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

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовой работе на тему**

**“Обработка текста в языке 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](#_Toc435563467)

[1.Задание 3](#_Toc435563468)

[2.Уточнение задания 3](#_Toc435563469)

[3.Контрольные примеры 3](#_Toc435563470)

[4.Описание переменных главной функции 4](#_Toc435563471)

[5.Схема главной функции 4](#_Toc435563472)

[6.Описание функций 5](#_Toc435563473)

[6.1.Функция menu 5](#_Toc435563474)

[6.2.Функция reference 5](#_Toc435563475)

[6.3.Функция waitforenter 5](#_Toc435563476)

[6.4.Функция strlength 5](#_Toc435563477)

[6.5.Функция \_getchar 6](#_Toc435563478)

[6.6.Функция putstring 7](#_Toc435563479)

[6.7.Функция getstr 7](#_Toc435563480)

[6.8.Функция isseparator 8](#_Toc435563481)

[6.9.Функция findwords 9](#_Toc435563482)

[6.10.Функция wordsort 10](#_Toc435563483)

[6.11.Функция wordtostr 11](#_Toc435563484)

[6.12.Функция dsfree 12](#_Toc435563485)

[6.13.Функция putstrings 13](#_Toc435563486)

[6.14.Функция wordproc 14](#_Toc435563487)

[6.15.Функция gettxt 15](#_Toc435563488)

[6.16.Функция isstrmatch 16](#_Toc435563489)

[7.Иерархия вызовов функций программы 18](#_Toc435563490)

[8.Краткое описание алгоритма 18](#_Toc435563491)

[9.Исходный код программы 19](#_Toc435563492)

[10.Тесты 26](#_Toc435563493)

[11.Результат 27](#_Toc435563494)

[Вывод 27](#_Toc435563495)

[*Приложение* 28](#_Toc435563496)

# Цель работы

Получить практические навыки работы с текстом в языке С / C++.

# 1.Задание

Задан текст, содержащий произвольное количество строк, в которых отдельные слова могут разделяться одним или несколькими пробелами и знаками пунктуации (перенос строки с одной на другую не используется). Сформировать новый текст, который является результатом следующего преобразования исходного текста: упорядочить слова в каждой строке в порядке возрастания длин слов и разделить их пробелами.

# 2.Уточнение задания

-Перед тем, как ввести текст, вводится строка-признак конца текста. Строка-признак конца текста не принадлежит тексту и обозначает прекращение его ввода.

-Длина строк текста и строки-признака конца текста не должна превышать 100 символов.

-Исходный текст должен содержать хотя бы одну строку.

-Если в строке текста есть пустые строки, то должно быть выведено сообщение о их наличии и количестве.

-Знаками пунктуации(разделителями) являются следующие символы:

\ \n . , ; : ' " ! ? ( - ) { } [ ] / | \ \ < > \_ "

# 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 Слова будут отсортированы  их длин возрастанию |
| 2 | 4444 22 333  1 7777777 55555 |  | 22 333 4444  1 55555 7777777 |
| 3 |  | Две пустые строки | Пустых строк в тексте: 2 |
| 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  6 7 8 9 10 11 12  13 14 15 16 |
| 6 | That&dont@belong%to#separators  Надеюсь, смысл программы  понятен. |  | That&dont@belong%to#separators  смысл Надеюсь программы  понятен |

# 4.Описание переменных главной функции

Описание переменных представлено в таблице 2.

*Таблица 2. Описание переменных главной функции*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| txt1, txt2 | char\*\* | Указатели на адреса первых символов первых строк исходного и результирующего текстов |
| strct | short | Количество строк исходного и результирующего текстов |
| isprogend | short | Флажок выхода из программы (1-выйти/0-нет) |
| istxt1, istxt2 | short | Флажки наличия исходного и результирующего текстов(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 waitforenter 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 фактических параметров функции* strlength

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

*Таблица 5. Описание локальных переменных функции* strlength

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

## 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 фактических параметров функции* putstring

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

*Таблица 9. Описание локальных переменных функции* putstring

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

## 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 фактических параметров функции* isseparator

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

*Таблица 13. 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) |

## 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 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 |

*Таблица 16. Описание локальных переменных функции* findwords

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

## 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 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 |

*Таблица 17. Описание локальных переменных функции* wordsort

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

## 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 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 |

*Таблица 20. Описание локальных переменных функции* strlength

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

## 6.12.Функция dsfree

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

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

**Прототип второй перегрузки функции:** void \*\*dsfree(void \*\*ds, short index1ct);

**Возврат функции:** NULL

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

**Описание формальных переменных второй перегрузки функции:** описание формальных переменных второй перегрузки функции dsfree представлено в таблице 22

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

**Пример вызова второй перегрузки функции:** txt1 = dsfree(str1);

**Описание фактических параметров первой перегрузки функции:** Описание фактических параметров первой перегрузки функции dsfree представлено в таблице 23

**Описание фактических параметров второй перегрузки функции:** Описание фактических параметров второй перегрузки функции dsfree представлено в таблице 24

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

**Схема второй перегрузки функции функции:** схема второй перегрузки функции dsfree представлена на рисунке 14 в Приложении

*Таблица 21. Описание формальных переменных первой перегрузки функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| ds | void\* | Указатель на первый элемент динамической структуры |

*Таблица 22. Описание формальных переменных второй перегрузки функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| ds | void\*\* | Указатель на адрес первой переменной динамической структуры |
| index1ct | short | Количество первых индексов динамической структуры |

*Таблица 23. Описание фактических параметров первой перегрузки функции dsfree*

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

*Таблица 24. Описание фактических параметров второй перегрузки функции dsfree*

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| txt1 | char\*\* | Указатель на первый символ первой строки текста |

## 6.13.Функция putstrings

**Назначение функции:** вывод текста на экран

**Прототип функции:** void putstrings(char \*\*strings, short strct);

**Описание формальных переменных функции:** описание формальных переменных функции putstrings представлено в таблице 25

**Пример вызова функции:** putstrings(txt1, str1ct);

**Описание фактических параметров функции:** описание фактических параметров функции putstrings представлено в таблице 26

