**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 №10**

*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)

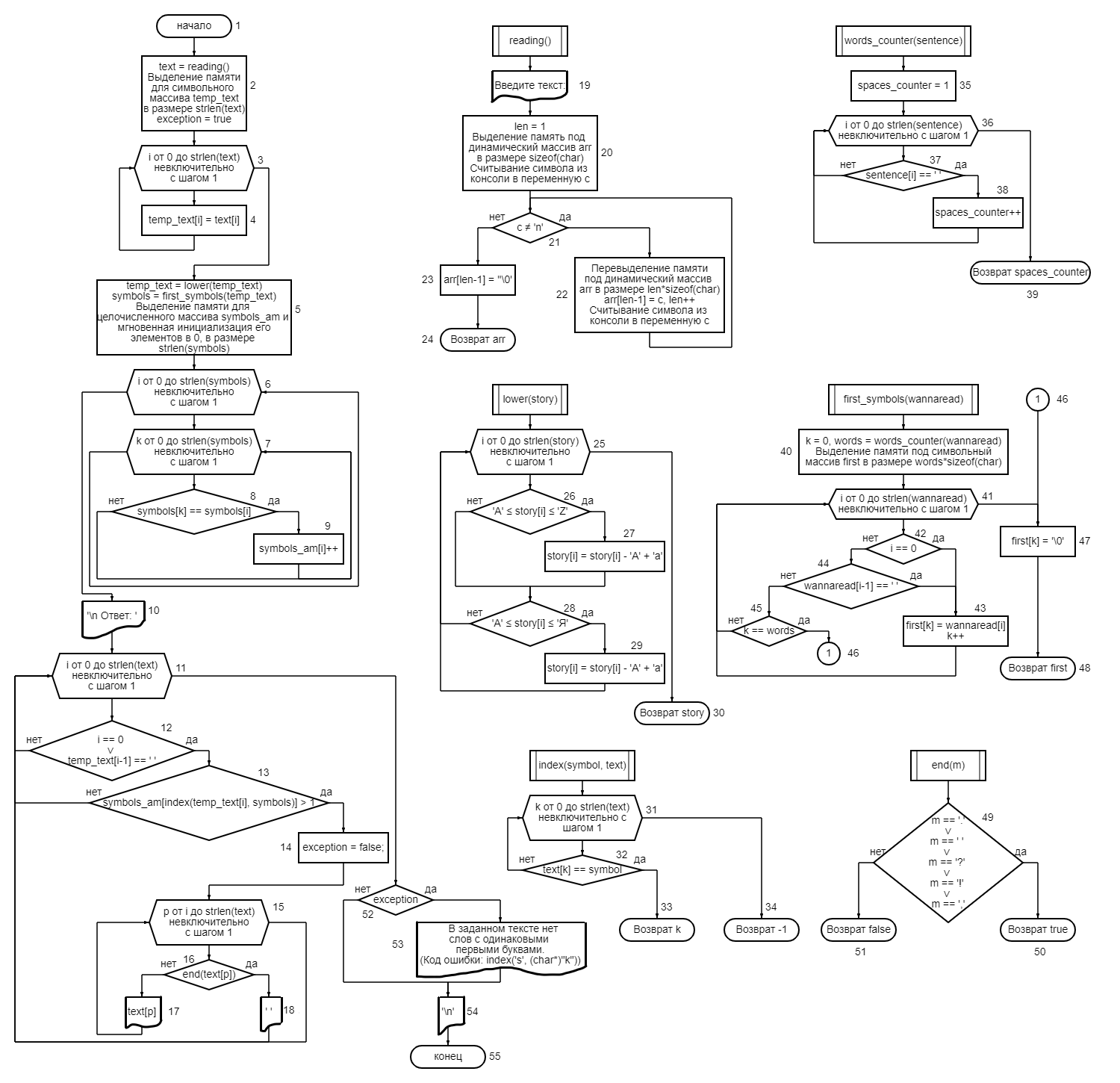
Москва 2021 г.

**Задание №10**

Задание: задан текст. Напечатать все слова, в которых повторяется первая буква слова.

