МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Вычислительной техники

ОТЧЕТ

по лабораторной работе № 5
по дисциплине «Программирование»
Тема: **«Обработка строк»**

Студент гр. 3312 _____ Мохно Даниил.

Преподаватель _____ Аббас Саддам

Санкт-Петербург
2023

Цель работы.

Целью работы является изучение обработки строк в языке Си и получение практических навыков в программировании на этом языке.

Задание (вариант 5)

Ввести строку текста, которая состоит из слов и произвольного количества символов-разделителей до и после слов. Массив символов-разделителей вводится после ввода строки. Количество символов-разделителей задается. Определить номер и вывести слово максимальной длины (если таких слов несколько, вывести данные по последнему слову).

Постановка задачи и описание решения

Для начала создадим два массива символов, заведомо большой длинны. Первый массив - строка, второй - символы разделители. Через функцию fgets получим строку. Затем получаем количество символов разделителей, и передаём его и массив символов разделителей, в функцию получения символов разделителей. В ней перебираем массив пока итератор не станет равен количеству разделителей, получая с каждой итерацией символ разделитель и записывая его в массив. Теперь вызываем функцию нахождения индекса первого символа максимального слова, его длинны и порядкового номера слова (для удобства дальше в тексте называется find_max**)**. Ей параметрами передаём кол-во символов разделителей, массив символов разделителей, строку, полученную от пользователя, указатель на переменную, содержащую индекс первого символа максимально большого слова в строке и указатель на переменную, содержащую длину максимально большого слова в строке. В функции зададим длину текущего слова равной 0, индекс первого символа текущего слова равным 0 а порядковый номер текущего слова равным 1. Далее будем перебирать строку пока не дойдём до символа перехода на следующую строку (\n). В каждой итерации проверяем чтобы функция проверки символа вернула 1. Если да, увеличиваем текущую длину на 1, если при этом указатель

на длину максимального слова будет ссылаться на значение меньшее или равное длине текущего слова, то заменяем значение этого указателя длинной текущего слова, значение указателя ссылающегося на индекс первого символа максимально длинного слова в строке заменяем значением индекса первого символа текущего слова, в порядковый номер искомого слова записываем порядковый номер текущего слова. Если же функция проверки символа вернула 0, то обнуляем значение длинны текущего слова, а значение индекса первого символа слова делаем равным текущему значению итератора + 1 (+1 нужно потому, что символ текущего индекса — это символ разделитель) и инкриминируем значение порядкового номера текущего слова. Функция проверки символа принимает в себя параметрами i-тый символ строки, кол-во символов разделителей, и сам массив символов разделителей. Функция перебирает символы из массива символов разделителей и сравнивает их с переданным символом. На каждой итерации проверяем, если переданный символ из строки равен символу из массива, значит это символ разделитель, и мы возвращаем 0. Если символ не совпадёт не с одним из символов массива, то мы возвращаем 1. Функция find_max возвращает порядковый номер слова максимальной длинны. Теперь, когда мы имеем индекс первого символа слова максимальной длинны, его длину и порядковый номер, мы выводим их на экран с помощью функции вывода. Передадим в функцию индекс первого символа слова максимальной длинны, длину слова, и строку. Функция перебирает строку начиная с переданного индекса до переданного индекса + длинна слова, и выводит каждый элемент, таким образом мы выведем нужное нам слово. После чего выводим порядковый номер слова.

Описание переменных

- **Функция main():**

№	Имя переменно	Тип	Назначение
1	string[]	char	Строка слов и произвольного кол-ва символов разделителей
2	n	int	Кол-во символов разделителей
3	divisions[]	char	Массив символов разделителей
4	start_word_index	int	Индекс с которого начинается слово максимальной длинны
5	length	int	Длинна максимального слова
6	word_number	int	Порядковый номер слова

- **Ф-я получения символов разделителей get_divs(int n, char divs[]):**

№	Имя переменно	Тип	Назначение
1	n	int	Кол-во символов разделителей
2	divs[]	char	Массив символов разделителей
3	i	int	Итератор

- **Ф-я проверки, является ли символ разделителем check_s(char symb, int n, char divs[]):**

№	Имя переменно	Тип	Назначение
1	symb	char	Проверяемый символ
2	n	int	Кол-во символов разделителей
3	divs[]	char	Массив символов разделителей
4	j	int	Итератор