**Описание локальных переменных функции:** описание локальных переменных функции putstrings представлено в таблице 27

**Схема функции:** схема функции putstrings представлена на рисунке 15 в Приложении

*Таблица 25. Описание формальных переменных функции* putstrings

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| strings | char\*\* | Указатель на адрес первого символа первой строки текста |
| strct | short | Количество строк текста |

*Таблица 26. Описание фактических параметров функции* putstrings

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| txt1 | char\*\* | Указатель на адрес первого элемента первой строки исходного текста |
| str1ct | short | Количество строк исходного текста |

*Таблица 27. Описание локальных переменных функции* putstrings

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| voidstrct | short | Количество пустых строк в тексте |
| i | short | Вспомогательная переменная |

## 6.14.Функция wordproc

**Назначение функции:** сортировка слов в строках текста

**Прототип функции:** char \*\*wordproc(char \*\*txt1, short strct);

**Возврат функции:** указатель на первый символ первой строки результирующего текста

**Описание формальных переменных функции:** описание формальных переменных функции wordproc представлено в таблице 28

**Пример вызова функции:** txt2 = wordproc(txt1, strct);

**Описание фактических параметров функции:** описание фактических параметров функции wordproc представлено в таблице 29

**Описание локальных переменных функции:** описание локальных переменных функции wordproc представлено в таблице 30

**Схема функции:** схема функции wordproc представлена на рисунке 16 в Приложении

*Таблица 28. Описание формальных переменных функции* wordproc

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| txt1 | char\*\* | Указатель на адрес первого символа первой строки исходного текста |
| strct | short | Количество строк исходного и результирующего текстов |

*Таблица 29. Описание фактических параметров функции* wordproc

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| txt1 | char\*\* | Указатель на адрес первого символа первой строки исходного текста |
| strct | short | Количество строк исходного и результирующего текстов |

*Таблица 30. Описание локальных переменных функции* wordproc

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| txt2 | char\*\* | Указатель на адрес первого символа первой строки исходного текста |
| wordstart | short\* | Адрес первого элемента массива, хранящего индексы начал слов строки |
| wordlen | short\* | Адрес первого элемента массива, хранящего длины слов строки |
| wordct | short | Количество слов в строке |
| i | short | Вспомогательная переменная |

## 6.15.Функция gettxt

**Назначение функции:** ввод текста

**Прототип функции:** char \*\*gettxt(short \*strct);

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

**Описание формальных переменных функции:** описание формальных переменных функции gettxt представлено в таблице 31

**Пример вызова функции:** txt1 = gettxt(&strct);

**Описание фактических параметров функции:** описание фактических параметров функции gettxt представлено в таблице 32

**Описание локальных переменных функции:** описание локальных переменных функции gettxt представлено в таблице 33

**Схема функции:** схема функции gettxt представлена на рисунке 17 в Приложении

*Таблица 31. Описание формальных переменных функции* gettxt

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| strct | short\* | Адрес переменной, хранящей количество строк текста |

*Таблица 32. Описание фактических параметров функции* gettxt

|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| &strct | short\* | Адрес переменной, хранящей количество строк текста |

*Таблица 33. Описание локальных переменных функции* gettxt

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| txt | char\*\* | Указатель на адрес первого символа первой строки текста |
| regex | char\* | Указатель на первый символ строки-признака конца текста |
| isbreak | short | Флажок выхода из цикла(1-выйти/0-нет) |

## 6.16.Функция isstrmatch

**Назначение функции:** проверка, совпадают ли 2 строки

**Прототип функции:** short isstrmatch(char \*str, char \*regex);

**Возврат функции:** 1-если строки совпадают/0-если нет

**Описание формальных переменных функции:** описание формальных переменных функции isstrmatch представлено в таблице 34

**Пример вызова функции:** bool istrue = isstrmatch(str1, str2);

**Описание фактических параметров функции:** описание фактических параметров функции isstrmatch представлено в таблице 35

**Описание локальных переменных функции:** описание локальных переменных функции isstrmatch представлено в таблице 36

**Схема функции:** схема функции isstrmatch представлена на рисунке 18 в Приложении

*Таблица 34. Описание формальных переменных функции* isstrmatch

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| str, regex | char\* | Указатели на первые символы первой и второй строк |

*Таблица 35. Описание фактических параметров функции* isstrmatch

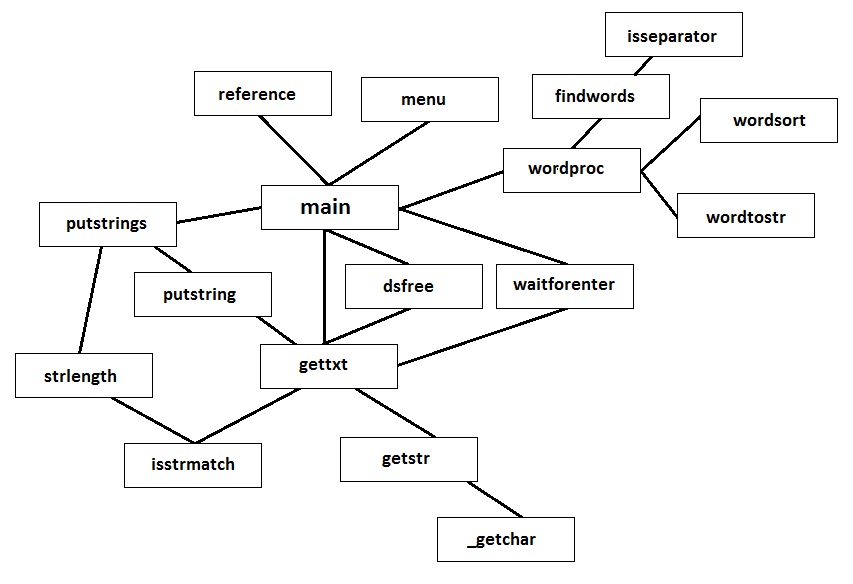
|  |  |  |
| --- | --- | --- |
| **Фактический параметр** | **Тип** | **Назначение** |
| str1, str2 | char\* | Указатели на первые символы первой и второй строк |

*Таблица 36. Описание локальных переменных функции* isstrmatch

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Назначение** |
| isres | short | Флажок результата (1-строки совпадают/0-нет) |
| i | short | Вспомогательная переменная |