1. Таблица имён:

|  |  |  |
| --- | --- | --- |
| Исходные данные | | |
| text | Указатель на символьный массив | Исходный текст, введённый пользователем |
| Рабочие переменные | | |
| temp\_text | Указатель на символьный массив | Копия исходного текста, чтобы при работе не испортить оригинал |
| text[i], text[p] | Символьный | Символ текста, введённого пользователем |
| temp\_text[i] temp\_text[i – 1] | Целочисленный | Символ копии текста, введённого пользователем |
| symbols | Указатель на символьный массив | Массив из всех первых букв слова в тексте, введённого пользователем |
| symbols[i] | Cимвольный | Первая буква i по счёту слова в text |
| symbols[k] | Cимвольный | Первая буква k по счёту слова в text |
| symbols\_am | Указатель на целочисленный массив | Массив, который показывает количество встреченных раз символа symbols[i]\symbols[k] в массиве symbols |
| arr | Указатель на символьный массив | Динамический массив для считывания текста, что задаёт пользователь |
| arr[len - 1] | Символьный | Новое место для записи символа с текста, заданного пользователем |
| len | Целочисленный | Постоянно меняющаяся длина динамического массива |
| с | Символьный | Символ, в который считывается весь текст, введённый пользователем |
| story | Указатель на символьный массив | Массив, в котором заглавные буквы преобразуются в строчные |
| spaces\_counter | Целочисленный | Счётчик пробелов в предложении |
| sentence | Указатель на символьный массив | Массив для подсчёта слов в тексте |
| wannaread | Указатель на символьный массив | Массив для анализа первых букв в слове в изначальном тексте пользователя |
| first | Указатель на символьный массив | Массив первых букв слов текста пользователя |
| first[k] | Символьный | Место, куда записывается первая буква слова из исходного текста |
| k | Целочисленный | Переменная-счётчик для заполнения массива first |
| k, i, p | Целочисленные | Переменные-счётчики в циклах |
| symbol | Символьный | Символ для поиска в тексте |
| text | Указатель на символьный массив | Текст, в котором ищется символ |
| text[k] | Символ | Символ, с которым сравнивается symbol для установления его позиции в тексте |
| m | Символьный | Символ текста для проверки слова на окончание одним из символов |
| Результат | | |
| Все такие text[p], при которых p лежит в отрезке от [0; s] или [i – 1; s] при text[i-1] – пробел, где s - позиция ближайшего символа конца слова в отрезке от [0; strlen(text) – 1] или [i-1; strlen(text) – 1] | | |

1. Блок-схема:
2. Отладочный пример:

