Министерство образования и науки России

Федеральное государственное автономное образовательное

учреждение высшего образования

“Санкт-Петербургский государственный электротехнический

университет им. В. И. Ульянова (Ленина)”

(СПБГЭТУ “ЛЭТИ”)

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

**ОТЧЕТ**

**по лабораторной работе № 6 на тему**

**“Строки языка С / C++” по**

**дисциплине “Программирование”**

**Выполнил:** *Студент группы 5307 Грицай Н. Д.*

**Проверила:** *к.т.н., доцент Сискович Т.И.*

**Подпись преподавателя:\_\_\_\_\_\_\_\_**

Санкт-Петербург

2015

Оглавление

[Цель работы 3](#_Toc435367483)

[1.Задание 3](#_Toc435367484)

[2.Уточнение задания 3](#_Toc435367485)

[3.Контрольные примеры 3](#_Toc435367486)

[4.Описание переменных главной функции 4](#_Toc435367487)

[5.Схема главной функции 4](#_Toc435367488)

[6.Описание функций 4](#_Toc435367489)

[6.1.Функция menu 4](#_Toc435367490)

[6.2.Функция reference 5](#_Toc435367491)

[6.3.Функция waitforenter 5](#_Toc435367492)

[6.4.Функция strlength 5](#_Toc435367493)

[6.5.Функция \_getchar 6](#_Toc435367494)

[6.6.Функция putstring 6](#_Toc435367495)

[6.7.Функция getstr 7](#_Toc435367496)

[6.8.Функция isseparator 8](#_Toc435367497)

[6.9.Функция findwords 8](#_Toc435367498)

[6.10.Функция wordsort 9](#_Toc435367499)

[6.11.Функция wordtostr 10](#_Toc435367500)

[6.12.Функция dsfree 12](#_Toc435367501)

[7.Иерархия вызовов функций программы 13](#_Toc435367502)

[8.Краткое описание алгоритма 13](#_Toc435367503)

[9.Исходный код программы 14](#_Toc435367504)

[10.Результат 19](#_Toc435367505)

[Вывод 19](#_Toc435367506)

[*Приложение* 20](#_Toc435367507)

# Цель работы

Получить практические навыки использования строк языка С / C++.

# 1.Задание

Разработать алгоритм и написать программу, выполняющую многократно по желанию пользователя ввод строки, контрольный вывод, обработку исходной строки и вывод результата.

Результирующая строка формируется из исходной по следующему правилу: слова отсортированы по возрастанию их длин и разделены одним пробелом.

# 2.Уточнение задания

-В программе строки реализовать в виде динамических структур типа char.

-Слова могут разделяться одним или несколькими пробелами и знакам (разделителями).

-Разделителями являются следующие символы: . , ; : ' " \t ! ? ( ) { } [ ] / | \ < > - \_

-Длина строки не может превышать 100 символов

# 3.Контрольные примеры

Контрольные примеры представлены в таблице 1.

*Таблица 1. Контрольные примеры*

|  |  |  |  |
| --- | --- | --- | --- |
| **№ примера** | **Исходные данные** | | **Результат** |
| **Строка str1** | **Примечания** | **Строка str2** |
| 1 | Слова будут отсортированы по возрастанию их длин |  | по их длин Слова будут возрастанию отсортированы |
| 2 | 4444 22 333 1 7777777 55555 |  | 1 22 333 4444 55555 7777777 |
| 3 |  | пустая строка | “В строке нет ни одного слова” |
| 4 | В строке нет ни одного слова |  | В ни нет слова строке одного |
| 5 | 1.2,3:4?5:6;7[8]9)10(11<12>13-14\_15!16 |  | 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 |
| 6 | Данные\*символы&dont@belong%to#separators |  | Данные\*символы&dont@belong%to#separators |

# 4.Описание переменных главной функции

Описание переменных представлено в таблице 2.

*Таблица 2. Описание переменных главной функции*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str1, str2 | char\* | Указатели на первые символы исходной и результирующей строк |
| wordstart | short\* | Указатель на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\* | Указатель на первый элемент массива, хранящего длины слов |
| wordct | short | Количество слов в строке |
| isprogend | short | Флажок выхода из программы (1-выйти/0-нет) |
| isstr1, isstr2 | short | Флажки наличия исходной и результирующей строк (1-есть/0-нет) |

# 5.Схема главной функции

Схема главной функции представлена на рисунках 1.1, 1.2 и 1.3 в Приложении.

# 6.Описание функций

## 6.1.Функция menu

**Назначение функции:** вывод меню на экран

**Прототип функции:** char menu();

**Возврат функции:** символ, введенный пользователем

**Пример вызова функции:** char c = menu();

**Схема функции:** схема функции menu представлена на рисунке 2 в Приложении

## 6.2.Функция reference

**Назначение функции:** вывод справки на экран

**Прототип функции:** void reference();

**Пример вызова функции:** reference();

**Схема функции:** схема функции reference представлена на рисунке 3 в Приложении

## 6.3.Функция waitforenter

**Назначение функции:** ожидание нажатия пользователем клавиши Enter

**Прототип функции:** void waitforenter();

**Пример вызова функции:** waitforenter();

**Схема функции:** схема функции dsfree представлена на рисунке 4 в Приложении

## 6.4.Функция strlength

**Назначение функции:** поиск длины строки

**Прототип функции:** short strlength(char \*str);

**Возврат функции:** длина строки

**Описание формальных переменных функции:** описание формальных переменных функции strlength представлено в таблице 3

**Пример вызова функции:** int i = strlength(str1);

**Описание фактических параметров функции:** описание фактических параметров функции strlength представлено в таблице 4

**Описание локальных переменных функции:** описание локальных переменных функции strlength представлено в таблице 5

**Схема функции:** схема функции strlength представлена на рисунке 5 в Приложении

*Таблица 3. Описание формальных переменных функции* strlength

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str | char\* | Указатель на первый символ строки |