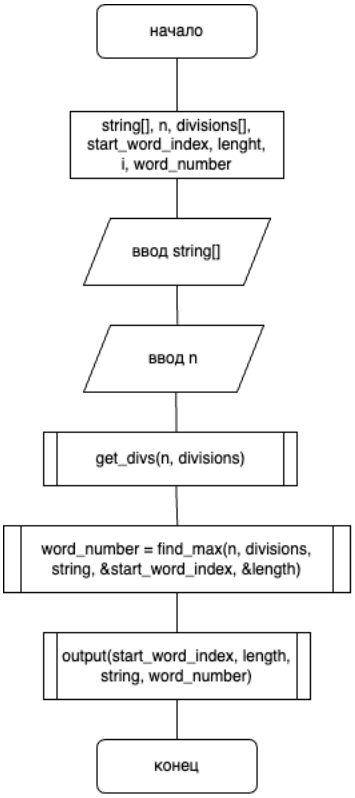
- **Функция нахождения индекса первого символа максимального слова, его длинны и порядкового номера слова find_max(int n, char divs[], char str[], int *start_max_word_index, int *max_len):**

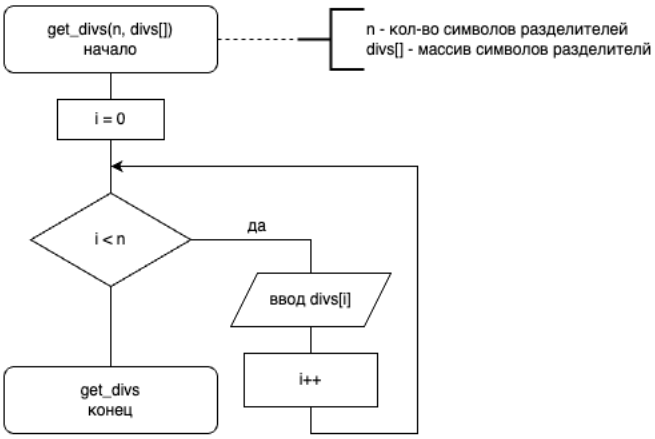
№	Имя переменно	Тип	Назначение
1	n	int	Кол-во символов разделителей
2	divs[]	char	Массив символов разделителей
3	str[]	char	Строка слов и произвольного кол-ва символов разделителей
4	start_max_word_index	int *	Указатель на переменную, содержащую индекс первого символа максимально большого слова
5	max_len	int *	Указатель на переменную, содержащую длину максимально большого слова
6	cur_len	int	Длинна текущего слова в итерации цикла
7	i	int	Итератор
8	start_word_index	int	Индекс начального символа слова в итерации цикла
9	word_number	int	Порядковый номер текущего слова
10	final_word_nubmer	int	Порядковый номер искомого слова