Вариант I

1. Начало  
2. Переходим к выполнению подпрограммы reading():  
19. Вывод “Введите текст: ”  
20. Переприсваивание переменных  
len = 1, выделение памяти под динамический массив arr в размере (1), Считывание символа ‘М’ в переменную с  
21. ‘М’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (1)  
Переприсваивание переменных arr[0] = ‘M’, len++ = 1 + 1 = 2  
Считывание в переменную с символа ’а’  
21. ‘а’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (2)  
Переприсваивание переменных arr[1] = ‘а’, len++ = 2 + 1 = 3  
Считывание в переменную с символа ’м’  
21. ‘м’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (3)  
Переприсваивание переменных arr[2] = ‘м’, len++ = 3 + 1 = 4  
Считывание в переменную с символа ’а’  
21. ‘а’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (4)  
Переприсваивание переменных arr[3] = ‘а’, len++ = 4 + 1 = 5  
Считывание в переменную с символа ’,’  
21. ‘,’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (5)  
Переприсваивание переменных arr[4] = ‘,’, len++ = 5 + 1 = 6  
Считывание в переменную с символа ’ ’  
21. ‘ ’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (6)  
Переприсваивание переменных arr[5] = ‘ ’, len++ = 6 + 1 = 7  
Считывание в переменную с символа ’г’  
21. ‘г’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (7)  
Переприсваивание переменных arr[6] = ‘г’, len++ = 7 + 1 = 8  
Считывание в переменную с символа ’д’  
21. ‘д’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (8)  
Переприсваивание переменных arr[7] = ‘д’, len++ = 8 + 1 = 9  
Считывание в переменную с символа ’е’  
21. ‘е’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (9)  
Переприсваивание переменных arr[8] = ‘е’, len++ = 9 + 1 = 10  
Считывание в переменную с символа ’ ’  
21. ‘ ’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (10)  
Переприсваивание переменных arr[9] = ‘ ’, len++ = 10 + 1 = 11  
Считывание в переменную с символа ’о’  
21. ‘о’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (11)  
Переприсваивание переменных arr[10] = ‘о’, len++ = 11 + 1 = 12  
Считывание в переменную с символа ’ч’  
21. ‘ч’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (12)  
Переприсваивание переменных arr[11] = ‘ч’, len++ = 12 + 1 = 13  
Считывание в переменную с символа ’к’  
21. ‘к’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (13)  
Переприсваивание переменных arr[12] = ‘к’, len++ = 13 + 1 = 14  
Считывание в переменную с символа ’и’  
21. ‘и’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (14)  
Переприсваивание переменных arr[13] = ‘и’, len++ = 14 + 1 = 15  
Считывание в переменную с символа ’?’  
21. ‘?’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти под динамический массив arr в размере (15)  
Переприсваивание переменных arr[14] = ‘?’, len++ = 15 + 1 = 16  
Считывание в переменную с символа ’\n’  
21. ‘\n’ != ‘\n’, нет, переход к блоку 23  
23. Переприсваивание переменных arr[15] =’\0’  
24. Возврат ”Мама, где мои очки?\0” из подпрограммы  
Переприсваивание переменных text = ”Мама, где мои очки?”  
Выделение памяти под символьный массив temp\_text в размере (16)  
exception = true  
3. Вход в цикл: Переменная i = 0  
4. Переприсваивание переменных: temp\_text[0] = ‘М’  
3. Вход в цикл: Переменная i = 1  
4. Переприсваивание переменных: temp\_text[1] = ‘а’  
3. Вход в цикл: Переменная i = 2  
4. Переприсваивание переменных: temp\_text[2] = ‘м’  
3. Вход в цикл: Переменная i = 3  
4. Переприсваивание переменных: temp\_text[3] = ‘а’  
3. Вход в цикл: Переменная i = 4  
4. Переприсваивание переменных: temp\_text[4] = ‘,’  
3. Вход в цикл: Переменная i = 5  
4. Переприсваивание переменных: temp\_text[5] = ‘ ’  
3. Вход в цикл: Переменная i = 6  
4. Переприсваивание переменных: temp\_text[6] = ‘г’  
3. Вход в цикл: Переменная i = 7  
4. Переприсваивание переменных: temp\_text[7] = ‘д’  
3. Вход в цикл: Переменная i = 8  
4. Переприсваивание переменных: temp\_text[8] = ‘е’  
3. Вход в цикл: Переменная i = 9  
4. Переприсваивание переменных: temp\_text[9] = ‘ ’  
3. Вход в цикл: Переменная i = 10  
4. Переприсваивание переменных: temp\_text[10] = ‘о’  
3. Вход в цикл: Переменная i = 11  
4. Переприсваивание переменных: temp\_text[11] = ‘ч’  
3. Вход в цикл: Переменная i = 12  
4. Переприсваивание переменных: temp\_text[12] = ‘к’  
3. Вход в цикл: Переменная i = 13  
4. Переприсваивание переменных: temp\_text[13] = ‘и’  
3. Вход в цикл: Переменная i = 14  
4. Переприсваивание переменных: temp\_text[14] = ‘?’  
3. Вход в цикл: Переменная i = 15  
4. Переприсваивание переменных: temp\_text[14] = ‘\0’  
5. Переход к выполнению подпрограммы lower(temp\_text):  
25. Вход в цикл: Переменная i = 0  
26. ‘A’ <= ‘М‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘М‘ <= ‘Я’, да, переход к блоку 29  