*Таблица 4. Описание локальных переменных функции* strlength

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i | short | Вспомогательная переменная |

*Таблица 5. Описание фактических параметров функции* strlength

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str1 | char\* | Указатель на первый символ строки |

## 6.5.Функция \_getchar

**Назначение функции:** Ввод символа в режиме реального времени и вывод его на экран

**Прототип функции:** char \_getchar();

**Пример вызова функции:** char c = \_getchar();

**Описание локальных переменных функции:** описание локальных переменных функции \_getchar представлено в таблице 6

**Схема функции:** схема функции \_getchar представлена на рисунке 6 в Приложении

*Таблица 6. Описание локальных переменных функции* \_getchar

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| c | char | Символ, введенный пользователем |

## 6.6.Функция putstring

**Назначение функции:** вывод строки на экран

**Прототип функции:** void putstring(char \*str);

**Описание формальных переменных функции:** описание формальных переменных функции putstring представлено в таблице 7

**Пример вызова функции:** putstring(str1);

**Описание фактических параметров функции:** описание фактических параметров функции putstring представлено в таблице 8

**Описание локальных переменных функции:** описание локальных переменных функции putstring представлено в таблице 9

**Схема функции:** схема функции putstring представлена на рисунке 7 в Приложении

*Таблица 7. Описание формальных переменных функции* putstring

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str | char\* | Указатель на первый символ строки |

*Таблица 8. Описание локальных переменных функции* putstring

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i | short | Вспомогательная переменная |

*Таблица 9. Описание фактических параметров функции* putstring

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str1 | char\* | Указатель на первый символ строки |

## 6.7.Функция getstr

**Назначение функции:** ввод строки

**Прототип функции:** char \*getstr();

**Возврат функции:** указатель на первый символ строки

**Пример вызова функции:** str1 = getstr();

**Описание локальных переменных функции:** описание локальных переменных функции getstr представлено в таблице 10

**Схема функции:** схема функции getstr представлена на рисунке 8 в Приложении

*Таблица 10. Описание локальных переменных функции* getstr

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str | char\* | Указатель на первый символ строки |
| count | short | Вспомогательная переменная |

## 6.8.Функция isseparator

**Назначение функции:** проверка, является ли символ разделителем

**Прототип функции:** short isseparator(char c);

**Возврат функции:** 1 - если символ является разделителем/0 – если нет

**Описание формальных переменных функции:** описание формальных переменных функции isseparator представлено в таблице 11

**Пример вызова функции:** bool istrue = isseparator(c);

**Описание фактических параметров функции:** описание фактических параметров функции strlength представлено в таблице 12

**Описание локальных переменных функции:** описание локальных переменных функции isseparator представлено в таблице 13

**Схема функции:** схема функции isseparator представлена на рисунке 9 в Приложении

*Таблица 11. Описание формальных переменных функции* isseparator

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| c | char | Символ |

*Таблица 12. Описание локальных переменных функции* isseparator

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| separators | char[] | Указатель на первый элемент массива, хранящего символы – разделители |
| isres | short | Флажок – результат (1-символ является разделителем/0-нет) |

*Таблица 13. Описание фактических параметров функции* isseparator

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| с | char | Символ |

## 6.9.Функция findwords

**Назначение функции:** поиск слов в строке и формирование массивов их начал и длин

**Прототип функции:** void findwords(char \*str, short \*\*wordstart, short \*\*wordlen, short \*wordct);

**Описание формальных переменных функции:** описание формальных переменных функции findwords представлено в таблице 14

**Пример вызова функции:** findwords(str, &wordstart, &wordlen, &wordct);

**Описание фактических параметров функции:** описание фактических параметров функции findwords представлено в таблице 15

**Описание локальных переменных функции:** описание локальных переменных функции findwords представлено в таблице 16

**Схема функции:** схема функции findwords представлена на рисунке 10 в Приложении

*Таблица 14. Описание формальных переменных функции* findwords

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str | char\* | Указатель на первый символ строки |
| wordstart | short\*\* | Адрес указателя на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\*\* | Адрес указателя на первый элемент массива, хранящего длины слов |
| wordct | short\* | Адрес переменной, хранящей количество слов в строке |

*Таблица 15. Описание локальных переменных функции* findwords

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i | short | Вспомогательная переменная |

*Таблица 16. Описание фактических параметров функции* findwords

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str | char\* | Указатель на первый символ строки |
| &wordstart | short\*\* | Адрес указателя на первый элемент массива, хранящего индексы начал слов |
| &wordlen | short\*\* | Адрес указателя на первый элемент массива, хранящего длины слов |
| &wordct | short\* | Адрес переменной, хранящей количество слов в строке |

## 6.10.Функция wordsort

**Назначение функции:** сортировка элементов массивов, хранящих индексы начал слов и их длины, по возрастанию длин слов

**Прототип функции:** void wordsort(short \*wordstart, short \*wordlen, short wordct);

**Описание формальных переменных функции:** описание формальных переменных функции wordsort представлено в таблице 15

**Пример вызова функции:** wordsort(wordstart, wordlen, wordct);

**Описание фактических параметров функции:** описание фактических параметров функции wordsort представлено в таблице 16

**Описание локальных переменных функции:** описание локальных переменных функции wordsort представлено в таблице 17

**Схема функции:** схема функции wordsort представлена на рисунке 11 в Приложении

*Таблица 15. Описание формальных переменных функции* wordsort

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| wordstart | short\* | Указатель на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\* | Указатель на первый элемент массива, хранящего длины слов |
| wordct | short | Количество слов |

*Таблица 16. Описание локальных переменных функции* wordsort

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i, j, t | short | Вспомогательные переменные |

