**Правительство Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего профессионального образования   
"Национальный исследовательский университет   
"Высшая школа экономики"**

Московский институт электроники и математики Национального

исследовательского университета "Высшая школа экономики"

Департамент прикладной математики

**ОТЧЕТ**

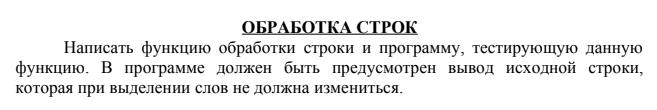
**По лабораторной работе № 7**

**По курсу «Алгоритмизация и программирование»**

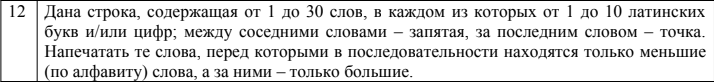
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | ФИО студента | | Номер группы | Дата |
| Индюченко Никита Андреевич | БПМ211 | 26.01.2022 |
|  |
|  |
|  |

**Москва – 2022г.**

**ЗАДАНИЕ (вариант №12)**

****

*Текст задания Вашего варианта*

**

**РЕШЕНИЕ**

Код программы с комментариями

#define \_CRT\_SECURE\_NO\_WARNINGS // чтобы могли пользоваться строковыми функциями (убрать ограничения)

#include <stdio.h>

#include <string.h> // библиотека, в которой str функции

#include <stdlib.h>

void print\_string(char\* \_str);// функция, которая выводит слова, которые удовлетворяют условию задачи

void print\_original\_str(char\* \_str);// функция, которая выводит исходную строку

void scanf\_string(char\* \_str);// функция, которая заполняет строку

int main(void) {

char\* s; // объявление строки

s = (void\*)malloc(330 \* sizeof(char)); // выделение памяти

if (s == NULL)

{

printf("Memory is not allocated\n");

return 1;

}

printf("Enter string:\n");

//scanf\_string(s); //считывание слов в строку

fgets(s, 330, stdin);

print\_string(s); // вывод строки, в которой хранятся слова по условию задачи

printf("Original string:\n");

//print\_original\_str(s); // вывод исходной строки

puts(s);

free(s);

}

void scanf\_string(char\* \_str)

{

char tmp;

int i = 0;

do

{

scanf\_s("%c", &tmp);

\_str[i] = tmp;

i++;

} while ((tmp != '.')||(tmp!='\n'));

}

void print\_string(char\* \_str)

{

char\* str\_1;

str\_1 = (void\*)malloc(330 \* sizeof(char));

if (str\_1 == NULL)

{

printf("Memory is not allocated in function <print\_string>\n");

}

else

{

char str\_2[338] = "String:\n"; // строка, в которой будут храниться все подходящие слова по условию

str\_1 = strcpy(str\_1, \_str); // strcpy - копирует строку str в str\_1 (включая '\0'), возвращает указатель на str\_1 (char\*)

int str\_in\_1\_word = 0; // счётчик кол-во слов

for (int i = 1; i < strlen(\_str); i++)

{

if (((\_str[i] == ',') || (\_str[i] == ' ') || (\_str[i] == '.')) && (\_str[i - 1] != ',') && (\_str[i - 1] != ' ') && (\_str[i - 1] != '.'))

{

str\_in\_1\_word++;

}

if (\_str[i] == '.')

{

break;

}

}

if (str\_in\_1\_word == 0)

{

printf("The string consists of zero word\n");

}

else if (str\_in\_1\_word == 1) // проверка на слова

{

printf("The string consists of one word\n");

}

else if (str\_in\_1\_word == 2)

{

printf("the string consists of two words\n");

}

else if (str\_in\_1\_word > 29)

{

printf("More than 30 words\n");

}

else // если 3 и более

{

char\* word\_1, \* word\_2, \* word\_3; // объявление строк и выделение памяти

word\_1 = (void\*)malloc(10 \* sizeof(char));

word\_2 = (void\*)malloc(10 \* sizeof(char));

word\_3 = (void\*)malloc(10 \* sizeof(char));

if ((word\_1 == NULL) || (word\_2 == NULL) || (word\_3 == NULL))

{

printf("Memory is not allocated in function <print\_string>\n");