29. Переприсваивание переменных story[0] = ‘М‘ – ‘A’ + ‘a’ = ‘м’  
25. Вход в цикл: Переменная i = 1  
26. ‘A’ <= ‘а‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘а‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 2  
26. ‘A’ <= ‘м‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘а‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 3  
26. ‘A’ <= ‘а‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘а‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 4  
26. ‘A’ <= ‘,‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘,‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 5  
26. ‘A’ <= ‘ ‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘ ‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 6  
26. ‘A’ <= ‘г‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘г‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 7  
26. ‘A’ <= ‘д‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘д‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 8  
26. ‘A’ <= ‘е‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘е‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 9  
26. ‘A’ <= ‘ ‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘ ‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 10  
26. ‘A’ <= ‘о‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘о‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 11  
26. ‘A’ <= ‘ч‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘ч‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 12  
26. ‘A’ <= ‘к‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘к‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 13  
26. ‘A’ <= ‘и‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘и‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 14  
26. ‘A’ <= ‘?‘ <= ‘Z’, нет, переход к блоку 28  
26. ‘A’ <= ‘?‘ <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 15. Граница достигнута, переход в блок 30  
30. Возврат “мама, где очки?”  
Переприсваивание переменных   
temp\_text = lower(temp\_text) = “мама, где очки?\0”  
Переходим к выполнению подпрограммы first\_symbols(temp\_text):  
40. Переходим к выполнению подпрограммы words\_counter(wannaread):  
35. Переприсваивание переменных spaces\_counter = 1  
36. Вход в цикл: Переменная i = 0  
37. ‘м’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 1  
37. ‘а’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 2  
37. ‘м’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 3  
37. ‘а’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 4  
37. ‘,’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 5  
37. ‘ ’ == ‘ ‘, да, переход к блоку 38  
38. Переприсваивание переменных spaces\_counter++ = 1 + 1 = 2  
36. Вход в цикл: Переменная i = 6  
37. ‘г’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 7  
37. ‘д’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 8  
37. ‘е’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 9  
37. ‘ ’ == ‘ ‘, да, переход к блоку 38  
38. Переприсваивание переменных spaces\_counter++ = 2 + 1 = 3  
36. Вход в цикл: Переменная i = 10  
37. ‘о’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 11  
37. ‘ч’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 12  
37. ‘к’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 13  
37. ‘и’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 14  
37. ‘?’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 15  
37. ‘\0’ == ‘ ‘, нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 16. Граница достигнута, переход к блоку 39  
39. Возврат 3 из подпрограммы  
Переприсваивание переменных   
k = 0, words = words\_counter(wannaread) = 3  
Выделение памяти под символьный массив first в размере 3  
41. Вход в цикл: Переменная i = 0  
42. 0 == 0, да, переход в блок 43  
43. Переприсваивание переменных first[0] = 'м’, k++ = 0 + 1 = 1  
41. Вход в цикл: Переменная i = 1  
42. 1 == 0, нет, переход в блок 44  
44. 'м' == ' ', нет, переход в блок 45  
45. 1 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 2  
42. 2 == 0, нет, переход в блок 44  
44. 'а' == ' ', нет, переход в блок 45  
45. 1 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 3  
42. 3 == 0, нет, переход в блок 44  
44. 'м' == ' ', нет, переход в блок 45  
45. 1 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 4  
42. 4 == 0, нет, переход в блок 44  
44. 'а' == ' ', нет, переход в блок 45  
45. 1 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 5  
42. 5 == 0, нет, переход в блок 44  
44. ',' == ' ', нет, переход в блок 45  
45. 1 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 6  
42. 6 == 0, нет, переход в блок 44  
44. ' ' == ' ', да, переход в блок 43  