*Таблица 17. Описание фактических параметров функции* wordsort

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| wordstart | short\* | Указатель на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\* | Указатель на первый элемент массива, хранящего длины слов |
| wordct | short | Количество слов |

## 6.11.Функция wordtostr

**Назначение функции:** формирование строки из исходной по заданным длинам и индексам начал слов

**Прототип функции:** char \*wordtostr(char \*str1, short \*wordstart, short \*wordlen, short wordct);

**Возврат функции:** указатель на первый символ результирующей строки

**Описание формальных переменных функции:** описание формальных переменных функции wordtostr представлено в таблице 18

**Пример вызова функции:** char c = wordtostr(str1, wordstart, wordlen, wordct);

**Описание фактических параметров функции:** описание фактических параметров функции wordtostr представлено в таблице 19

**Описание локальных переменных функции:** описание локальных переменных функции wordtostr представлено в таблице 20

**Схема функции:** схема функции wordtostr представлена на рисунке 12 в Приложении

*Таблица 18. Описание формальных переменных функции* strlength

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str1 | char\* | Указатель на первый символ исходной строки |
| wordstart | short\* | Указатель на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\* | Указатель на первый элемент массива, хранящего длины слов |
| wordct | short | Количество слов |

*Таблица 19. Описание локальных переменных функции* strlength

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| i, j, k | short | Вспомогательная переменная |

*Таблица 20. Описание фактических параметров функции* strlength

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str1 | char\* | Указатель на первый символ исходной строки |
| wordstart | short\* | Указатель на первый элемент массива, хранящего индексы начал слов |
| wordlen | short\* | Указатель на первый элемент массива, хранящего длины слов |
| wordct | short | Количество слов |

## 6.12.Функция dsfree

**Назначение функции:** освобождение памяти, выделенной под динамическую структуру

**Прототип функции:** void \*dsfree(void \*ds);

**Возврат функции:** NULL

**Описание формальных переменных функции:** описание формальных переменных функции dsfree представлено в таблице 21

**Пример вызова функции:** str1 = dsfree(str1);

**Описание фактических параметров функции:** Описание фактических параметров функции dsfree представлено в таблице 22

**Схема функции:** схема функции dsfree представлена на рисунке 13 в Приложении

*Таблица 21. Описание формальных переменных функции dsfree*

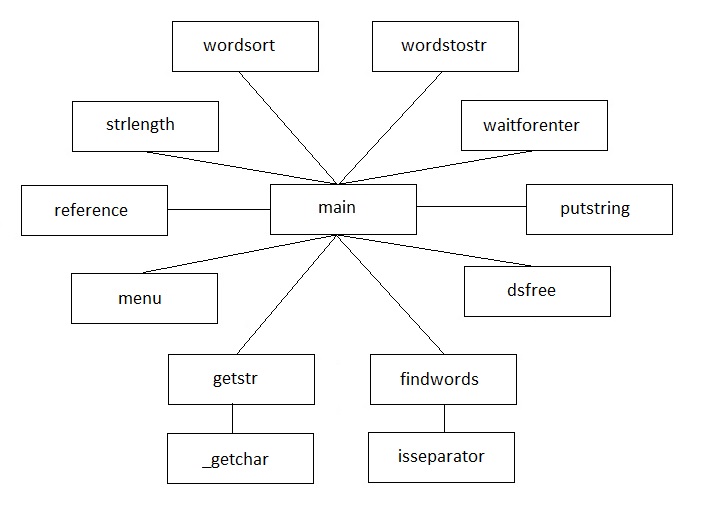
|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| ds | void\* | Указатель на переменную |

*Таблица 22. Описание фактических параметров функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str1 | char\* | Указатель на первый символ строки |

# 7.Иерархия вызовов функций программы

Иерархия вызовов функций программы представлена на рисунке 10



*Рисунок 10. Иерархия вызовов функций программы*

# 8.Краткое описание алгоритма

1. Вывод меню на экран и переход к пункту, выбранному пользователем с помощью функции menu:

1)Ввод строки с помощью функции getstr. Переход в пункт 2

2)Вывод исходной строки с помощью функции putstring или сообщение о том, что ее нет. Переход в пункт 2

3)Обработка исходной строки с помощью функций findwords, wordsort, wordtostr или сообщение о том, что нет исходной строки, или сообщение о том, что результирующая строка уже сформирована. Переход в пункт 2

4)Вывод результирующей строки с помощью функции putstring или сообщение о том, что ее нет. Переход в пункт 2

5)Вывод справки на экран с помощью функции reference. Переход в пункт 2

6)Ввод isprogend

Если пользователь выбрал неверный пункт меню, то сообщение об этом

1. Если(!isprogend), то: переход в пункт 1, иначе: выход из программы

# 9.Исходный код программы

#include "stdio.h"

#include "malloc.h"

#include "Windows.h"

#include "conio.h"

//возврат символа, введенного пользователем, в режиме реального времени

char \_getchar();

//ввод строки и возврат указателя на первый символ строки

char \*getstr();

//вывод меню на экран и возврат выбранного пользователем пункта

char menu();

//вывод строки на экран

void putstring(char \*str);

//вывод справки на экран

void reference();

//проверка, является ли символ разделителем

short isseparator(char c);

//сортировка индексов слов по возрастанию их длин

void wordsort(short \*wordstart, short \*wordlen, short wordct);

//формирование структур начала и длин слов строки

void findwords(char \*str, short \*\*wordstart, short \*\*wordlen, short \*wordct);

//формирование результирующей строки по индексам начал и длин слов из исходной строки

char \*wordtostr(char \*str1, short \*wordstart, short \*wordlen, short wordct);

//освобождение памяти динамической структуры

void \*dsfree(void \*ds);

int main()

{

char \*str1 = NULL, \*str2 = NULL;

short \*wordstart = NULL, \*wordlen = NULL, wordct, isprogend = 0, isstr1 = 0,

isstr2 = 0;

