**МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное**

**учреждение высшего образования**

**«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»**

**(РУТ(МИИТ)**

Кафедра «Вычислительные системы, сети и информационная безопасность»

**Отчет По Дисциплине**

**«Алгоритмизация и программирование»**

**Лабораторная работа №9**

*Направление:* 10.03.01*Информационная безопасность*

*Профиль:**Безопасность компьютерных систем*

Выполнил:  
студент группы УИБ-112

Орлов Андрей Витальевич

Проверил:

Старший преподаватель Никольская Марина Николаевна

(должность, ФИО)

Старший преподаватель Цыганова Наталия Алексеевна

(должность, ФИО)

Москва 2021 г.

**Задание №9**

Задание: задан текст. Между словами пробел, за последним словом точка. Определить порядковый номер слова заданного текста.

1. Таблица имён:

|  |  |  |
| --- | --- | --- |
| Исходные данные | | |
| main\_text | Символьный динамический массив | Исходный текст, заданный пользователем |
| main\_word | Символьный динамический массив | Слово, заданное пользователем, которое будет искаться в заданном тексте. |
| Рабочие переменные | | |
| len | Целочисленный | Переменная для увеличения размерности символьных динамических массивов со считыванием каждого символа с консоли. |
| c | Символьный | Переменная для последовательного считывания символов с консоли |
| something | Символьный динамический массив | Переменная для обработки исходных данных типа символьных массивов с замещением заглавных букв на строчные |
| something[i] | Символьный | Переменная для установления, является ли буква заглавной или строчной |
| s | Символьный динамический массив | Постепенно увеличивающийся массив символов, который увеличивается соответственно с вводимыми пользователем символами |
| fact | Целочисленный | Переменная-флаг, обозначающая обнаружение слова в полном соответствии в заданном тексте |
| sequence | Целочисленный | Переменная для сравнения слов на полное соответствие друг другу, анализирует только символы текста, но не слова. |
| position | Целочисленный | Местоположение вхождения первого символа найденного слова в тексте |
| i | Целочисленный | Переменная-счётчик в циклах |
| k | Целочисленный | Переменная-счётчик в циклах |
| Результат | | |
| counter | Целочисленный | Переменная-счётчик, отвечает за подсчёт пробелов перед найденным словом, таким образом находя порядковый номер слова. |

1. Блок-схема:

Diagram

Description automatically generated

1. Отладочный пример:

