
            flag =0;

     }
     else
       flag=0;
  }
  if( solution){
     printf("A required word is in this sequence\n");
  }
  else
     printf("A required word isn't in this sequence\n");




  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 |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |

**10. Замечания автора** по существу работы

Замечания отсутствуют, работа конструктивная.

**11. Выводы**

От лабораторной работы получил исключительно положительные эмоции и впечатления. По моему мнению, знания, приобретенные мною на данной лабораторной работе, помогли мне лучше осознать принципы работы операционных систем, принципы представления множеств в ЯП Си, принципы работы битовых операций в Си, отладчика Си, более пристально изучить язык программирования Си, научиться отлаживать свой код и находить ошибки разного типа, что несомненно поможет мне при решении практических задач.

Недочёты при выполнении задания могут быть устранены следующим образом: --

Подпись студента _____