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("title = 1.6 Сортировка слов в строке");

do

{

system("cls");

switch (menu())

{

case '1': //ввод строки

if (str1 != NULL)

str1 = (char\*)dsfree(str1);

isstr1 = 1;

isstr2 = 0;

system("cls");

printf\_s("Введите строку:\n");

rewind(stdin);

str1 = getstr();

if (strlength(str1) == STR\_MAX\_LEN)

printf\_s("\nДлина строки достигла лимита. ");

printf\_s("\nВвод завершен.\n");

waitforenter();

break;

case '2': //вывод исходной строки

system("cls");

if (isstr1)

{

if (strlength(str1))

{

printf\_s("Исходная строка:\n");

putstring(str1);

}

else

printf\_s("В исходной строке нет символов");

}

else

printf\_s("Ошибка! Сначала введите строку");

printf\_s("\n");

waitforenter();

break;

case '3': //обработка исходной строки

system("cls");

if (isstr1)

{

if (!isstr2)

{

if (str2 != NULL)

str2 = (char\*)dsfree(str2);

if (wordstart != NULL)

wordstart = (short\*)dsfree(wordstart);

if (wordlen != NULL)

wordlen = (short\*)dsfree(wordlen);

isstr2 = 1;

findwords(str1, &wordstart, &wordlen, &wordct);

wordsort(wordstart, wordlen, wordct);

str2 = wordtostr(str1, wordstart, wordlen, wordct);

printf\_s("Строка обработана");

}

else

printf\_s("Строка уже обработана!");

}

else

printf\_s("Ошибка! Сначала введите исходную строку");

printf\_s("\n");

waitforenter();

break;

case '4': //вывод слов

system("cls");

if (isstr2)

{

if (wordct)

{

printf\_s("Результирующая строка:\n");

putstring(str2);

}

else

printf\_s("В строке нет ни одного слова");

}

else

printf\_s("Ошибка! Сначала обработайте строку");

printf\_s("\n");

waitforenter();

break;

case '5': //справка

system("cls");

reference();

waitforenter();

break;

case '6': //выход

system("cls");

printf\_s("Вы уверены, что хотите выйти?(1-Да/0-Нет): ");

scanf\_s("%hi", &isprogend);

rewind(stdin);

break;

default:

printf\_s("Ошибка! Введите существующий пункт меню\n");

rewind(stdin);

waitforenter();

}

} while (!isprogend);

if (str1 != NULL)

str1 = (char\*)dsfree(str1);

if (str2 != NULL)

str2 = (char\*)dsfree(str2);

if (wordstart != NULL)

wordstart = (short\*)dsfree(wordstart);

if (wordlen != NULL)

wordlen = (short\*)dsfree(wordlen);

return 0;

}

//Описание: вывод меню на экран

//Возврат: символ, введенный пользователем

char menu()

{

printf\_s("==Меню==\n");

printf\_s("1 - Ввод строки\n");

printf\_s("2 - Вывод исходной строки\n");

printf\_s("3 - Обработка исходной строки\n");

printf\_s("4 - Вывод слов\n");

printf\_s("5 - Справка\n");

printf\_s("6 - Выход\n");

printf\_s("\nВведите пункт меню: ");

rewind(stdin);

return getchar();

}

//Описание: вывод справки на экран

void reference()

{

printf\_s("==Справка==\n");

printf\_s("-Слова строки сортируются по увеличению их длин\n");

printf\_s("-Длина строки не превышает %d\n", STR\_MAX\_LEN);

printf\_s("-Разделителями являются следующие символы: \"\\n.,;:\'\"\\t!?(){}[]/|\\<>-\_\"\n");

}

//Описание: ввод символа в режиме реального времени

//Возврат: символ, введенный пользователем

void waitforenter()

{

printf\_s("Для продолжения нажмите клавишу Enter");

rewind(stdin);

for (; \_getch() != '\r';); //'\r' == '\t'

}

//Описание: поиск длинны строки, введенной пользователем

//Возврат: длинна строки, веденной пользователем

short strlength(char \*str)

{

short i;

for (i = 0; \*(str + i) != '\0'; i++);

return (i);

}

//Описание: ввод символа в режиме раельного времени

//Возврат: символ, введенный пользователем

char \_getchar()

{

char c;

c = \_getch();

if (c == '\r')//'\r' == '\n' в функции \_getch()

c = '\n';

printf\_s("%c", c);

return c;

}

//Описание: вывод строки на экран

void putstring(char \*str)

{

for (short i = 0; \*(str + i) != '\0'; i++)

putchar(\*(str + i));

}

//Описание: вывод текста на экран

void putstrings(char \*\*strings, short strct)

{

short voidstrct = 0;

for (short i = 0; i < strct; i++)

{

putstring(\*(strings + i));

printf\_s("\n");

if (!strlength(\*(strings + i)))

voidstrct++;

}

if (voidstrct)

printf\_s("\nПустых строк в тексте: %hi\n", voidstrct);

}

//Описание: освобождение памяти динамической структуры

//Возврат: NULL

void \*dsfree(void \*ds)

{

free(ds);

return NULL;

}

void \*\*dsfree(void \*\*ds, short index1ct)

{

for (short i = 0; i < index1ct; i++)

free(\*(ds + i));

free(ds);

return NULL;

}

//Описание: ввод строки

//Возврат: указатель на первый элемент строки

char \*getstr()

{

char \*str = (char\*)malloc(sizeof(char));

short count = 0;

while (((\*(str + count) = \_getchar()) != '\n') && (count < STR\_MAX\_LEN))

{

if (\*(str + count) == '\b')

{

if (count)

{

printf\_s(" \b");

count--;

}

}

else

{

count++;

}

str = (char\*)realloc(str, sizeof(char) \* (count + 1));

}

\*(str + count) = '\0';

return str;