Вариант I

1. Начало  
38. Переприсваивание переменных counter = 0  
2. Вывод «Введите текст: »  
3. Переходим к выполнению подпрограммы read\_string():  
20. Работа с переменными:  
len = 1, выделяем память под s (1 байт), считываем введённым пользователем символ с консоли в переменную с = 'М'  
21. 'М' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (1 байт), s[0] = 'M', считываем в переменную с следующий символ с консоли 'А', len++ = 1+1=2  
21. 'A' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (2 байта), s[1] = 'A', считываем в переменную с следующий символ с консоли 'ш', len++ = 2+1=3  
21. 'ш' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (3 байта), s[2] = 'ш', считываем в переменную с следующий символ с консоли 'А', len++ = 3+1=4  
21. 'A' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (4 байта), s[3] = 'A', считываем в переменную с следующий символ с консоли 'ш', len++ = 4+1=5  
21. ' ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (5 байтов), s[4] = ' ', считываем в переменную с следующий символ с консоли 'в', len++ = 5+1=6  
21. 'в ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (6 байтов), s[5] = 'в', считываем в переменную с следующий символ с консоли 'Ы', len++ = 6+1=7  
21. 'Ы ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (7 байтов), s[6] = 'Ы', считываем в переменную с следующий символ с консоли 'ш', len++ = 7+1=8  
21. 'ш ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (8 байтов), s[7] = 'ш', считываем в переменную с следующий символ с консоли 'Л', len++ = 8+1=9  
21. 'Л ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (9 байтов), s[8] = 'Л', считываем в переменную с следующий символ с консоли 'а', len++ = 9+1=10  
21. 'а ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (10 байтов), s[9] = 'а', считываем в переменную с следующий символ с консоли ' ', len++ = 10+1=11  
21. ' ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (11 байтов), s[10] = ' ', считываем в переменную с следующий символ с консоли 'п', len++ = 11+1=12  
21. 'п' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (12 байтов), s[11] = 'п', считываем в переменную с следующий символ с консоли 'о', len++ = 12+1=13  
21. 'о ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (13 байтов), s[12] = 'о', считываем в переменную с следующий символ с консоли 'г', len++ = 13+1=14  
21. 'г' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (14 байтов), s[13] = 'г', считываем в переменную с следующий символ с консоли 'у', len++ = 14+1=15  
21. 'у' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (15 байтов), s[14] = 'у', считываем в переменную с следующий символ с консоли 'л', len++ = 15+1=16  
21. 'л' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (16 байтов), s[15] = 'л', считываем в переменную с следующий символ с консоли 'я', len++ = 16+1=17  
21. 'я' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (17 байтов), s[16] = 'я', считываем в переменную с следующий символ с консоли 'т', len++ = 17+1=18  
21. 'т' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (18 байтов), s[17] = 'т', считываем в переменную с следующий символ с консоли 'ь', len++ = 18+1=19  
21. 'ь' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (19 байтов), s[18] = 'ь', считываем в переменную с следующий символ с консоли '.', len++ = 19+1=20  
21. '.' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (20 байтов), s[19] = '.', считываем в переменную с следующий символ с консоли '\n',   
len++ = 20 + 1 = 21  
21. '\n' != '\n', нет, переход к блоку 23  
23. Переприсваивание переменных s[len - 1] = '\0'  
24. Возврат s из подпрограммы  
Переприсваивание переменных main\_text = “МАшА вЫшЛа погулять.”  
4. Вывод «Введите ваше слово: »  
5. Переходим к выполнению подпрограммы read\_string():  
20. Работа с переменными:  
len = 1, выделяем память под s (1 байт), считываем введённым пользователем символ с консоли в переменную с = 'в'  