}

else

{

word\_1 = strtok(str\_1, " ,."); // выделение слова из строки str\_1 по токенам при помощи strtok - типа char\* возвращает указатель на первый элемент не входящий в список токенов

word\_2 = strtok(NULL, " ,."); // так как выделили первое слово, то второе начинается с указателя который запомнил в прошлом вызове (пишем NULL)

word\_3 = strtok(NULL, " ,."); // аналогично берём 3-ье слово

int i = 0; // счётчик проходов в цикле

while (word\_3 != NULL)

{

i++;

if (i + 1 == str\_in\_1\_word) break; // чтобы не учитывал случай, когда второе слово является последним в строке.

if ((strcmp(word\_2, word\_1) > 0) && (strcmp(word\_3, word\_2) > 0)) // strcmp сравнивает строки, возвращает типа int. если первая строка больше 2-ой, то >0, обратное - <0. Равны - 0

{

strcat(str\_2, word\_2);//вставка в str\_2 слово word\_2. Типа char\*

strcat(str\_2, " ");

}

word\_1 = strcpy(word\_1, word\_2);

word\_2 = strcpy(word\_2, word\_3);

word\_3 = strtok(NULL, " ,.");

}

puts(str\_2);

// аналог вывода строки, написанная в библиотеке "string.h". Тип int

free(word\_1);

word\_2 = NULL;

free(word\_2);

word\_3 = NULL;

free(word\_3);

}

}

str\_1 = NULL;

free(str\_1);

}

}

void print\_original\_str(char\* \_str)

{

for (int i = 0; i < strlen(\_str); i++) // функция strlen возвращает кол-во символв в строке (int).

{

printf("%c", \_str[i]);

if (\_str[i] == '.') break;

}

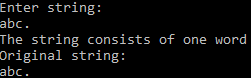
}

**ТЕСТЫ**

**Тест № 1**

**Если 1 слово**

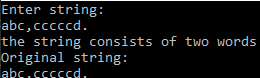
*Результаты теста 1*

**

**Тест № 2**

**Если 2 слова**

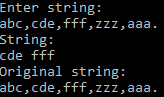
*Результаты теста 2*

**

**Тест № 3**

**Пример с двумя словами, которые подходят и являются соседними**

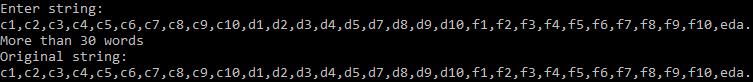
*Результаты теста 3*

**

**Тест № 4**

**Если будет более 30 слов**

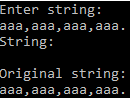
*Результаты теста 4*

**

**Тест № 5**

**Если все слова одинаковые**

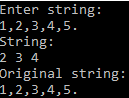
*Результаты теста 5*

****

**Тест № 6**

**Проверка на строку из цифр**

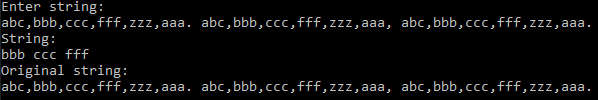
*Результаты теста 6*

****

**Тест № 7**

**Проверка на несколько предложений, то есть чтобы не сравнивал слова после точки.**

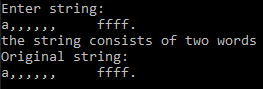
*Результаты теста 7*

****

**Тест № 8**

**Проверка на несколько знаков препинаний (,) или пробелов.**

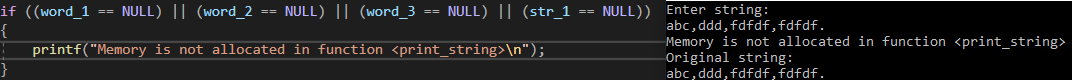
*Результаты теста 8*

****

**Тест № 9**

**Проверка на выделение памяти исходной строки, а также дополнительной динамической памяти в функции**

*Результаты теста 9*

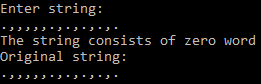
****

****

**Тест № 10**

**Если нет слов в строке, а только знаки препинания**

*Результаты теста 10*

****

**Тест № …**

*Результаты теста …*