}

//Описание: проверка, является ли символ разделителем

//Возврат: (1/0)

short isseparator(char c)

{

char separators[] = { " \n.,;:\'\"\t!?(){}[]/|\\<>-\_" };

bool isres = 0;

for (short i = 0; (separators[i] != '\0') && !isres; i++)

if (c == separators[i])

isres = 1;

return isres;

}

//Описание: поиск слов в строке

void findwords(char \*str, short \*\*wordstart, short \*\*wordlen, short \*wordct)

{

\*wordct = 0;

for (short i = 0; \*(str + i) != '\0'; i++)

if (!isseparator(\*(str + i)))

{

\*wordlen = (short\*)realloc(\*wordlen, sizeof(short) \* ((\*wordct) +

1));

\*wordstart = (short\*)realloc(\*wordstart, sizeof(short) \* ((\*wordct) +

1));

\*((\*wordstart) + \*wordct) = i;

for (; !isseparator(\*(str + i)) && (\*(str + i) != '\0'); i++);

\*((\*wordlen) + \*wordct) = (i - \*((\*wordstart) + \*wordct));

(\*wordct)++;

i--;

}

}

//Описание: сортировка слов по возрастанию

void wordsort(short \*wordstart, short \*wordlen, short wordct)

{

for (short i = 0; i < wordct - 1; i++)

for (short j = (wordct - 1); j > i; j--)

if (\*(wordlen + j - 1) > \*(wordlen + j))

{

short t = \*(wordlen + j - 1);

\*(wordlen + j - 1) = \*(wordlen + j);

\*(wordlen + j) = t;

t = \*(wordstart + j - 1);

\*(wordstart + j - 1) = \*(wordstart + j);

\*(wordstart + j) = t;

}

}

//Описание: формирует строку по заданым началам и длиннам слов исходной строки

//Возврат: указатель на первый символ строки

char \*wordtostr(char \*str1, short \*wordstart, short \*wordlen, short wordct)

{

char \*str2 = (char\*)malloc(sizeof(char));

short strlen = 0, i = -1;

for (short j = 0; j < wordct; j++)

{

i++;

strlen += (\*(wordlen + j)) + 1;

str2 = (char\*)realloc(str2, sizeof(char) \* strlen);

for (short k = 0; k < \*(wordlen + j); i++, k++)

\*(str2 + i) = \*(str1 + (\*(wordstart + j)) + k);

\*(str2 + i) = ' ';

}

if (wordct)

\*(str2 + i) = '\0';

else

\*str2 = '\0';

return str2;

}

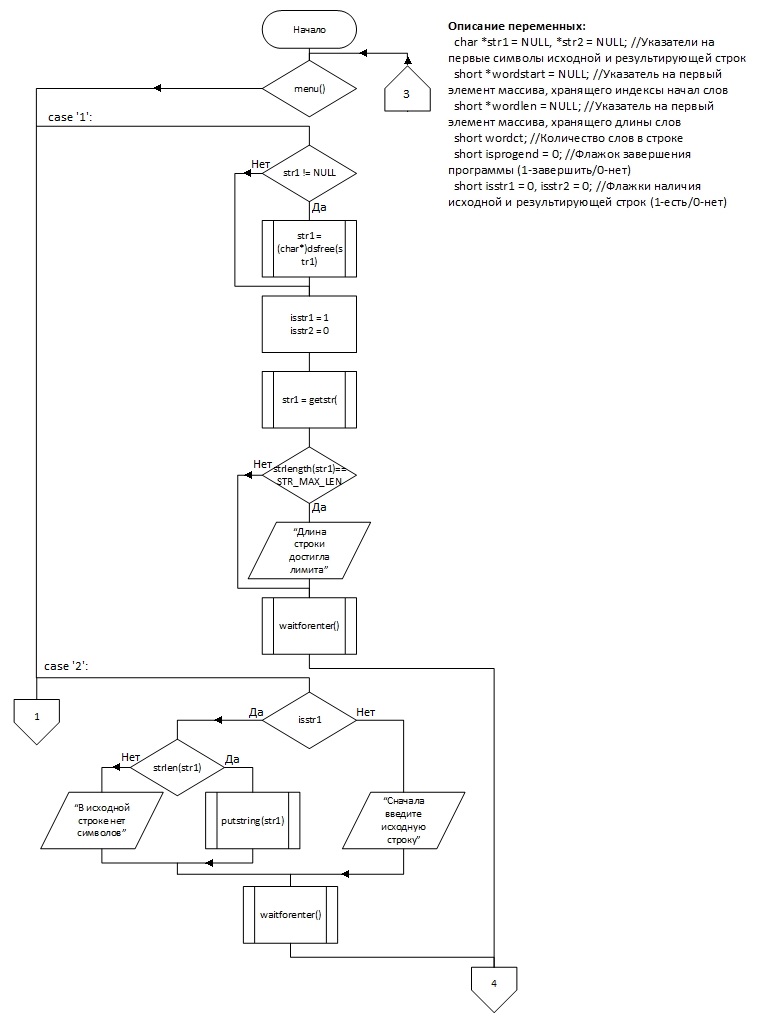
# 10.Результат

При выполнении программы получены результаты, совпадающие с контрольными примерами, представленными в таблице 1. Ошибки не обнаружены.