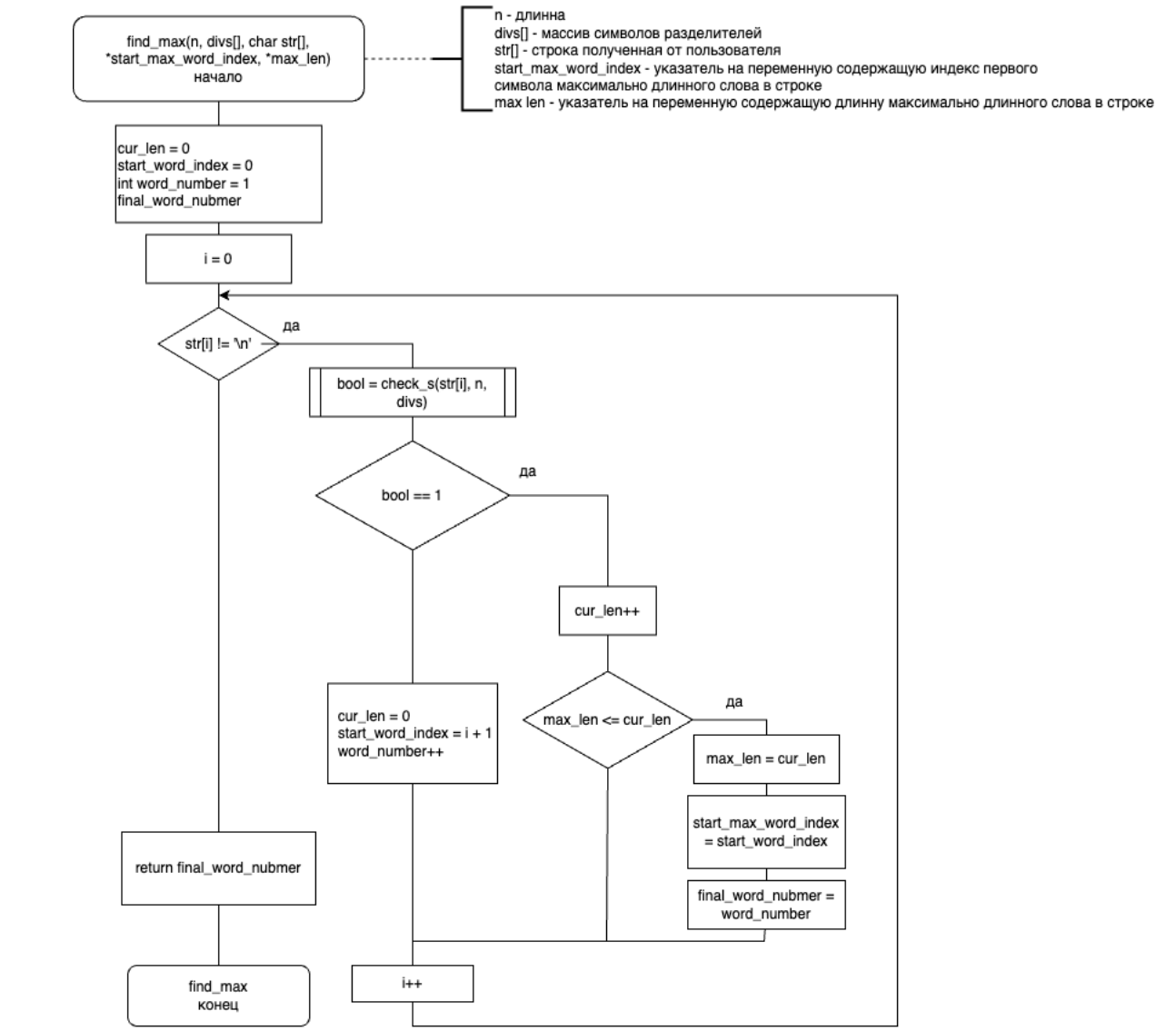
- **Функция вывода слова максимальной длинны и его длинны на экран output(int stw, int len, char str[], int word_numb)**

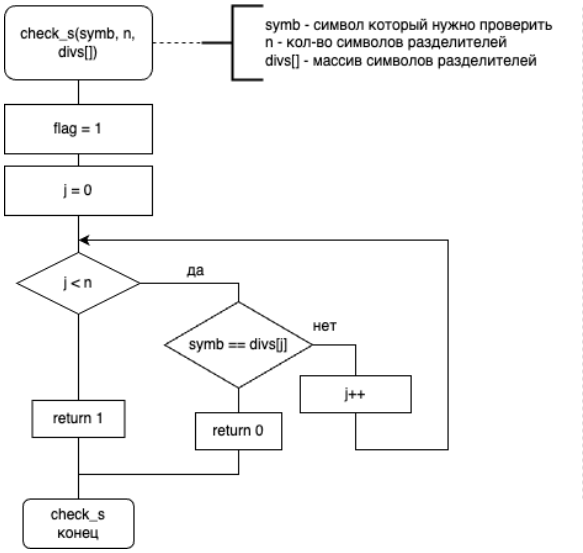
№	Имя переменно	Тип	Назначение
1	stw	int	Индекс максимально длинного слова в строке
2	len	int	Длинна максимально длинного слова в строке
3	str[]	char	Строка слов и произвольного кол-ва символов разделителей
4	i	int	Итератор
5	word_numb	int	Порядковый номер текущего слова