# 7.Иерархия вызовов функций программы

Иерархия вызовов функций программы представлена на рисунке 19



*Рисунок 19. 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 текста с помощью функции gettxt. Переход в пункт 2

2)Вывод исходного текста с помощью функции putstrings или сообщение о том, что его нет. Переход в пункт 2

3)Обработка исходного текста с помощью функции wordproc или сообщение о том, что нет исходного текста, или сообщение о том, что результирующий текст уже сформирован. Переход в пункт 2

4)Вывод результирующего текста с помощью функции putstrings или сообщение о том, что его 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"

#define STR\_MAX\_LEN 100

//Ввод символа в режиме реального времени, вывод его на экран и возврат его значения

int \_getchar();

//Ввод строки и возврат указателя на ее первый элемент

char \*getstr();

//Ввод текста и возврат указателя на адрес его первого элемента

char \*\*gettxt(short \*strct);

//Ожидание действия пользователя

void waitforenter();

//Вывод меню на экран и возврат символа, введенного пользователем

char menu();

//Вывод справки на экран

void reference();

//Вывод строки на экран

void putstring(char \*str);

//Вывод текста на экран

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

//Проверка, является ли символ разделителем

short isseparator(char c);

//Проверка строк на совпадение

short isstrmatch(char \*str, char \*regex);

//Поиск длины строки

short strlength(char \*str);

//Поиск слов в строке

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

//Сортировка слов по возрастанию

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

//Формирование строки по заданым началам и длиннам слов исходной строки и возврат указателя на первый символ строки

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

//Сортировка слов строк текста по возрастанию их длин и возврат указателя на адрес первого элемента результирующего текста

char \*\*wordproc(char \*\*txt1, short strct);

//Освобождение памяти динамической структуры и возврат NULL

void \*dsfree(void \*ds);

void \*\*dsfree(void \*\*ds, short index1size);

int main()

{

char \*\*txt1 = NULL, \*\*txt2 = NULL;

short isprogend = 0, istxt1 = 0, istxt2 = 0, strct;

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

system("title = Курсовая работа. Обработка текста");

do

{

system("cls");

switch (menu())

{

case '1': //ввод текста

if (txt1 != NULL)

txt1 = (char\*\*)dsfree((void\*\*)txt1, strct);

if (txt2 != NULL)

txt2 = (char\*\*)(dsfree((void\*\*)txt2, strct));

istxt1 = 1;

istxt2 = 0;

system("cls");

rewind(stdin);

txt1 = gettxt(&strct);

printf\_s("\nВвод завершен!\n");

waitforenter();

break;

case '2': //вывод исходного текста

system("cls");

if (istxt1)

{

printf\_s("Исходный текст:\n");

putstrings(txt1, strct);

}

else

printf\_s("Ошибка! Сначала введите исходный текст");

printf\_s("\n");

waitforenter();

break;

case '3': //обработка исходного текста

system("cls");

if (istxt1)

{

if (txt2 == NULL)

{

istxt2 = 1;

txt2 = wordproc(txt1, strct);

printf\_s("Текст обработан");

}

else

printf\_s("Текст уже был обработан");

}

else

printf\_s("Ошибка! Сначала введите исходный текст");

printf\_s("\n");

waitforenter();

break;

case '4': //вывод текста

system("cls");

if (istxt2)

{

printf\_s("Результирующий текст:\n");

putstrings(txt2, strct);

}

else

printf\_s("Ошибка! Сначала обработайте исходный текст");

printf\_s("\n");

waitforenter();

break;

case '5': //справка

system("cls");

reference();

printf\_s("\n");

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");

waitforenter();

}

} while (!isprogend);

if (txt1 != NULL)

txt1 = (char\*\*)dsfree((void\*\*)txt1, strct);

if (txt1 != NULL)

txt1 = (char\*\*)dsfree((void\*\*)txt1, strct);

return 0;

}

//Описание: вывод меню на экран

//Возврат: символ, введенный пользователем

char menu()

{

printf\_s("==Меню==\n");

printf\_s("1 - Ввод текста\n");

printf\_s("2 - Вывод исходного текста\n");

printf\_s("3 - Обработка исходного текста\n");

printf\_s("4 - Вывод результата\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("-Перед тем, как ввести текст, вводится строка-признак конца текста. Строка конца

текста не принадлежит тексту\n");

printf\_s("-Длина строк текста и строки-признака конца текста не превышает %d\n",

STR\_MAX\_LEN);

printf\_s("-Исходный текст должен содержать хотя бы одну строку\n");

printf\_s("-Обработанный текс соделржит слова строк исходного, упорядоченные по

возрастанию и разделенные пробелом.\n");

printf\_s("-Если в строке текста есть пустые строки, то выводится сообщение о их

наличии и количестве\n");

printf\_s("-Разделителями являются следующие символы: \"

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

}

//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 длины строки

//Возврат: длина строки

short strlength(char \*str)

{

short i;

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

return (i);

}

//Описание: ввод символа в режиме раельного времени и вы вод его на экран

//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 символ 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 их длин

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;

}

//Описание: сортировка слов в строках текста

//Возврат: указатель на адрес первого символа первой строки результирующего текста

char \*\*wordproc(char \*\*txt1, short strct)

{

char \*\*txt2;

short \*wordstart = NULL, \*wordlen = NULL, wordct;

txt2 = (char\*\*)malloc(sizeof(char\*) \* strct);

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

{

findwords(\*(txt1 + i), &wordstart, &wordlen, &wordct);

wordsort(wordstart, wordlen, wordct);

\*(txt2 + i) = wordtostr(\*(txt1 + i), wordstart, wordlen, wordct);

}

return txt2;

}

//Описание: ввод текста

//Возврат: указатель на адрес первого символа первой строки текста

char \*\*gettxt(short \*strct)

{

char \*\*txt, \*regex = NULL;

short isbreak;

//ввод строки-признака конца текста

system("cls");

do

{

if (regex != NULL)

dsfree(regex);

printf\_s("Введите строку конца текста: ");

rewind(stdin);

regex = getstr();

if (!strlength(regex))

printf\_s("Ошибка! Строка-признак конца текста должна состоять хотя бы из

одного символа\n");

else

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

{

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

waitforenter();

}

} while (!strlength(regex));

//ввод текста

do

{

isbreak = 0;

txt = NULL;

system("cls");

printf\_s("Строка-признак конца текста:\n");

putstring(regex);

printf\_s("\nВведите текст:\n");

for (\*strct = 0; !isbreak; (\*strct)++)

{

txt = (char\*\*)realloc(txt, sizeof(char\*) \* ((\*strct) + 1));

rewind(stdin);

\*(txt + \*strct) = getstr();

isbreak = isstrmatch(\*(txt + \*strct), regex);

if ((strlength(\*(txt + \*strct)) == STR\_MAX\_LEN) && !isbreak)

printf\_s("\nДлина строки достигла лимита. Ввод прекращен.