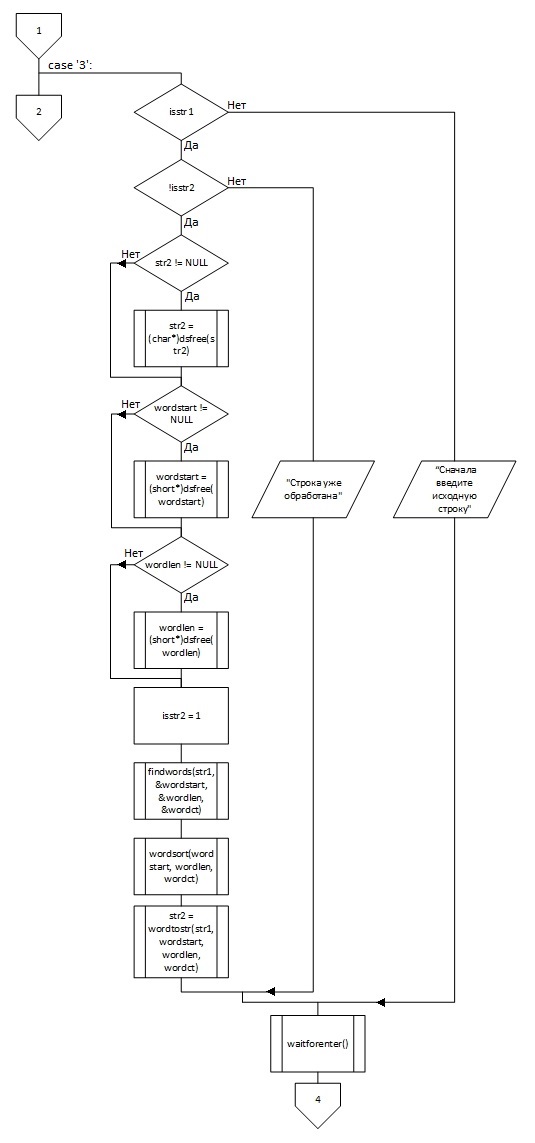
# Вывод

В ходе выполнения работы были получены практические навыки работы с строками в языке С / C++.