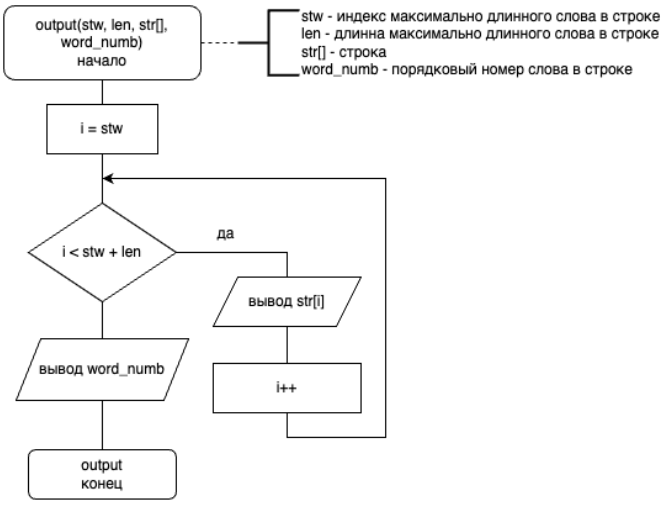
Схема алгоритма






find_max(n, divs[], char str[], *start_max_word_index, *max_len)
начало
n - длинна
divs[] - массив символов разделителей
str[] - строка полученная от пользователя
start_max_word_index - указатель на переменную содержащую индекс первого символа максимально длинного слова в строке
max len - указатель на переменную содержащую длинну максимально длинного слова в строке
cur_len = 0
start_word_index = 0
int word_number = 1
final_word_nubmer
i = 0
str[i] != '\n'
да
bool = check_s(str[i], n, divs)
bool == 1
да
cur_len++
max_len <= cur_len
да
max_len = cur_len
start_max_word_index = start_word_index
final_word_nubmer = word_number
cur_len = 0
start_word_index = i + 1
word_number++
i++
return final_word_nubmer
find_max
конец


check_s(symb, n, divs[])
symb - символ который нужно проверить
n - кол-во символов разделителей
divs[] - массив символов разделителей
flag = 1
j = 0
j < n
да
symb == divs[j]
нет
j++
return 1
return 0
check_s
конец


output(stw, len, str[], word_numb)
начало
stw - индекс максимально длинного слова в строке
len - длинна максимально длинного слова в строке
str[] - строка
word_numb - порядковый номер слова в строке
i = stw
i < stw + len
да
вывод str[i]
i++
вывод word_numb
output
конец

Контрольные примеры

Пример №	Входные данные	Выходные данные
1	hello_world*how+are*you 3 _*+	world 2
2	Neque.porro,quisquam:esto quit 4 :. ,	quisquam 3
3	Neque.porro,quisquam:esto quit 3 :.,	esto quit 4
4	iii,oooo.ppppp 2 ,.	ppppp 3

Текст программы

```
#include <stdio.h>

#define SIZE 100

void get_divs(int n, char divs[]);

int check_s(char symb, int n, char divs[]);

int find_max(int n, char divs[], char str[], int *start_max_word_index, int *max_len);

void output(int stw, int len, char str[], int word_numb);

int main()
{
    char string[SIZE];
    int n;
    char divisions[SIZE];
    int start_word_index = 0;
    int length = 0;
    int word_number;

    fgets(string, SIZE, stdin);
    scanf("%i\n", &n);
    get_divs(n, divisions);

    word_number = find_max(n, divisions, string, &start_word_index,
             &length);

    output(start_word_index, length, string, word_number);

    return 0;
}

void get_divs(int n, char divs[])
```

```
{
    for (int i = 0; i < n; i++)
        scanf("%c", &divs[i]);
}

int check_s(char symb, int n, char divs[])
{
    for (int j = 0; j < n; j++)
        if (symb == divs[j])
            return 0;
    return 1;
}

int find_max(int n, char divs[], char str[], int *start_max_word_index, int *max_len)
{
    int cur_len = 0, i;
    int start_word_index = 0;
    int word_number = 1, final_word_nubmer;
    for (i = 0; str[i] != '\n'; i++)
    {

        if (check_s(str[i], n, divs))
        {
            cur_len++;
            if (*max_len <= cur_len)
            {
                *max_len = cur_len;
                *start_max_word_index = start_word_index;
                final_word_nubmer = word_number;
            }
        }
        else
        {
            cur_len = 0;
            start_word_index = i + 1;
            word_number++;
        }
    }
    return final_word_nubmer;
}

void output(int stw, int len, char str[], int word_numb)
{
    for (int i = stw; i < stw + len; i++)
    {
        printf("%c", str[i]);
    }
    printf("\n%i\n", word_numb);
}
```

Примеры выполнения программы

Пример 1

```
"/Users/daniilmohno/Library/
hello_world*how+are*you
3
_*+
world
2
```

Пример 2

```
"/Users/daniilmohno/Library/Mob
Neque.porro,quisquam:esto quit
4
:. ,
quisquam
3
```

Пример 3

```
"/Users/daniilmohno/Library/Mobi
Neque.porro,quisquam:esto quit
3
:.,
esto quit
4
```

Пример 4

```
"/Users/daniilmohno/Library/
iii,oooo.ppppp
2
,.
ppppp
3
```

Выводы.

В результате выполнения работы была изучена обработка строк в языке Си и получены практические навыки в программировании на этом языке.