21. ‘в’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (1 байт), s[0] = ‘в’, считываем в переменную с следующий символ с консоли ‘ы’, len++ = 1+1=2  
21. ‘ы’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (2 байта), s[1] = ‘ы’, считываем в переменную с следующий символ с консоли ‘ш’, len++ = 2+1=3  
21. ‘ш’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (3 байта), s[2] = ‘ш’, считываем в переменную с следующий символ с консоли ‘л’, len++ = 3+1=4  
21. ‘л’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (4 байта), s[3] = ‘л’, считываем в переменную с следующий символ с консоли ‘а’, len++ = 4+1=5  
21. ‘а’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (4 байта), s[4] = ‘а’, считываем в переменную с следующий символ с консоли ‘\n’, len++ = 5+1=6  
21. ‘\n’ != ‘\n’, нет, переход к блоку 23  
23. Переприсваивание переменных s[5] = ‘\0’  
24. Возврат s из подпрограммы  
Переприсваивание переменных main\_word = “вышла”  
6. Переходим к выполнению подпрограммы find(main\_text, main\_word)  
25. Переход к выполнению подпрограммы format(text)  
14. Вход в цикл: Переменная k = 0  
15. ‘M’ >= ‘A’ && ‘M’ <= ‘Z’, нет, переход к блоку 17  
17. ‘M’ >= ‘A’ && ‘M’ <= ‘Я’, да, переход к блоку 18  
18. Переприсваивание переменных something[0] = ‘M’ – ‘A’ + ‘a’ = ‘м’  
14. Вход в цикл: Переменная k = 1  
15. ‘А’ >= ‘A’ && ‘А’ <= ‘Z’, нет, переход к блоку 17  
17. ‘А’ >= ‘A’ && ‘А’ <= ‘Я’, да, переход к блоку 18  
18. Переприсваивание переменных something[1] = ‘А’ – ‘A’ + ‘a’ = ‘а’  
14. Вход в цикл: Переменная k = 2  
15. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 3  
15. ‘А’ >= ‘A’ && ‘А’ <= ‘Z’, нет, переход к блоку 17  
17. ‘А’ >= ‘A’ && ‘А’ <= ‘Я’, да, переход к блоку 18  
18. Переприсваивание переменных something[3] = ‘А’ – ‘A’ + ‘a’ = ‘а’  
14. Вход в цикл: Переменная k = 4  
15. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 5  
15. ‘в’ >= ‘A’ && ‘в’ <= ‘Z’, нет, переход к блоку 17  
17. ‘в’ >= ‘A’ && ‘в’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 6  
15. ‘Ы’ >= ‘A’ && ‘Ы’ <= ‘Z’, нет, переход к блоку 17  
17. ‘Ы’ >= ‘A’ && ‘Ы’ <= ‘Я’, да, переход к блоку 18  
18. Переприсваивание переменных something[6] = ‘Ы’ – ‘A’ + ‘a’ = ‘ы’  
14. Вход в цикл: Переменная k = 7  
15. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 8  
15. ‘Л’ >= ‘A’ && ‘Л’ <= ‘Z’, нет, переход к блоку 17  
17. ‘Л’ >= ‘A’ && ‘Л’ <= ‘Я’, да, переход к блоку 18  
18. Переприсваивание переменных something[8] = ‘Л’ – ‘A’ + ‘a’ = ‘л’  
14. Вход в цикл: Переменная k = 9  
15. ‘а’ >= ‘A’ && ‘а’ <= ‘Z’, нет, переход к блоку 17  
17. ‘а’ >= ‘A’ && ‘а’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 10  
15. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 11  
15. ‘п’ >= ‘A’ && ‘п’ <= ‘Z’, нет, переход к блоку 17  
17. ‘п’ >= ‘A’ && ‘п’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 12  
15. ‘о’ >= ‘A’ && ‘о’ <= ‘Z’, нет, переход к блоку 17  
17. ‘о’ >= ‘A’ && ‘о’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 13  
15. ‘г’ >= ‘A’ && ‘г’ <= ‘Z’, нет, переход к блоку 17  
17. ‘г’ >= ‘A’ && ‘г’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 14  
15. ‘у’ >= ‘A’ && ‘у’ <= ‘Z’, нет, переход к блоку 17  
17. ‘у’ >= ‘A’ && ‘у’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 15  
15. ‘л’ >= ‘A’ && ‘л’ <= ‘Z’, нет, переход к блоку 17  
17. ‘л’ >= ‘A’ && ‘л’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 16  
15. ‘я’ >= ‘A’ && ‘я’ <= ‘Z’, нет, переход к блоку 17  
17. ‘я’ >= ‘A’ && ‘я’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 17  
15. ‘т’ >= ‘A’ && ‘т’ <= ‘Z’, нет, переход к блоку 17  
17. ‘т’ >= ‘A’ && ‘т’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 18  
15. ‘ь’ >= ‘A’ && ‘ь’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ь’ >= ‘A’ && ‘ь’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 19  