Введите следующую строку:\n");

}

if (!(--(\*strct)))

{

printf\_s("\nОшибка! В тексте должна быть хотя бы одна строка\n");

waitforenter();

}

} while (!(\*strct));

return txt;

}

//Описание: проверка на совпадение двух строк

//Возврат: (1/0)

short isstrmatch(char \*str, char \*regex)

{

short isres = 1;

if (strlength(str) == strlength(regex))

{

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

if (\*(str + i) != \*(regex + i))

isres = 0;

}

else

{

isres = 0;

}

return isres;

}

# 10.Тесты

**Тест №1:**

Проверка программы на корректное выполнение задачи с помощью тестов, приведенных в контрольных примерах в таблице 1 Контрольные примеры.

**Тест №2:**

Ситуация: во время ввода текста зажать клавишу любого символа(например ‘р’).

Ожидаемая реакция программы: сообщение о том, что строка достигла максимального размера и переход на ввод следующей строки.

**Тест №3:**

Ситуация: во время ввода строки-признака конца текста, нажать Enter не введя никаких символов.

Ожидаемая реакция программы: сообщение о том, что строка-признак конца текста должна состоять хотя бы из одного символа и повторный ввод строки-признака конца текста.

**Тест №4:**

Ситуация: во время ввода текста сразу ввести строку конца текста.

Ожидаемая реакция программы: сообщение о том, что текст должен состоять хотя бы из одной строки и повторный ввод текста.

**Тест №5:**

Ситуация: во время ввода текста нажать клавишу backspace.

Ожидаемая реакция программы: удаление символа перед курсором.

**Тест №6:**

Ситуация: во время ввода строки-признака конца текста, зажать клавишу любого символа(например ‘p’).

Ожидаемая реакция программы: Прекращение ввода строки-признака конца текста, сообщение о том, что ее длина достигла предела и переход к вводу текста.

**Тест №7:**

Ситуация: не производя обработки исходного текста, выбрать пункт вывода результирующего текста на экран.

Ожидаемая реакция программы: сообщение о том, что текст еще не обработан.

**Тест №8:**

Ситуация: обработка уже обработанного текста.

Ожидаемая реакция программы: сообщение о том, что текст уже обработан.

**Тест №9:**

Ситуация:не совершая ввод исходного текста выбрать пункт вывода исходного текста на экран. Ожидаемая реакция программы: сообщение о том, что еще не введен исходный текст.

**Тест №10:**

Ситуация: не совершая ввод исходного текста, выбрать пункт его обработки.

Ожидаемая реакция программы: сообщение о том, что сначала надо ввести исходный текст.

**Тест №11:**

Ситуация: во время ввода пункта меню, ввести несуществующий пункт(например ‘8’).

Ожидаемая реакция программы: сообщение о том, что введен некорректный пункт меню.

**Тест №12:**

Ситуация: во время подтверждения выхода из программы, выбрать нет(0).

Ожидаемая реакция программы: возврат в меню.

**Тест №13:**

Ситуация: подтверждение выхода из программы.

Ожидаемая реакция программы: выход из программы.

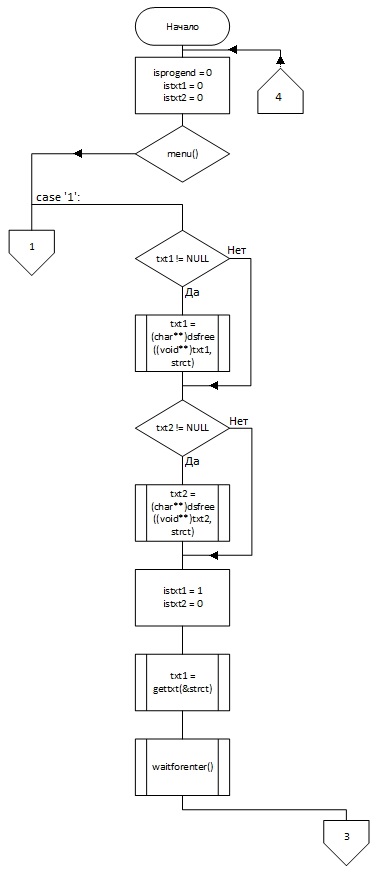
# 11.Результат

При проверке программы с помощью тестов, приведенных в пункте 10 Тесты, ошибки не обнаружены. Некоторые результаты приведены на рисунках():

