Министерство образования и науки Российской Федерации

МГТУ им .Н.Э Баумана

Факультет «Робототехника и комплексная автоматизация»

Кафедра «РК6»

**Отчет по лабораторной работе**

По курсу «Объектно-ориентированное программирование»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Выполнил: |  |  | Студент | Журавлев Н.В. |
|  |  |  | Группа | РК6-12Б |
|  |  |  |  |  |
| Проверил: |  |  |  | Кузьмина И.А. |
|  |  |  | Дата | 23.04.2020 |
|  |  |  | Подпись | \_\_\_\_\_\_\_\_\_\_\_ |

**Лабораторная работа №4**

Разработать программу для определения среднего размера слова в заданном текстовом файле и количества различных слов, длина которых не больше средней величины. Прoграмма должна анализировать только слова текста, состоящие из латинских букв. Имя текстового файла, где измеряются длины слов, должно передаваться программе через аргумент комадной строки её вызова. Результат измерений должен отображаться через поток протокола стандартной диагностики (stderr).

**Описание алгоритма**

1)Считывается по 1 символу,пока нет символов разделителей.  
2) Возращает курсор на количество считанных символов

3) Затем выделяется память на их количество,и создаётся строка  
4)Затем посимвольная проверка,если успешно то прибавляет длину строки и количество,и так пока не конец файла

5) Возращение в начало файла.

6)Считка строки,подсчёт всех меньше средней длины.

**Текст программы**

[#include](https://vk.com/im?sel=139380717&st=%23include) <stdlib.h>  
[#include](https://vk.com/im?sel=139380717&st=%23include) <stdio.h>  
[#include](https://vk.com/im?sel=139380717&st=%23include) <string.h>  
[#define](https://vk.com/im?sel=139380717&st=%23define) B\_B 97  
[#define](https://vk.com/im?sel=139380717&st=%23define) B\_E 122  
  
char\* scan(FILE\* file) {  
char c;  
int r = fscanf(file, "%c", &c);  
int size = 0;  
while (r != -1 && c != '\0' && c != ' ' && c != '\t' && c != '\n') {  
r = fscanf(file, "%c", &c);  
size++;  
}  
if (r == -1) {  
return NULL;  
}  
fseek(file, -1\*(size\*sizeof(char)+1), 1);  
char\* str = (char \*)malloc(size\*sizeof(char));  
if (str == NULL) {  
printf("Memory is out");  
return NULL;  
}  
int i = 0;  
r = fscanf(file, "%c", &c);  
while (r != -1 && c != '\0' && c != ' ' && c != '\t' && c != '\n') {  
str[i++] = c;  
r = fscanf(file, "%c", &c);  
}  
str[i] = '\0';  
return str;  
}  
  
int check\_symbol(char\* str) {  
if (str[0] == '\0') {  
return 0;  
}  
  
for (int i = 0; i < strlen(str); i++) {  
if (str[i] < B\_B || str[i] > B\_E) {  
return 0;  
}  
}  
  
return 1;  
}  
  
double average(FILE\* file) {  
char\* str;  
int sum = 0;  
int n = 0;  
  
while (str = scan(file)) {  
int check = check\_symbol(str);  
if (check) {  
n++;  
sum = sum + strlen(str);  
}  
free(str);  
}  
  
double average\_value = (double)sum / n;  
return average\_value;  
}  
int count\_word(FILE\* file, double sr) {  
char\* str;  
int count = 0;  
  
while (str = scan(file)) {  
int check = check\_symbol(str);  
if (check) {  
if (strlen(str) < sr) {  
count++;  
}  
}  
free(str);  
}  
  
return count;  
}  
  
int main(int argc, const char\*\* argv) {  
if (argc != 2) {  
return -1;  
}  
const char\* point\_file = argv[1];  
FILE\* file = fopen(point\_file, "r");  
if (file == NULL) {  
printf("file not open :(");  
return 1;  
}  
double average\_value = average(file);  
fseek(file, 0, 0);  
fprintf(stderr,"average\_value: %lf\n", average\_value);  
fprintf(stderr,"Count word < average\_value : %d\n", count\_word(file, average\_value));  
//printf("average\_value: %lf\n", average\_value);  
//printf("Count word < average\_value : %d\n", count\_word(file, average\_value));  
fclose(file);  
return 0;  
}

**Тесты**

**Ввод Вывод**

Aa1 aa a 1b b1 1

a al 1

a1

**Результат**

Cредняя длина слов, количество меньше этой длинны

**Списки литературы**

1)Волосатова Т. М., Родионов С. В. «Объектно-ориентированное программирование на С++» 2)http://bigor.bmstu.ru/?cnt/?doc=VU/base.cou