15. ‘.’ >= ‘A’ && ‘.’ <= ‘Z’, нет, переход к блоку 17  
17. ‘.’ >= ‘A’ && ‘.’ <= ‘Я’, нет, переход к блоку 14  
15. Вход в цикл: Переменная k = 20. Граница достигнута, переход к блоку 19  
19. Возврат something из подпрограммы  
Переприсваивание переменных text = “маша вышла погулять.”  
Переходим к выполнению подпрограммы format(word):  
14. Вход в цикл: Переменная k = 0  
15. ‘в’ >= ‘A’ && ‘в’ <= ‘Z’, нет, переход к блоку 17  
17. ‘в’ >= ‘A’ && ‘в’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 1  
15. ‘ы’ >= ‘A’ && ‘ы’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ы’ >= ‘A’ && ‘ы’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 2  
15. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ш’ >= ‘A’ && ‘ш’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 3  
15. ‘л’ >= ‘A’ && ‘л’ <= ‘Z’, нет, переход к блоку 17  
17. ‘л’ >= ‘A’ && ‘л’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 4  
15. ‘а’ >= ‘A’ && ‘а’ <= ‘Z’, нет, переход к блоку 17  
17. ‘а’ >= ‘A’ && ‘а’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 5. Граница достигнута, переход к блоку 19  
19. Возврат something из подпрограммы  
Переприсваивание переменных word = “вышла”  
26. Вход в цикл: Переменная i = 0  
27. Переприсваивание переменных fact = 1, sequence = 0  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘м’, да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 1  
27. Переприсваивание переменных fact = 1, sequence = 1  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘а’, да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 2  
27. Переприсваивание переменных fact = 1, sequence = 2  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘ш’, да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 3  
27. Переприсваивание переменных fact = 1, sequence = 3  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘а’, да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 4  
27. Переприсваивание переменных fact = 1, sequence = 4  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘ ’, да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 5  
27. Переприсваивание переменных fact = 1, sequence = 5  
28. Вход в цикл: Переменная k = 0  
29. ‘в’ != ‘в’, нет, переход к блоку 31  
31. Переприсваивание переменных sequence++ = 5 + 1 = 6  
32. Вход в цикл: Переменная k = 1  
29. ‘ы’ != ‘ы’, нет, переход к блоку 31  
31. Переприсваивание переменных sequence++ = 6 + 1 = 7  
32. Вход в цикл: Переменная k = 2  
29. ‘ш’ != ‘ш’, нет, переход к блоку 31  
31. Переприсваивание переменных sequence++ = 7 + 1 = 8  
32. Вход в цикл: Переменная k = 3  
29. ‘л’ != ‘л’, нет, переход к блоку 31  
31. Переприсваивание переменных sequence++ = 8 + 1 = 9  
32. Вход в цикл: Переменная k = 4  
29. ‘а’ != ‘а’, нет, переход к блоку 31  
31. Переприсваивание переменных sequence++ = 9 + 1 = 10  
32. Вход в цикл: Переменная k = 5. Граница достигнута, переход к блоку 32  
32. 1 == 1, да, переход к блоку 33  
33. ‘ ‘ == ‘’ || ‘ ‘ == ‘\0’ || ‘ ‘ == ‘.’ || ‘ ‘ == ‘,’ || ‘ ‘ == ‘!’ || ‘ ‘ == ‘?’, да, переход к блоку 34  
34. 5 == 0 || ‘ ‘ == ‘ ‘, да, переход к блоку 36  
36. Возврат i из подпрограммы  
Переприсваивание переменных position = 5  
7. 5 != -1, да, переход к блоку 8  
8. Вход в цикл: Переменная i = 5  
9. ‘в‘ == ‘ ‘, нет, переход к блоку 8  
8. Вход в цикл: Переменная i = 4  
9. ‘ ‘ == ‘ ‘, да, переход к блоку 10  
10. Переприсваивание переменных counter++ = 1 + 1 = 2  
8. Вход в цикл: Переменная i = 3  
9. ‘а‘ == ‘ ‘, нет, переход к блоку 8  
8. Вход в цикл: Переменная i = 2  
9. ‘ш‘ == ‘ ‘, нет, переход к блоку 8  
8. Вход в цикл: Переменная i = 1  
9. ‘а‘ == ‘ ‘, нет, переход к блоку 8  
8. Вход в цикл: Переменная i = 0. Граница достигнута, переход к блоку 12.  
12. Вывод «Номер слова в заданном тексте: 2»  
13. Очистка динамической памяти у массивов main\_text, main\_word  
37. Конец