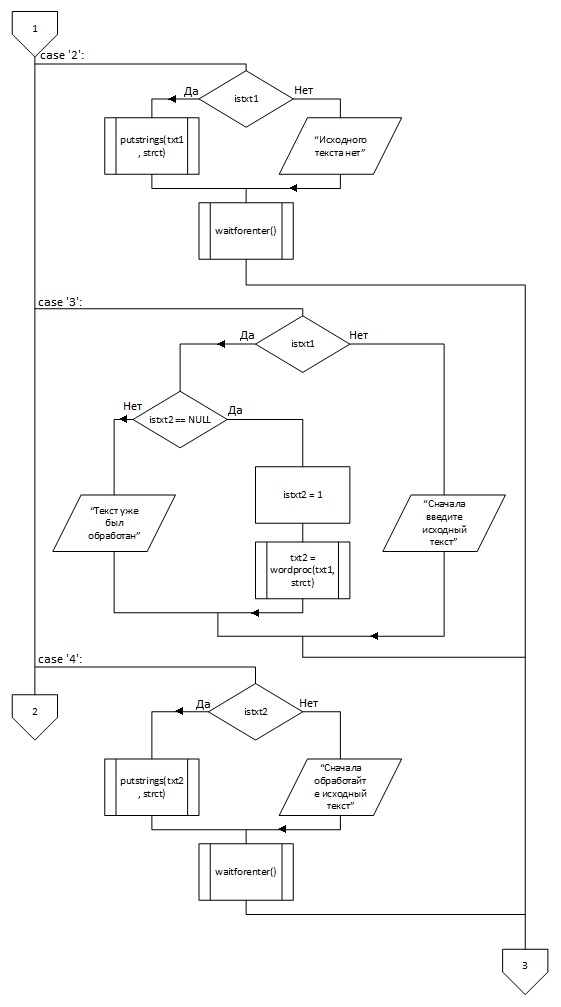
# Вывод

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

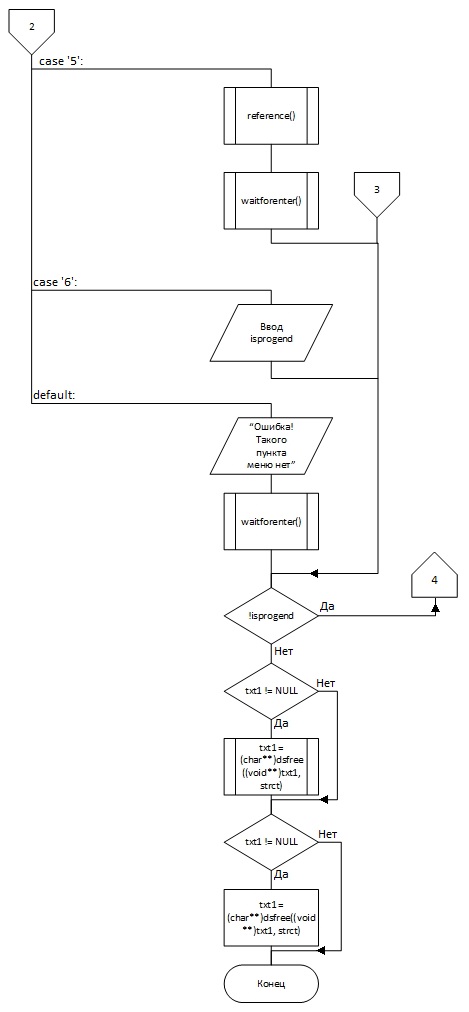
# *Приложение*



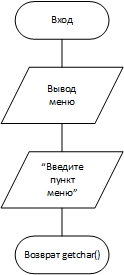
*Рисунок 1.1. Схема главной функции*

**

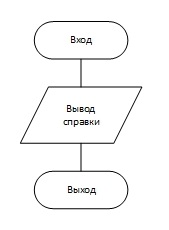
*Рисунок 1.2. Схема главной функции*

**

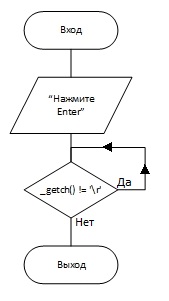
*Рисунок 1.3. Схема главной функции*

**

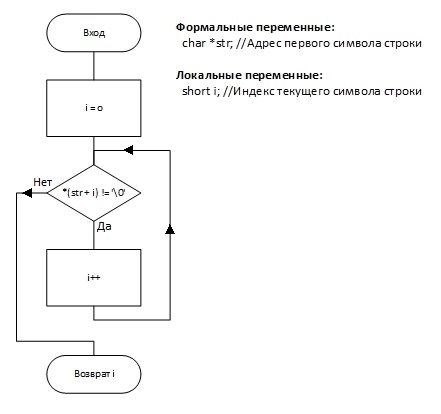