43. Переприсваивание переменных first[1] = 'г’, k++ = 1 + 1 = 2  
41. Вход в цикл: Переменная i = 7  
42. 7 == 0, нет, переход в блок 44  
44. 'г' == ' ', нет, переход в блок 45  
45. 2 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 8  
42. 8 == 0, нет, переход в блок 44  
44. 'д' == ' ', нет, переход в блок 45  
45. 2 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 8  
42. 8 == 0, нет, переход в блок 44  
44. 'е' == ' ', нет, переход в блок 45  
45. 2 == 3, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 9  
42. 9 == 0, нет, переход в блок 44  
44. ' ' == ' ', да, переход в блок 43  
43. Переприсваивание переменных first[2] = 'о’, k++ = 2 + 1 = 3  
41. Вход в цикл: Переменная i = 10  
42. 10 == 0, нет, переход в блок 44  
44. 'о' == ' ', нет, переход в блок 45  
45. 3 == 3, да, переход в блок 46  
46. Ссылка на блок 47  
47. Возврат “мго” из подпрограммы  
Переприсваивание переменных symbols = first\_symbols(temp\_text) = “мго”   
Выделение памяти под целочисленный массив symbols\_am в размере (16)  
6. Вход в цикл: Переменная i = 0  
7. Вход в цикл: Переменная k = 0  
8. 'м' == 'м', да, переход к блоку 9  
9. symbols\_am[0] = 0 + 1 = 1  
7. Вход в цикл: Переменная k = 1  
8. 'г' == 'м', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 2  
8. 'о' == 'м', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 3. Граница достигнута, переход к блоку 6  
6. Вход в цикл: Переменная i = 1  
7. Вход в цикл: Переменная k = 0  
8. 'м' == 'г', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 1  
8. 'г' == 'г', да, переход к блоку 9  
9. symbols\_am[1] = 0 + 1 = 1  
7. Вход в цикл: Переменная k = 2  
8. 'о' == 'г', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 3. Граница достигнута, переход к блоку 6  
6. Вход в цикл: Переменная i = 2  
7. Вход в цикл: Переменная k = 0  
8. 'м' == 'о', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 1  
8. 'г' == 'о', нет, переход к блоку 7  
7. Вход в цикл: Переменная k = 2  
8. 'о' == 'о', да, переход к блоку 9  
 9. symbols\_am[2] = 0 + 1 = 1  
6. Вход в цикл: Переменная i = 3. Граница достигнута, переход к блоку 10  
10. Вывод (перенос на следующую строку)“Ответ: ”  
11. Вход в цикл: Переменная i = 0  
12. 0 == 0 || ‘{Some garbage}’ == ‘ ’, да, переход к блоку 13  
13. Переходим к выполнению подпрограммы index(temp\_text[0], symbols)  
31. Вход в цикл: Переменная k = 0  
32. 'м' == 'м', да, переход к блоку 33  
33. Возврат 0 из подпрограммы  
1 > 1, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 1  
12. 1 == 0 || ‘м’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 2  
12. 2 == 0 || ‘а’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 3  
12. 3 == 0 || ‘м’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 4  
12. 4 == 0 || ‘а’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 5  
12. 5 == 0 || ‘,’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 6  
12. 6 == 0 || ‘ ’ == ‘ ’, да, переход к блоку 13  
13. Переходим к выполнению подпрограммы index(temp\_text[6], symbols)  
31. Вход в цикл: Переменная k = 0  
32. 'м' == 'г', нет, переход к блоку 31  
31. Вход в цикл: Переменная k = 1  
32. 'г' == 'г', да, переход к блоку 33  
33. Возврат 1 из подпрограммы  
1 > 1, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 7  
12. 9 == 0 || ‘г’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 8  
12. 8 == 0 || ‘д’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 9  
12. 9 == 0 || ‘е’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 10  
12. 10 == 0 || ‘ ’ == ‘ ’, да, переход к блоку 13  
13. Переходим к выполнению подпрограммы index(temp\_text[10], symbols)  
31. Вход в цикл: Переменная k = 0  
32. 'м' == 'о', нет, переход к блоку 31  
31. Вход в цикл: Переменная k = 1  
32. 'г' == 'о', нет, переход к блоку 31  
31. Вход в цикл: Переменная k = 2  
32. 'о' == 'о', да, переход к блоку 33  
33. Возврат 2 из подпрограммы  
1 > 1, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 11  
12. 11 == 0 || ‘о’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 12  
12. 12 == 0 || ‘ч’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 13  
12. 13 == 0 || ‘к’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 14  
12. 14 == 0 || ‘и’ == ‘ ’, нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 15. Граница достигнута, переход к блоку 51  
52. True, да, переходим к блоку 52  
53. Переходим к выполнению подпрограммы index(‘s’, (char\*)”k”):  
31. Вход в цикл: Переменная k = 0  
32. ‘s’ == ‘k’, нет, переход к блоку 31  
31. Вход в цикл: Переменная k = 1. Граница достигнута, переходим в блок 34  
34. Возврат -1 из подпрограммы  
Вывод “В данном тексте нет слов с одинаковыми буквами. (Код ошибки: -1)”  
54. Перевод текста на новую строку  
55. Конец