Вариант II

1. Начало  
38. Переприсваивание переменных counter = 0  
2. Вывод «Введите текст: »  
3. Переход к выполнению подпрограммы read\_string():  
20. Работа с переменными:  
len = 1, выделяем память под s (1 байт), считываем введённым пользователем символ с консоли в переменную с = 'B'  
21. 'B' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (1 байт), s[0] = 'B', считываем в переменную с следующий символ с консоли 'i', len++ = 1+1=2  
21. 'i' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (2 байта), s[1] = 'i', считываем в переменную с следующий символ с консоли 'g', len++ = 2+1=3  
21. 'g' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (3 байта), s[2] = 'g', считываем в переменную с следующий символ с консоли ' ', len++ = 3+1=4  
21. ' ' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (4 байта), s[3] = ' ', считываем в переменную с следующий символ с консоли 'a', len++ = 4+1=5  
21. 'a' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (5 байтов), s[4] = 'a', считываем в переменную с следующий символ с консоли 'l', len++ = 5+1=6  
21. 'l' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (6 байтов), s[5] = 'l', считываем в переменную с следующий символ с консоли 'i', len++ = 6+1=7  
21. 'i' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (7 байтов), s[6] = 'i', считываем в переменную с следующий символ с консоли 'b', len++ = 7+1=8  
21. 'b' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (8 байтов), s[7] = 'b', считываем в переменную с следующий символ с консоли 'i', len++ = 8+1=9  
21. 'i' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (9 байтов), s[8] = 'i', считываем в переменную с следующий символ с консоли '.', len++ = 9+1=10  
21. '.' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (10 байтов), s[9] = '.', считываем в переменную с следующий символ с консоли '\n',   
len++ = 10 + 1 = 11  
21. '\n' != '\n', нет, переход к блоку 23  
23. Переприсваивание переменных s[10] = ‘\0’  
24. Возврат s из подпрограммы  
Переприсваивание переменных main\_text = “Big alibi.”  
4. Вывод «Введите ваше слово: »  
5. Переход к выполнению подпрограммы read\_string():  
20. Работа с переменными:  
len = 1, выделяем память под s (1 байт), считываем введённым пользователем символ с консоли в переменную с = 'b'  
21. 'b' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (1 байт), s[0] = 'b', считываем в переменную с следующий символ с консоли 'i', len++ = 1+1=2  
21. 'b' != '\n', да, переход к блоку 22  
22. Перевыделение памяти под символьный массив s (2 байта), s[1] = 'i', считываем в переменную с следующий символ с консоли '\n', len++ = 2+1=3  
21. '\n' != '\n', нет, переход к блоку 23  
23. Переприсваивание переменных s[2] = '\0'  
24. Возврат s из подпрограммы  
Переприсваивание переменных main\_word = “bi”  
6. Переход к выполнению подпрограммы find(main\_text, main\_word)  
25. Переход к выполнению подпрограммы format(text)  
14. Вход в цикл: Переменная k = 0  
15. ‘B’ >= ‘A’ && ‘B’ <= ‘Z’, да, переход к блоку 16  
16. Переприсваивание переменных something[0] = 'B' – 'A' + 'a' = 'b'  
17. ‘b’ >= ‘A’ && ‘b’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 1  
15. ‘i’ >= ‘A’ && ‘i’ <= ‘Z’, нет, переход к блоку 17  
17. ‘i’ >= ‘A’ && ‘i’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 2  
15. ‘g’ >= ‘A’ && ‘g’ <= ‘Z’, нет, переход к блоку 17  
17. ‘g’ >= ‘A’ && ‘g’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 3  
15. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Z’, нет, переход к блоку 17  
17. ‘ ’ >= ‘A’ && ‘ ’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 4  
15. ‘a’ >= ‘A’ && ‘a’ <= ‘Z’, нет, переход к блоку 17  
17. ‘a’ >= ‘A’ && ‘a’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 5  
15. ‘l’ >= ‘A’ && ‘l’ <= ‘Z’, нет, переход к блоку 17  
17. ‘l’ >= ‘A’ && ‘l’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 6  
15. ‘i’ >= ‘A’ && ‘i’ <= ‘Z’, нет, переход к блоку 17  