*Рисунок 2. Схема функции menu*

**

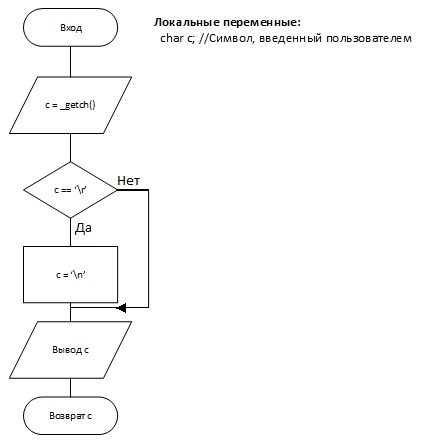
*Рисунок 3. Схема функции reference*

**

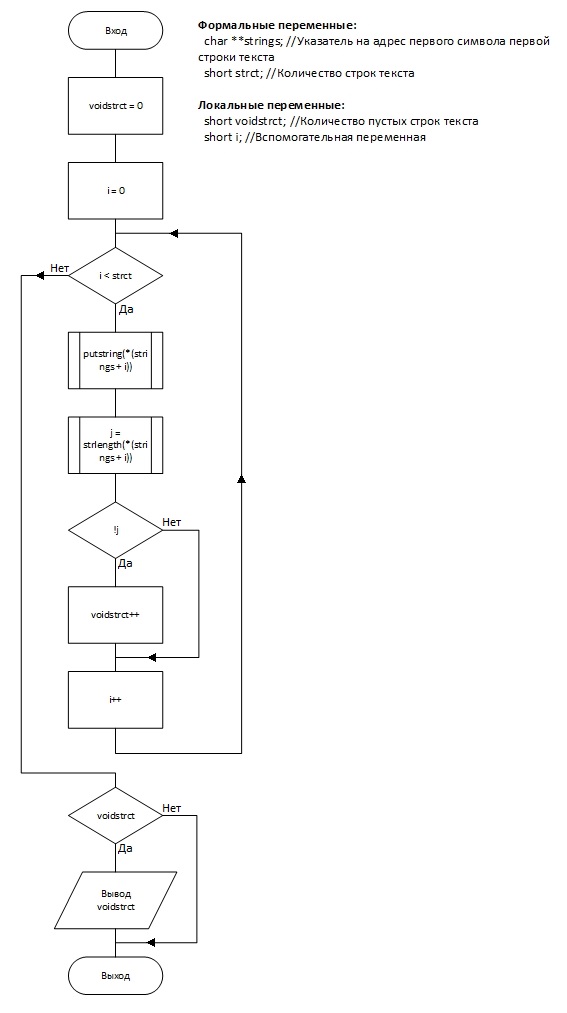
*Рисунок 4. Схема функции waitforenter*

**

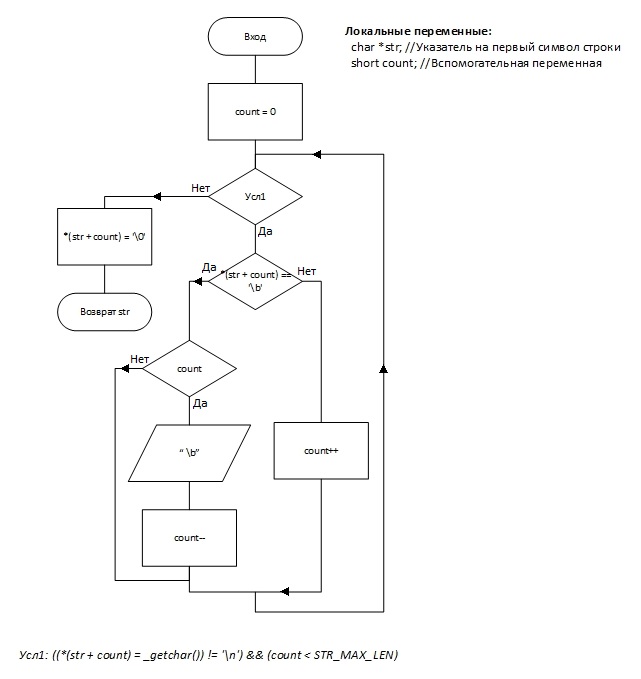
*Рисунок 5. Схема функции strlength*

**

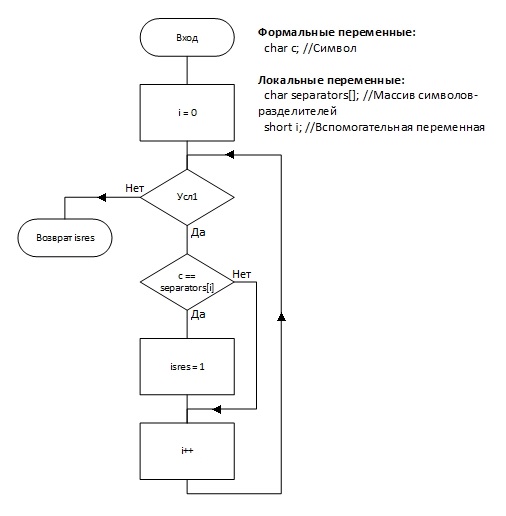
*Рисунок 6. Схема функции \_getchar*

**

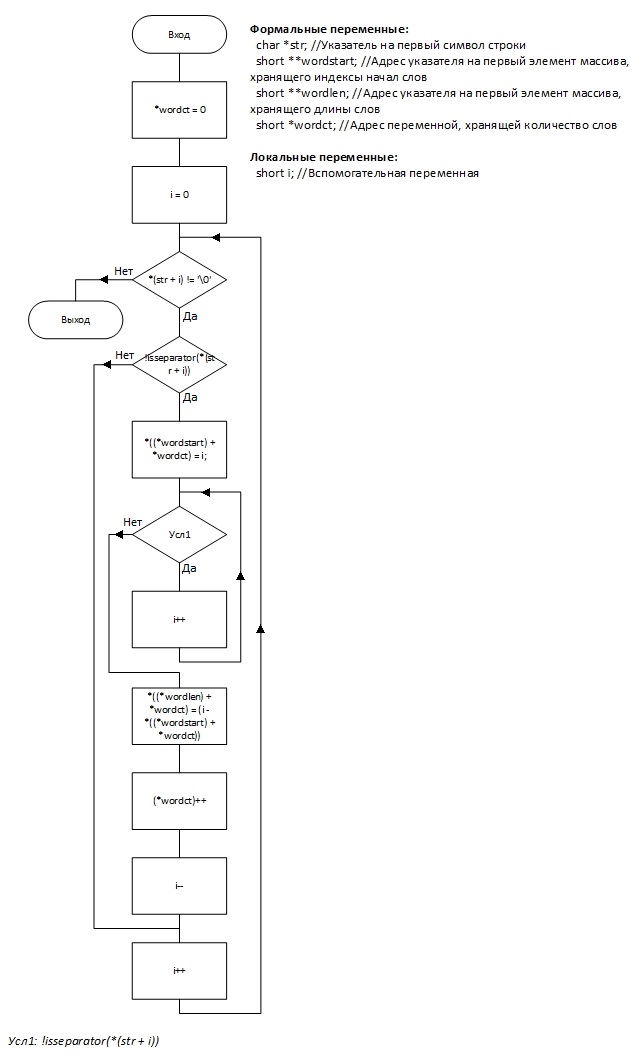
*Рисунок 7. Схема функции putstring*

