**Отчет по лабораторной работе № 13** по курсу «Фундаментальная информатика»

Студентка группы Агеева Алиса Ивановна, № по списку 2 Контакты e-mail : [alisa.agewa2003@gmail.com](mailto:alisa.agewa2003@gmail.com)

Работа выполнена: «13» декабря 2021г.

Преподаватель: каф. 806 Никулин Сергей Петрович Отчет сдан « » 20 г., итоговая оценка

Подпись преподавателя

1. **Тема:** Множества.
2. **Цель работы:** Составить программу проверки характеристик введенных последовательностей слов и печати развернутого ответа.
3. **Задание** (*вариант №2*)**:** Есть ли слово, где все согласные разные?
4. **Оборудование** (студенческое) AMD Ryzen 7 4800H with Radeon Graphics 2.90 GHz

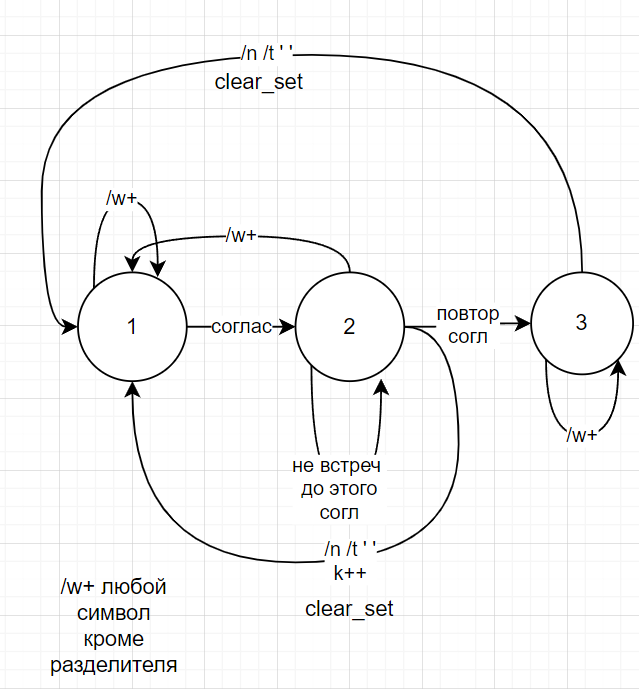
Программное обеспечение (студенческое):

Операционная система семейства Windows, наименование Windows 10 домашняя версия интерпретатор команд версия .

Система программирования версия Редактор текстов версия Утилиты операционной системы

Прикладные системы и программы Atom

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



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

План работы:

Слова во входной последовательности имеют произвольную длину и состоят только из строчных и прописных букв английского алфавита. Слова могут быть обрамлены любым количеством пробелов, запятых, знаков табуляции и переводов строк. Входной текст закодирован в произвольной кодировке, содержащей ASCII в качестве подмножества и допускающей ASCII-интерпретацию.

Вывести Yes, если результат проверки положительный, и No в противном случае. Как всегда, вывод должен завершаться Unix-переводом строки.

*Пункты 1-7 отчета составляются строго до начала лабораторной работы.*

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

alice@alice-VirtualBox:~$ cat l13.c  
  
#include <stdio.h>  
#include <ctype.h>  
#include <stdbool.h>  
#include <string.h>  
  
typedef unsigned set\_data\_elem;  
enum {  
    bits\_per\_char = 8,  
    bits\_per\_elem = sizeof(set\_data\_elem) \* bits\_per\_char,  
    datalen = (1 << bits\_per\_char) / bits\_per\_elem  
};  
  
typedef struct {  
    set\_data\_elem data[datalen];  
 } set;  
  
void set\_clear(set \*s) {  
    memset(s->data, 0, sizeof(s->data));  
}  
  
bool set\_in(const set \*s, int c) {  
    return (s->data[c/ bits\_per\_elem] & (1u << c % bits\_per\_elem)) !=0;  
}  
  
void set\_insert(set \*s, int c) {  
    s->data[c / bits\_per\_elem] |= 1u << (c % bits\_per\_elem);  
}  
  
int set\_size(const set \*s) {  
    int i, size = 0;  
    for(i = 0; i != 1 << bits\_per\_char; i++) {  
        if(set\_in(s, i)) {  
            size++;  
         }  
    }  
return size;  
}  
  
bool is\_razd(char c) {  
    if (c == ' ' | c == '\n' | c == EOF | c == ',') return true;  
    else return false;  
}  
bool is\_sogl(int c){  
    switch (c) {  
        case 'b':  
        case 'c':  
        case 'd':  
        case 'f':  
        case 'g':  
        case 'h':  
        case 'j':  
        case 'k':  
        case 'l':  
        case 'm':  
        case 'n':  
        case 'p':  
        case 'q':  
        case 'r':  
        case 's':  
        case 't':  
        case 'v':  
        case 'w':  
        case 'x':  
        case 'z':  
            return true;  
            break;  
        default:  
            return false;  
            break;  
  
 }  
}  
  
int main() {  
    char c;  
    int sost = 1;  
    int size;  
    int k = 0;  
    set num;  
    set\_clear(&num);  
    while (c != EOF) {  
      c = getchar();  
      switch (sost){  
        case 1: if (c != (is\_sogl(c))){  
          sost = 1;  
        } else {  
          sost = 2;  
          break;  
        }  
        case 2: size = set\_size(&num);  
        set\_insert(&num, tolower(c) - 'a');  
        if (is\_sogl(c)){  
          if (size != set\_size(&num)){  
          sost = 2;  
        } else {  
          sost = 3;  
        }  
      }  
        else if (is\_razd(c)){  
          sost = 1;  
          k+=1;  
          set\_clear(&num);  
        } else if (!(is\_sogl(c))){  
            sost = 1;  
          }  
        break;  
        case 3: if (is\_razd(c)){  
          sost = 1;  
          set\_clear(&num);  
        } else {  
          sost = 3;  
        }  
            break;  
      }  
      }  
  
    if (k != 0) printf("В тексте есть слова со всеми разными согласными\n");  
    if (k == 0) printf("В тексте нет таких слов\n");  
  
}

alice@alice-VirtualBox:~$ gcc l13.c  
alice@alice-VirtualBox:~$ ./a.out  
when we walk

В тексте есть слова со всеми разными согласными  
alice@alice-VirtualBox:~$ ./a.out  
wwen wwe wwalk

В тексте нет таких слов  
alice@alice-VirtualBox:~$ ./a.out  
wen  
we  
walk123

В тексте есть слова со всеми разными согласными  
alice@alice-VirtualBox:~$ ./a.out  
wwem wwe  
walk  
123f  
rttt

В тексте есть слова со всеми разными согласными  
alice@alice-VirtualBox:~$ ./a.out  
wem we12

В тексте есть слова со всеми разными согласными  
alice@alice-VirtualBox:~$ ./a.out  
wwe344

В тексте нет таких слов  
alice@alice-VirtualBox:~$

1. **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. | Дата | Время | Событие | Действие по исправлению | Примечание |
| или |
| дом. |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |

1. **Замечания автора** по существу работы : замечаний нет.
2. **Выводы :** было тяжело

Недочёты при выполнении задания могут быть устранены следующим образом:

Подпись студента