17. ‘i’ >= ‘A’ && ‘i’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 7  
15. ‘b’ >= ‘A’ && ‘b’ <= ‘Z’, нет, переход к блоку 17  
17. ‘b’ >= ‘A’ && ‘b’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 8  
15. ‘i’ >= ‘A’ && ‘i’ <= ‘Z’, нет, переход к блоку 17  
17. ‘i’ >= ‘A’ && ‘i’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 9  
15. ‘.’ >= ‘A’ && ‘.’ <= ‘Z’, нет, переход к блоку 17  
17. ‘.’ >= ‘A’ && ‘.’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 10. Граница достигнута, переход к блоку 19  
19. Возврат something из подпрограммы  
Переприсваивание переменных text = “big alibi”  
Переходим к выполнению подпрограммы format(word):  
14. Вход в цикл: Переменная k = 0  
15. ‘b’ >= ‘A’ && ‘b’ <= ‘Z’, нет, переход к блоку 17  
17. ‘b’ >= ‘A’ && ‘b’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 1  
15. ‘i’ >= ‘A’ && ‘i’ <= ‘Z’, нет, переход к блоку 17  
17. ‘i’ >= ‘A’ && ‘i’ <= ‘Я’, нет, переход к блоку 14  
14. Вход в цикл: Переменная k = 2. Граница достигнута, переход к блоку 19  
19. Возврат something из подпрограммы  
Переприсваивание переменных word = “bi”  
26. Вход в цикл: Переменная i = 0  
27. Переприсваивание переменных fact = 1, sequence = 0  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'b', нет, переход в блок 31  
31. Переприсваивание переменных sequence++ = 0 + 1 = 1  
28. Вход в цикл: Переменная k = 1  
29. 'i' != 'i', нет, переход в блок 31  
31. Переприсваивание переменных sequence++ = 1 + 1 = 2  
28. Вход в цикл: Переменная k = 2. Граница достигнута, переход к блоку 32  
32. 1 == 1, да, переход к блоку 33  
33. 'g' == ' ' || 'g' == ' \0' || 'g' == '.', нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 1  
27. Переприсваивание переменных fact = 1, sequence = 1  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'i', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 2  
27. Переприсваивание переменных fact = 1, sequence = 2  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'g', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 3  
27. Переприсваивание переменных fact = 1, sequence = 3  
28. Вход в цикл: Переменная k = 0  
29. 'b' != ' ', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 4  
27. Переприсваивание переменных fact = 1, sequence = 4  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'a', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 5  
27. Переприсваивание переменных fact = 1, sequence = 5  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'l', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 6  
27. Переприсваивание переменных fact = 1, sequence = 6  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'i', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 7  
27. Переприсваивание переменных fact = 1, sequence = 7  
28. Вход в цикл: Переменная k = 0  
29. 'b' != 'b', нет, переход в блок 31  
31. Переприсваивание переменных sequence++ = 7 + 1 = 8  
28. Вход в цикл: Переменная k = 1  
29. 'i' != 'i', нет, переход в блок 31  
31. Переприсваивание переменных sequence++ = 8 + 1 = 9  
28. Вход в цикл: Переменная k = 2. Граница достигнута, переход к блоку 32  
32. fact == 1, да, переход к блоку 33  
33. ‘.’ == ‘ ‘ || ‘.’ == ‘\0‘ || ‘.’ == ‘.‘ || ‘.’ == ‘,’ || ‘.’ == ‘!’ || ‘.’ == ‘?’, да, переход к блоку 34  
34. 7 == 0 || ‘i’ == ‘ ‘, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 8  
27. Переприсваивание переменных fact = 1, sequence = 8  
29. 'b' != 'i', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 9  
27. Переприсваивание переменных fact = 1, sequence = 9  
29. 'b' != '.', да, переход к блоку 30  
30. Переприсваивание переменных fact = 0  
32. 0 == 1, нет, переход к блоку 26  
26. Вход в цикл: Переменная i = 10. Граница достигнута, переход к блоку 35  
35. Возврат -1 из подпрограммы  
Переприсваивание переменных position = -1  
7. -1 != -1, нет, переход к блоку 11  
11. Вывод «Заданное слово в тексте отсутствует»  
13. Очистка динамической памяти у массивов main\_text, main\_word  
37. Конец