Вариант II

1. Начало  
2. Переходим к выполнению подпрограммы reading():  
19. Вывод “Введите текст: ”  
20. Переприсваивание переменных len = 1  
Выделение памяти под динамический массив arr в размере (1)  
Считывание символа ‘I’ из консоли в переменную с  
21. ‘I’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти в динамический массив arr в размере (1)  
Переприсваивание переменных arr[0] = ‘I’, len++ = 1 + 1 = 2  
Считывание символа ‘ ’ из консоли в переменную с  
21. ‘ ’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти в динамический массив arr в размере (2)  
Переприсваивание переменных arr[1] = ‘I’, len++ = 2 + 1 = 3  
Считывание символа ‘i’ из консоли в переменную с  
21. ‘i’ != ‘\n’, да, переход к блоку 22  
22. Перевыделение памяти в динамический массив arr в размере (3)  
Переприсваивание переменных arr[2] = ‘i’, len++ = 3 + 1 = 4  
Считывание символа ‘\n’ из консоли в переменную с  
21. ‘\n’ != ‘\n’, нет, переход к блоку 23  
23. Переприсваивание переменных arr[3] = ‘\0’  
24. Возврат “I i” из подпрограммы  
Переприсваивание переменных: text = “I i”, exception = true  
Выделение памяти для символьного массива temp\_text в размере (4)  
3. Вход в цикл: Переменная i = 0  
4. Переприсваивание переменных temp\_text[0] = ‘I’  
3. Вход в цикл: Переменная i = 1  
4. Переприсваивание переменных temp\_text[1] = ‘ ’  
3. Вход в цикл: Переменная i = 2  
4. Переприсваивание переменных temp\_text[2] = ‘i’  
3. Вход в цикл: Переменная i = 3. Граница достигнута, переход к блоку 5  
5. Переходим к выполнению подпрограммы lower(temp\_text)   
25. Вход в цикл: Переменная i = 0  
26. 'A' <= 'I' <= 'Z', да, переход к блоку 27  
27. story[0] = 'I' – 'A' + 'a’ = ‘i’  
28. ‘A’ <= ‘i' <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 1  
26. 'A' <= ' ' <= 'Z', нет, переход к блоку 28  
28. ‘A’ <= ‘ ' <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 2  
26. 'A' <= 'i' <= 'Z', нет, переход к блоку 28  
28. ‘A’ <= ‘i' <= ‘Я’, нет, переход к блоку 25  
25. Вход в цикл: Переменная i = 3. Граница достигнута, переход к блоку 30  
30. Возврат “i i” из подпрограммы  
Переходим к выполнению подпрограммы first\_symbols(temp\_text):  
40. Переходим к выполнению подпрограммы words\_couter(wannaread)  
35. Переприсваивание переменных spaces\_counter = 1  
36. Вход в цикл: Переменная i = 0  
37. 'i' == ' ', нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 1  
37. ' ' == ' ', да, переход к блоку 38  
38. Переприсваивание переменных spaces\_counter++ = 1 + 1 = 2  
36. Вход в цикл: Переменная i = 2  
37. 'i' == ' ', нет, переход к блоку 36  
36. Вход в цикл: Переменная i = 3. Граница достигнута, переход к блоку 39  
39. Возврат 2 из подпрограммы  
Переприсваивание переменных k = 0, words = 2  
Выделение памяти под символьный массив first в размере (2)  
41. Вход в цикл: Переменная i = 0  
42. 0 == 0, да, переход к блоку 43  
43. Переприсваивание переменных first[0] = 'i', k++ = 0 + 1 = 1  
41. Вход в цикл: Переменная i = 1  
42. 1 == 0, нет, переход к блоку 44  
44. 'i' == ' ', нет, переход к блоку 45  
45. 1 == 2, нет, переход к блоку 41  
41. Вход в цикл: Переменная i = 2  
42. 2 == 0, нет, переход к блоку 44  
44. ' ' == ' ', да, переход к блоку 43  
43. Переприсваивание переменных first[1] = 'i', k++ = 1 + 1 = 2  
41. Вход в цикл: Переменная i = 3. Граница достигнута , переход к блоку 47  
47. Переприсваивание переменных first[2] = '\0'  
48. Возврат “ii” из подпрограммы  
Переприсваивание переменных temp\_text = lower(temp\_text) = “i i”,   
symbols = first\_symbols(temp\_text) = “ii”  
Выделение памяти для целочисленного массива symbols\_am и мгновенная инициализация его элементов в 0, в размере (2)  
6. Вход в цикл: Переменная i = 0  
  