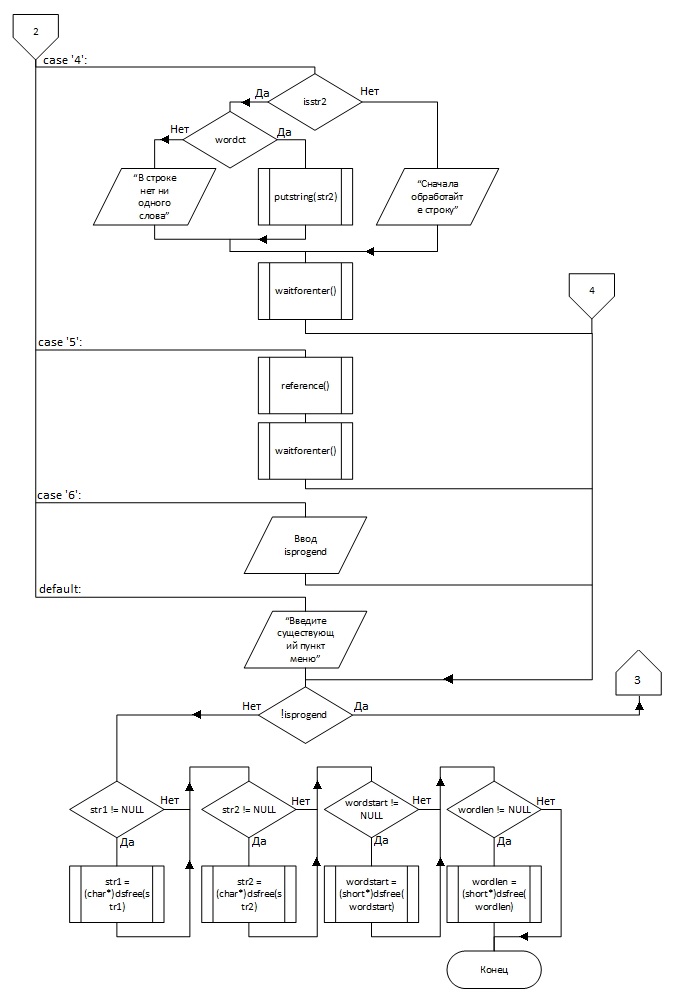
# *Приложение*



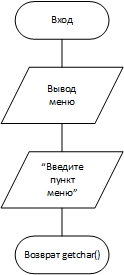
*Рисунок 1.1. Схема главной функции*

**

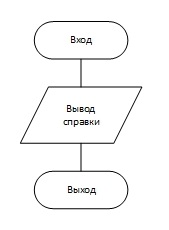
*Рисунок 1.2. Схема главной функции*

**

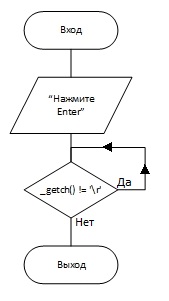
*Рисунок 1.3. Схема главной функции*

**

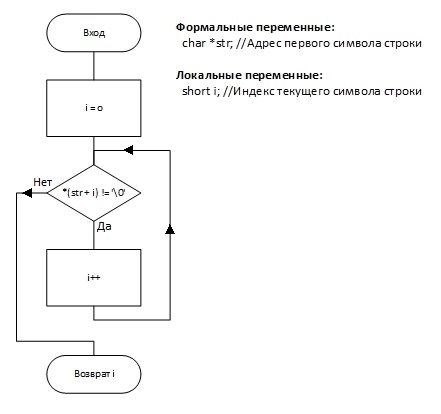
*Рисунок 2. Схема функции menu*

**

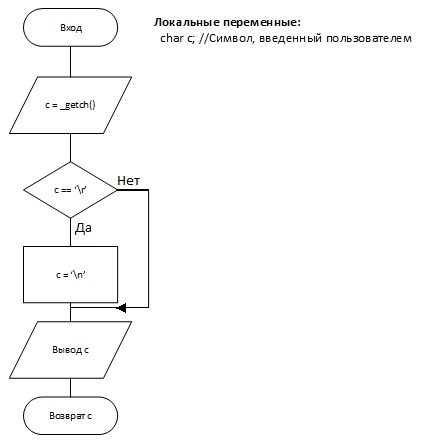
*Рисунок 3. Схема функции reference*

**

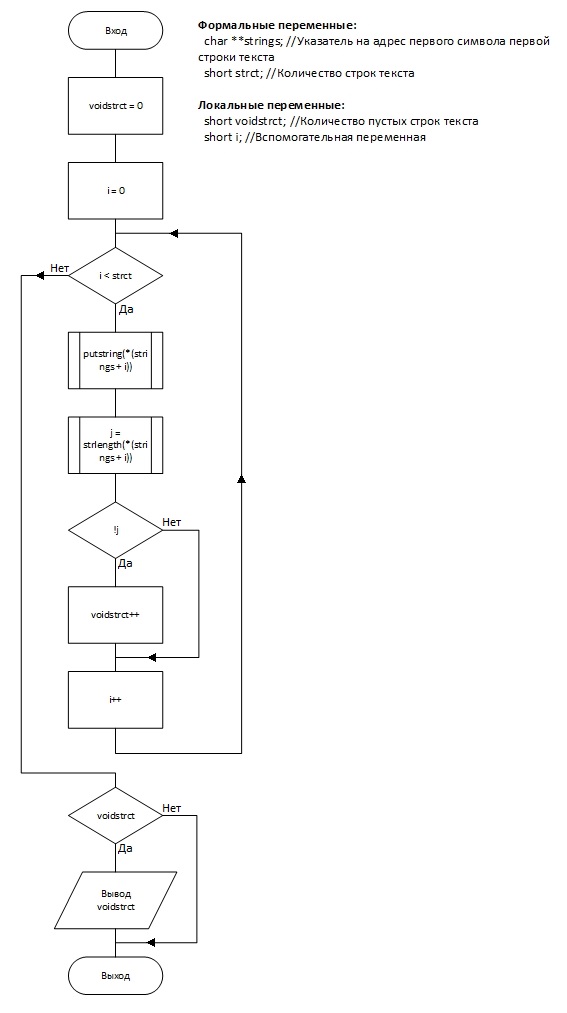
*Рисунок 4. Схема функции waitforenter*

**

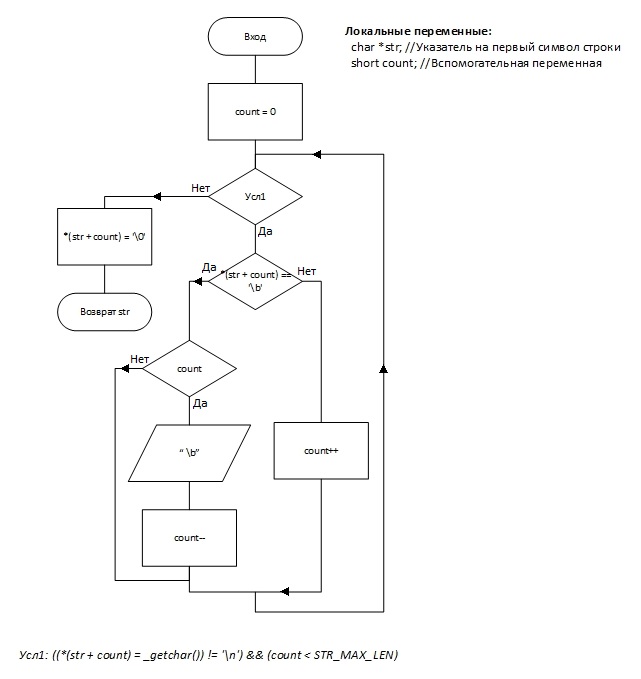