**

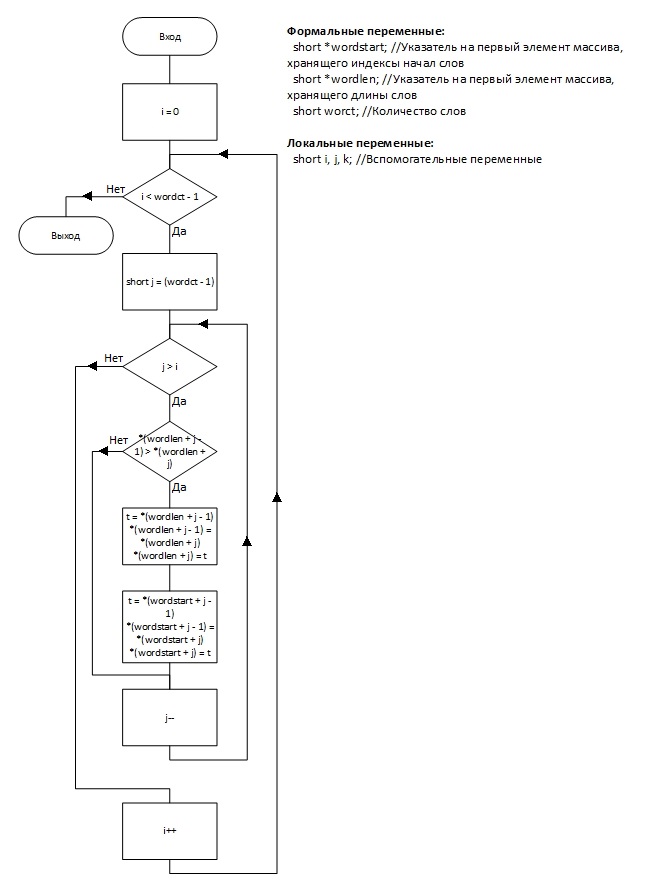
*Рисунок 8. Схема функции getstr*

**

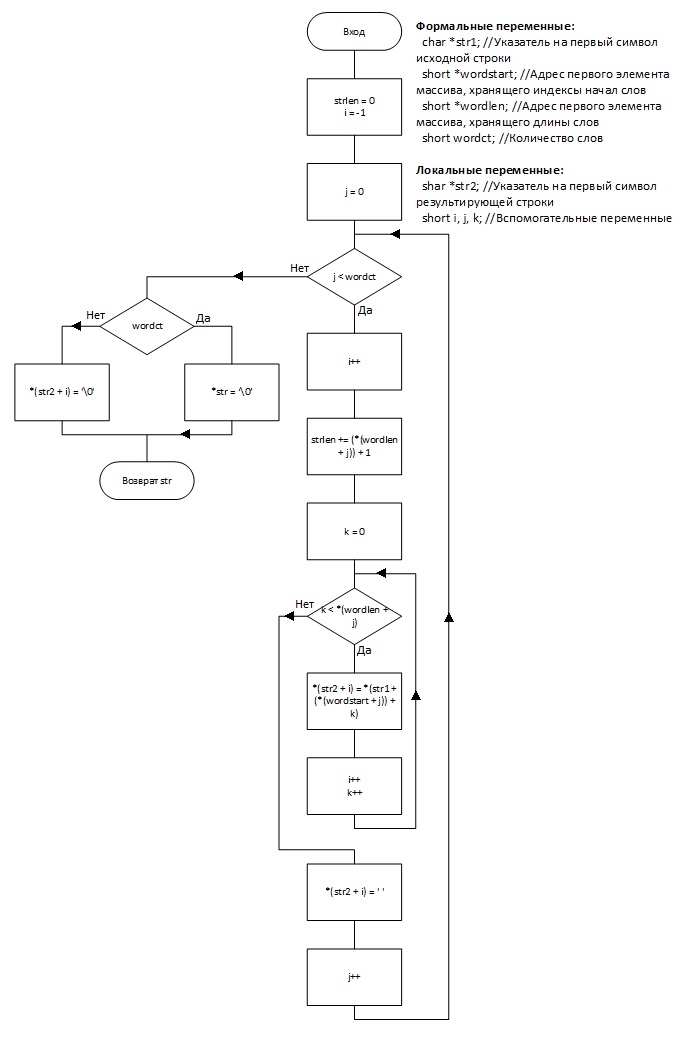
*Рисунок 9. Схема функции isseparator*

**

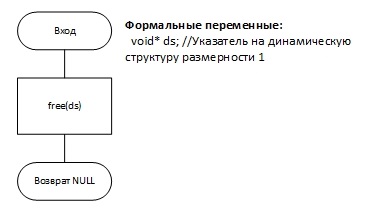
*Рисунок 10. Схема функции findwords*

**

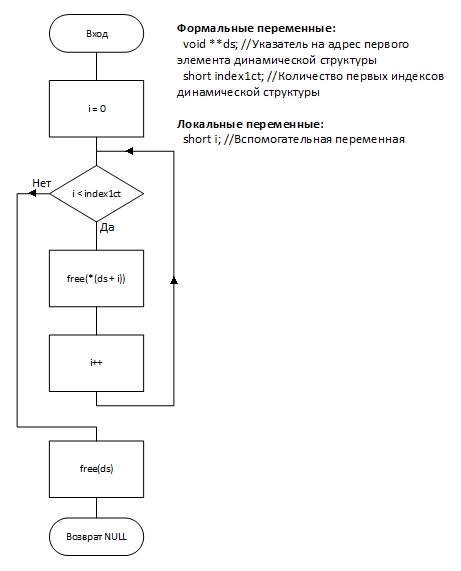
*Рисунок 11. Схема функции wordsort*

**

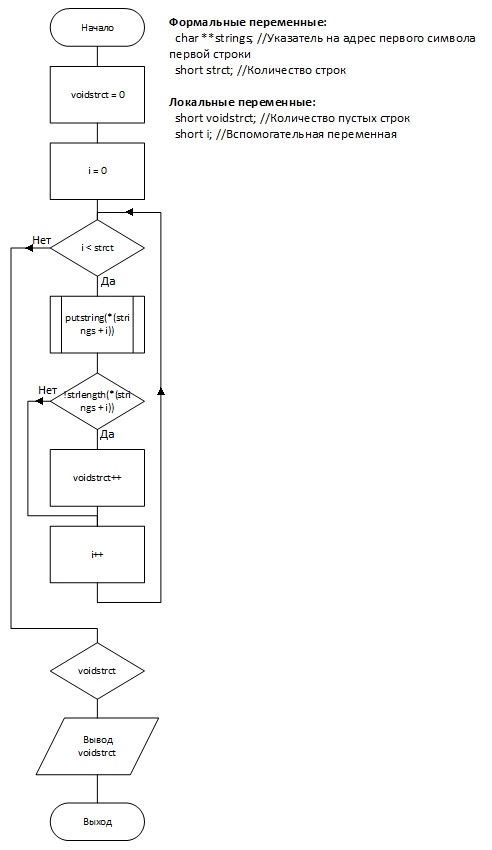
*Рисунок 12. Схема функции wordtostr*

**

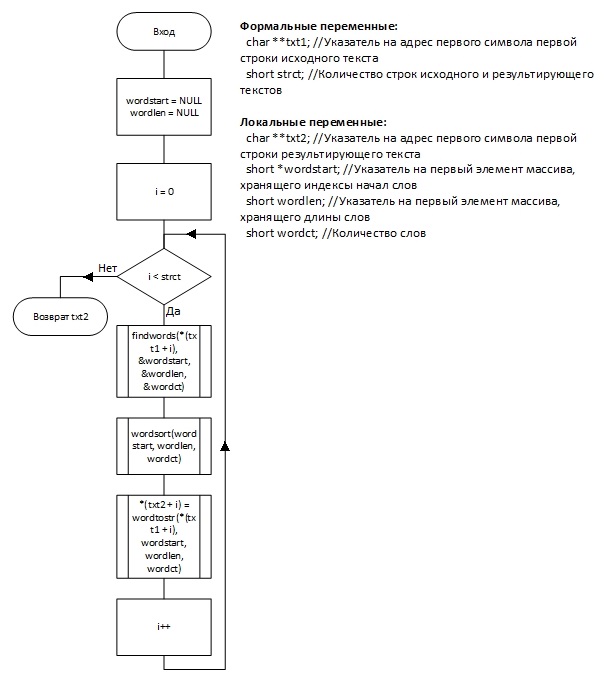