7. Вход в цикл: Переменная k = 0  
8. 'i' == 'i', да, переход к блоку 9  
9. Переприсваивание переменных symbols\_am[0]++ = 0 + 1 = 1  
7. Вход в цикл: Переменная k = 1  
8. 'i' == 'i', да, переход к блоку 9  
9. Переприсваивание переменных symbols\_am[0]++ = 1 + 1 = 2  
7. Вход в цикл: Переменная k = 2. Граница достигнута, переход к блоку 6  
6. Вход в цикл: Переменная i = 1  
7. Вход в цикл: Переменная k = 0  
8. 'i' == 'i', да, переход к блоку 9  
9. Переприсваивание переменных symbols\_am[1]++ = 0 + 1 = 1  
7. Вход в цикл: Переменная k = 1  
8. 'i' == 'i', да, переход к блоку 9  
9. Переприсваивание переменных symbols\_am[1]++ = 1 + 1 = 2  
7. Вход в цикл: Переменная k = 2. Граница достигнута, переход к блоку 6  
6. Вход в цикл: Переменная i = 2. Граница достигнута, переход к блоку 10  
10. Перенос текста на новую строку, вывод “Ответ” на экран  
11. Вход в цикл: Переменная i = 0  
12. 0 == 0 || <Some\_garbage> == ' ', да, переход к блоку 13  
13. Переходим к выполнению подпрограммы index(temp\_text[0], symbols)  
31. Вход в цикл: Переменная k = 0  
32. 'i' == 'i', да, переход к блоку 33  
33. Возврат 0 из подпрограммы  
2 > 1, да, переход к блоку 14  
14. Переприсваивание переменных exception = false  
15. Вход в цикл: Переменная p = 0  
16. Переходим к выполнению подпрограммы end(text[p])  
49. 'I' == '.' || 'I' == ' ' || 'I' == '?' || 'I' == '!' || 'I' == ',', нет, переход к блоку 51  
51. Возврат false из подпрограммы  
false, нет, переход к блоку 17  
17. Вывод ‘I’ на экран  
15. Вход в цикл: Переменная p = 1  
16. Переходим к выполнению подпрограммы end(text[p])  
49. ' ' == '.' || ' ' == ' ' || ' ' == '?' || ' ' == '!' || ' ' == ',', да, переход к блоку 50  
50. Возврат true из подпрограммы  
true, да, переход к блоку 18  
18. Вывод пробела на экран  
11. Вход в цикл: Переменная i = 1  
12. 1 == 0 || 'I' == ' ', нет, переход к блоку 11  
11. Вход в цикл: Переменная i = 2  
12. 2 == 0 || ' ' == ' ', да, переход к блоку 13  
13. Переход к выполнению подпрограммы index(temp\_text[i], symbols):  
31. Вход в цикл: Переменная k = 0  
32. 'i' == 'i', да, переход к блоку 33  
33. Возврат 0 из подпрограммы  
2 > 1, да, переход к блоку 14  
14. Переприсваивание переменных exception = false  
15. Вход в цикл: Переменная p = 2  
16. Переходим к выполнению подпрограммы end(text[p])  
49. 'i' == '.' || 'i' == ' ' || 'i' == '?' || 'i' == '!' || 'i' == ',', нет, переход к блоку 51  
51. Возврат false из подпрограммы  
false, нет, переход к блоку 17  
17. Вывод ‘I’ на экран  
15. Вход в цикл: Переменная p = 3. Граница достигнута, переход к блоку 11  
11. Вход в цикл: Переменная i = 3. Граница достигнута, переход к блоку 52  
52. false, нет, переход к блоку 54  
54. Перевод слова на новую строку  
55. Конец

1. Код программы:

#include <iostream>

using namespace std;

// Читаем строку, введённую пользователем

char\* reading() {

cout << "Введите текст:" << " ";

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

int len = 1; char c = getchar();

while (c != '\n') {

arr = (char\*)realloc(arr, len \* sizeof(char));

arr[len - 1] = c;

len++; c = getchar();

}

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

return arr;