*Рисунок 5. Схема функции strlength*

**

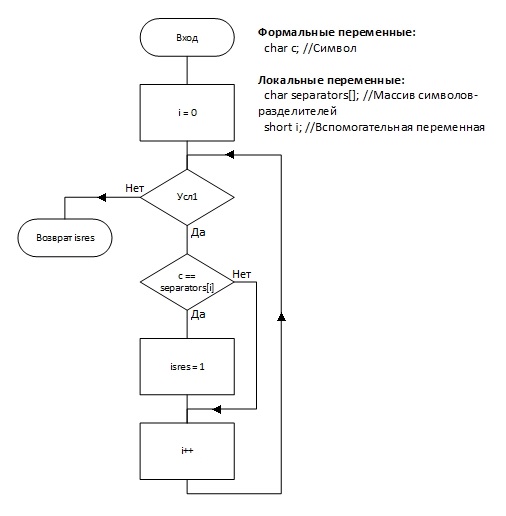
*Рисунок 6. Схема функции \_getchar*

**

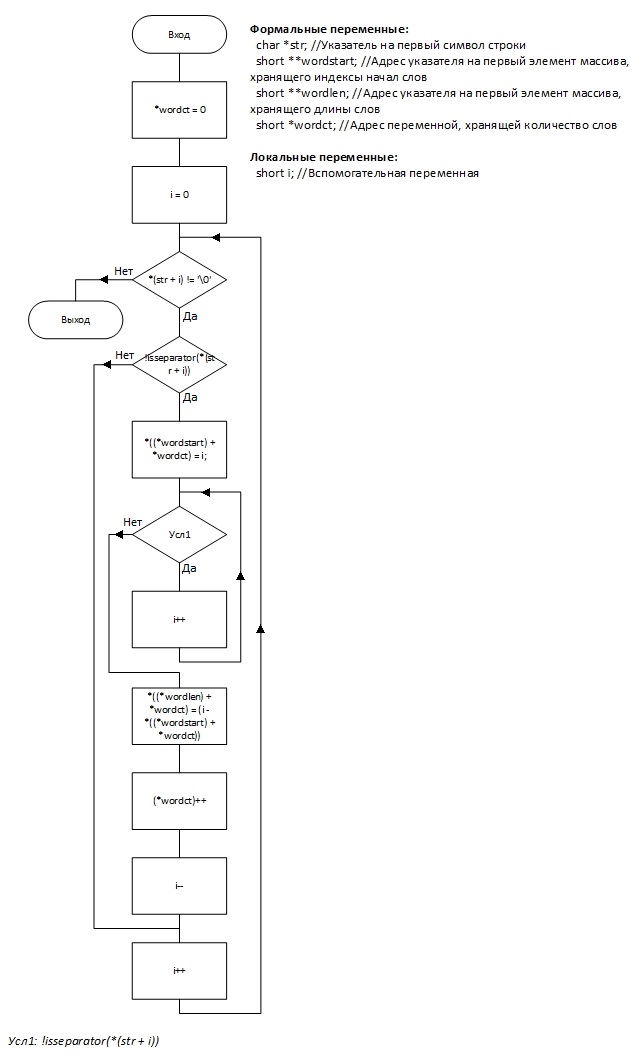
*Рисунок 7. Схема функции putstring*

**

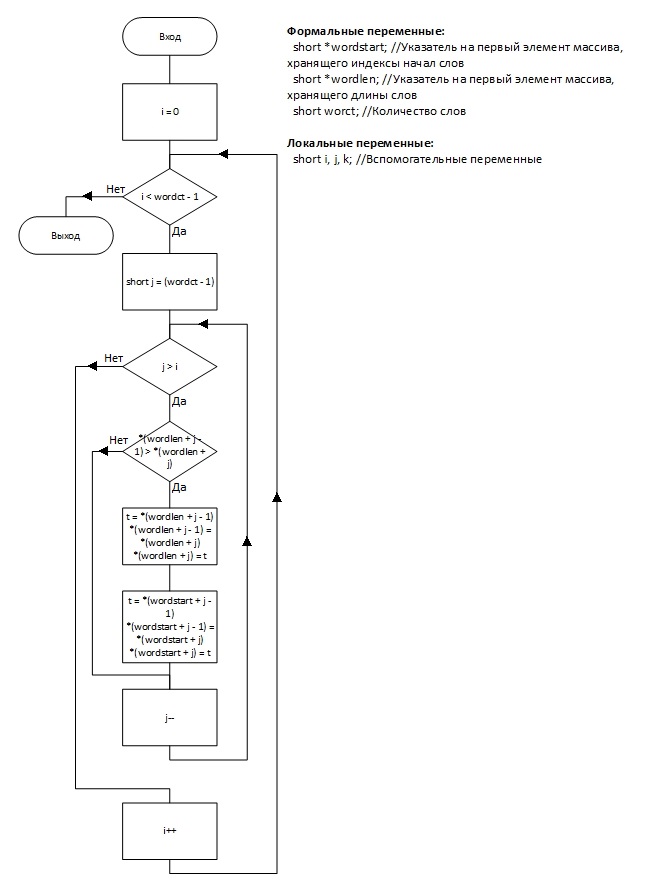
*Рисунок 8. Схема функции getstr*

**

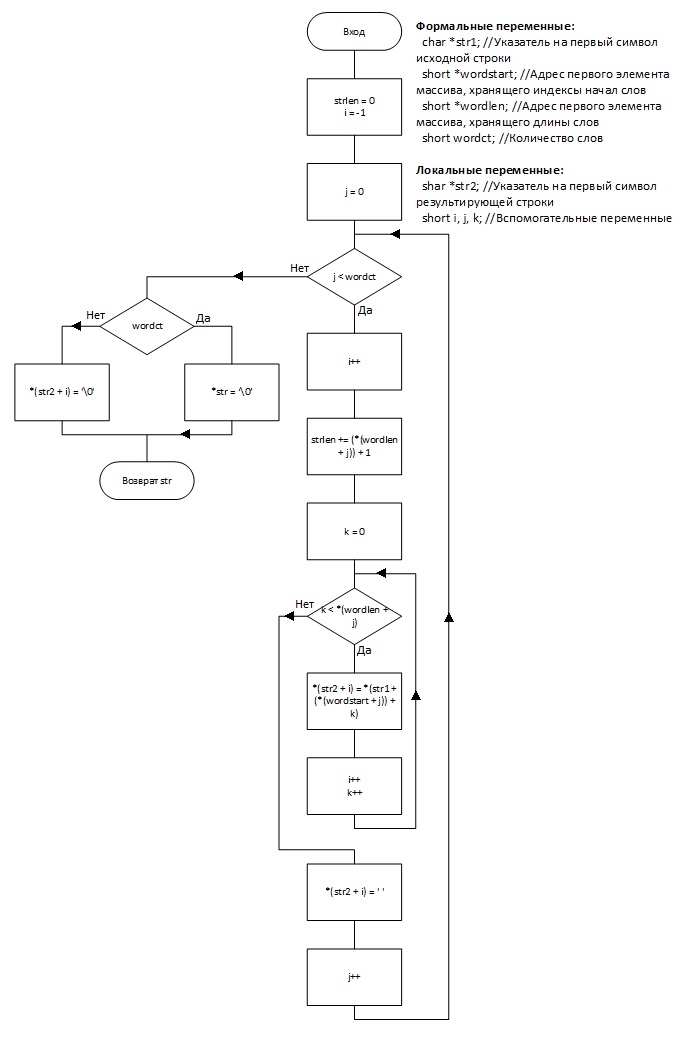
*Рисунок 9. Схема функции isseparator*

**

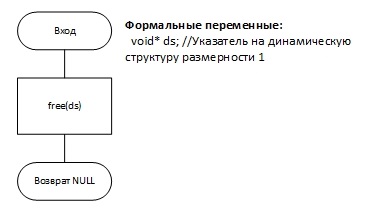
*Рисунок 10. Схема функции findwords*

**

*Рисунок 11. Схема функции wordsort*

**

*Рисунок 12. Схема функции wordtostr*

**

*Рисунок 13. Схема функции dsfree*