*Рисунок 13. Схема первой перегрузки функции dsfree*

**

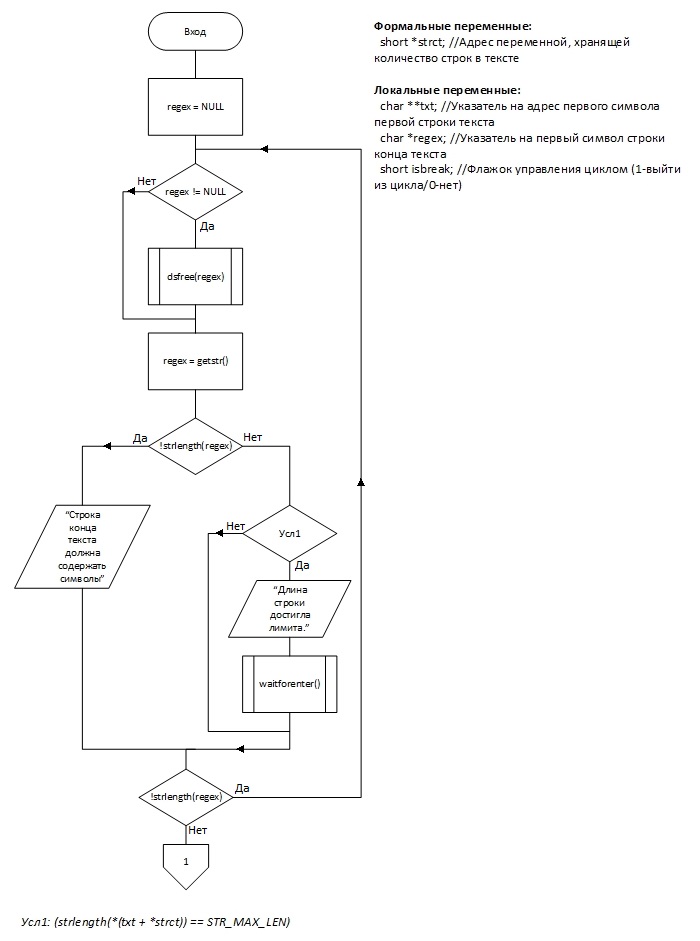
*Рисунок 14. Схема второй перегрузки функции dsfree*

**

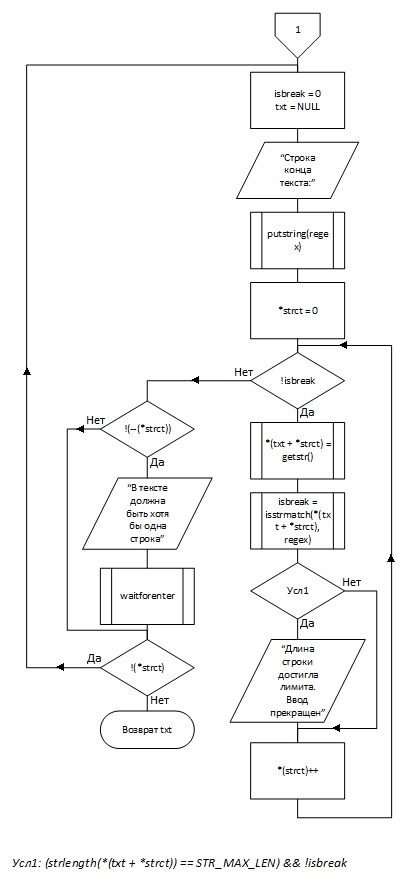
*Рисунок 15. Схема функции putstrings*

**

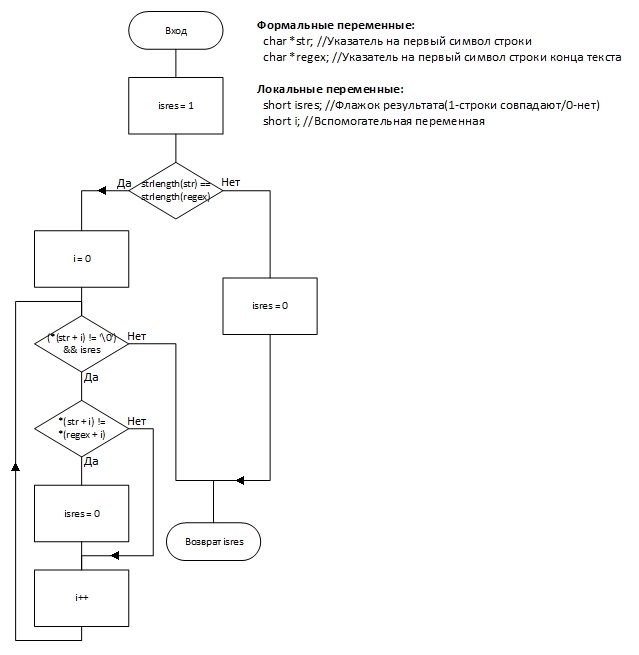
*Рисунок 15. Схема функции wordproc*

**

*Рисунок 16.1. Схема функции gettxt*

**

*Рисунок 16.2. Схема функции gettxt*

**

*Рисунок 17. Схема функции isstrmatch*