1. Код программы:

#include <iostream>

using namespace std;

//функция для чтения строки, задаваемой пользователем

char\* read\_string() {

int len = 1;

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

char c = getchar(); // считываем первый символ из консоли

while (c != '\n') {

s = (char\*)realloc(s, len \* sizeof(char)); // перераспределяем память

s[len - 1] = c;

c = getchar(); // и снова считываем символ из консоли

len++;

}

s[len - 1] = '\0';

return s;

}

//функция для преобразования всех букв строки в строчные

char\* format(char\* something) {

for (int i = 0; i < strlen(something); i++) {

if (something[i] >= 'A' && something[i] <= 'Z')

something[i] = something[i] + 'a' - 'A';

if (something[i] >= 'А' && something[i] <= 'Я')

something[i] = something[i] + 'а' - 'А';

}

return something;

}

//функция, отвечающая за поиск заданного слова в заданном тексте

int find(char\* text, char\* word) {

text = format(text); word = format(word); // для улучшенного поиска

for (int i = 0; i < strlen(text); i++) { // сделаем все символы строки

int fact = 1, sequence = i; // строго строчными

for (int k = 0; k < strlen(word); k++) {

if (word[k] != text[sequence]) {

fact = 0;

break;

}

sequence++;

}

if (fact == 1) { // цепочка проверок на независимость слова

if (text[sequence] == ' ' || text[sequence] == '\0' || text[sequence] == '.' || text[sequence] == ',' || text[sequence] == '!' || text[sequence] == '?') {

if (i == 0 || text[i - 1] == ' ')

return i;

}

}

}

return -1;

}

// точка входа в программу

int main() {

setlocale(LC\_ALL, "rus");

system("chcp 1251 >> null");

cout << "Введите текст: ";

char\* main\_text = read\_string();

cout << "Введите ваше слово: ";

char\* main\_word = read\_string();

int position = find(main\_text, main\_word), counter = 1;

if (position != -1) {

for (int i = position; i > 0; i--) {

if (main\_text[i] == ' ')

counter++; // подсчёт кол-ва пробелов перед словом

}

cout << "Номер слова в заданном тексте: " << counter << endl;

}

else

cout << "Заданное слово в тексте отсутствует." << endl;

main\_text = (char\*)realloc(NULL, 0); // освобождаем память

main\_word = (char\*)realloc(NULL, 0); // можно было бы через free,

system("pause"); // но так нагляднее!

return 0;

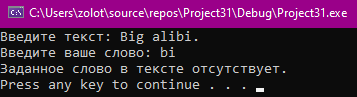
}

1. Результат выполнения работы программы:

Text

Description automatically generated

Отладочный пример №1



Отладочный пример №2

1. Вывод:

В ходе выполнения работы были изучены способы задать программе массив символов произвольной длины, используя при этом функции распределения памяти malloc, realloc (calloc не употреблён по причине ненадобности). Также были изучены способы инициализации функции, которая возвращает не просто значение, но указатель на динамический массив со значениями.

Было проделано изучение материала по части использования так называемых 'высших функций', способов ветвления алгоритмов и обработки символов, также в ходе многострочной отладки были выявлены исключения, которые были успешно устранены на программном уровне.

На контрольных примерах мы убедились, что программа работает превосходно при указанных введённых данных.

Был оформлен комплект документации на программный код.