}

// Для преобразования заглавных букв в строчные

char\* lower(char\* story) {

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

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

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

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

story[i] = story[i] - 'А' + 'а';

}

return story;

}

// Считаем количество слов в предложении

int words\_counter(char\* sentence) {

int spaces\_counter = 1;

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

if (sentence[i] == ' ')

spaces\_counter++;

}

return spaces\_counter;

}

// Читаем все первые символы в строке

char\* first\_symbols(char\* wannaread) {

int k = 0, words = words\_counter(wannaread);

char\* first = (char\*)malloc(words \* sizeof(char));

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

if (i == 0) {

first[k] = wannaread[i];

k++;

}

else if (wannaread[i - 1] == ' ') {

first[k] = wannaread[i];

k++;

}

else if (k == words) {

first[k] = '\0';

break;

}

}

return first;

}

// Узнаем номер позиции - первого вхождения заданного символа в строку

int index(char symbol, char\* text) {

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

if (text[k] == symbol)

return k;

}

return -1;

}

// Проверяем, правда ли слово закончилось на один из этих символов

bool end(char m) {

if (m == '.' || m == ' ' || m == '?' || m == '!' || m == ',')

return true;

return false;

}

int main() {

setlocale(LC\_ALL, "rus");

system("chcp 1251 >> null");

char\* text = reading();

bool exception = true;

// Выделяем память в другой ячейке памяти для копии текста

char\* temp\_text = (char\*)malloc(strlen(text) \* sizeof(char));

// Работаем с копией, чтобы не портить исходный

for (int i = 0; i < strlen(text); i++)

temp\_text[i] = text[i];

// Копию текста понижаем для поиска одинаковых букв в разных регистрах

temp\_text = lower(temp\_text);

// Берём все первые символы в строке

char\* symbols = first\_symbols(temp\_text);

// Делаем аналогию словаря из двух массивов - symbols, symbols\_am, где соответственно находится буква и её количество среди первых букв слов в тексте

int\* symbols\_am = (int\*)calloc(strlen(symbols), sizeof(int));

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

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

// Для каждой буковки проверяем её количество в массиве symbols

if (symbols[k] == symbols[i])

symbols\_am[i]++;

}

}

cout << endl << "Ответ: ";

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

// Проверяем, точно ли это слово с левой стороны

if (i == 0 || temp\_text[i - 1] == ' ') {

// Если первая буква слова встречается больше одного раза

if (symbols\_am[index(temp\_text[i], symbols)] > 1) {

exception = false;

for (int p = i; p < strlen(text); p++) {

// Пока не встретим символ завершения слова, печатаем побуквенно слово.

if (end(text[p])) {

cout << " ";

break;

}

cout << text[p];

}

}

}

}

if (exception)

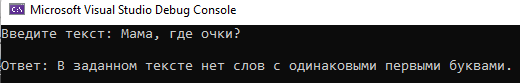
printf("В заданном тексте нет слов с одинаковыми первыми буквами. (Код ошибки: %d)", index('s', (char\*)"k"));  
 // Для красоты вывода.

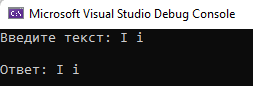
cout << endl;

return 0;

}

1. Результат выполнения работы программы:



Отладочный пример №1  


Отладочный пример №2

1. Вывод:

В ходе выполнения работы были изучены способы получения текста, задаваемого пользователем через консольную строку в символьный массив через работу с указателями. Изучили способы мгновенной инициализации элементов целочисленного массива при выделении ему памяти через функцию calloc(). Были изучены словари, и на основе их идеи были созданы два парных массива – symbols & symbols\_am, первый содержал в себе первые буквы всех слов, а второй на таких же позициях размещал количество таких букв среди всех таких же в первом массиве. Задание выполнялось в несколько этапов – понимание смысла задачи, быстрый план по пунктам на листочке бумаги, формирование блок-схемы и кода, шлифовка кода, подготовка отчёта. Функции названы мнемонично их функционалу.

Было проделано тщательное изучение материалов, посвящённых с/с++ и работе с этими языками, в том числе книга Т.А. Павловской (Программирование на языке высокого уровня)

На контрольных примерах мы убедились, что наш код работает исправно.

Был оформлен комплект документации на программный код.