|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ ИУ - Информатика и системы управления

КАФЕДРА ИУ5

**Отчёт по проектно – технологической практике**

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тип практики учебная

Название предприятия кафедра ИУ5

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

Руководитель практики **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2019 г.*

**Оглавление**

* Постановка задачи – 3 страница
* Разработка алгоритма – 4 страница
* Текст программы – 9 страница
* Тестирование – 13 страница

**Постановка задачи**

Целью домашнего задания является закрепление приобретенных в процессе выполнения лабораторных работ практических навыков реализации на языке С++ следующих приемов структурного программирования:

1. Использование методов структурного проектирования программ при разработке алгоритма выполнения задания.
2. Использование структур и массивов из структур для хранения данных.
3. Использования методов структурного программирования при разработке библиотеки базовых функций для реализации отдельных шагов алгоритма.
4. Организация обмена данными между функциями с использованием указателей и ссылок.
5. Создание и обработка динамических массивов.
6. Отладка и тестирование программ.

Необходимо ввести с клавиатуры текст который завершается точкой. Текст состоит из малых латинских букв (строчных) слова разделены не менее чем одним пробелом.

БЕЗ ИСПОЛЬЗОВАНИЯ типа STRING:

* разработать функцию для ввода такого текста с клавиатуры. Функция должна контролировать ввод, чтобы вводить только указанные в задании символы и выводить строку с введенным текстом; распечатать преобразованный текст.
* Не использовать библиотеку <iostream>
* Вариант 25: оставить в предложениях все различные слова текста, указав для каждого из них количество его вхождений в текст.

**Разработка алгоритма**

Так как пользоваться библиотекой <iostream> запрещено, нам необходимо воспользоваться иными способами вывода, такими как \_getch() и \_putch(). Тогда, для вывода полного массива, использую циклы for. Также при вводе теста необходимо контролировать ввод BACKSPACE. Учитывая вышеперечисленные условия, нужно создать отдельную функцию ввода текста, которая будет контролировать каждый введенный символ, и выполнять соответствующие действия.

Далее, когда текст введен в массив, создаем второй массив – массив слова, с которым будем сравнивать остальные слова текста (в ходе работы программы мы по очереди берем следующие слова текста).

Для того чтобы находить совпадения подстроки в строке существует функция strstr, но условие задания нам запрещает ее использовать. Поэтому, создаем ее аналог, возвращаемым значением которой будет указатель на первое вхождение подстроки в строку. Потом удостоверяемся, является ли подстрока отдельным словом, путем проверки символов до и после нее.

Как только нашлось совпадение слов, по заданию необходимо его удалить – разрабатываем функцию удаления. В нее передаем адрес начала повторяющегося слова и заменяем каждый символ следующим, идущим после слова (т.е. «накладываем» последующие символы) и уменьшаем длину массива на длину слова.

char\* word - сравниваемое слово

int count – количество повторений слова в тексте

int l\_word – длина слова

int dlina – длина массива

char\* tekst – сам текст

int j разыменования элемента массива

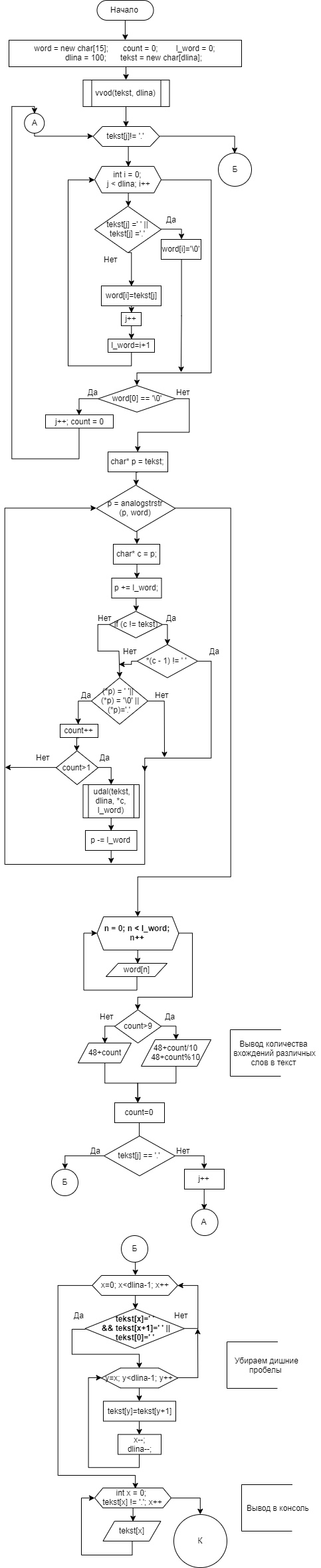
char\* p – указатель на конец слова

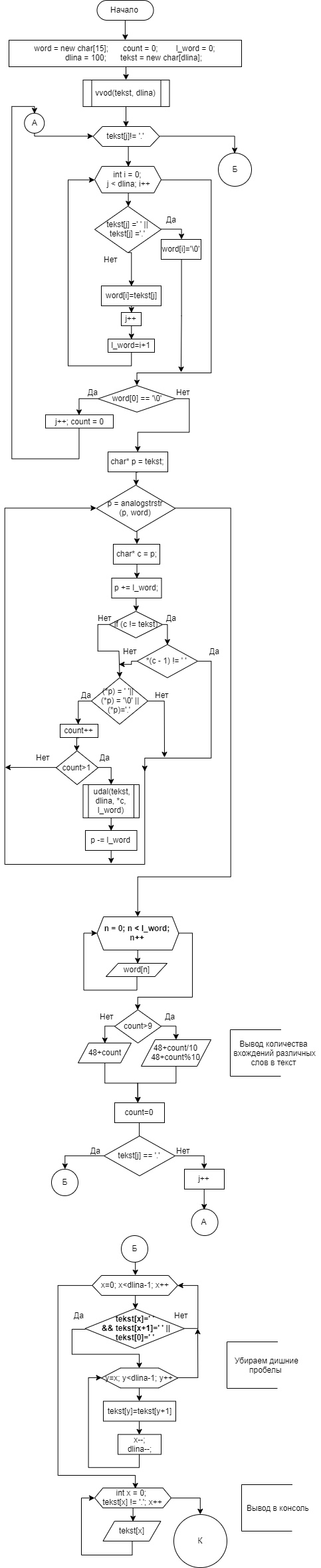
char\* c – указатель на начало слова

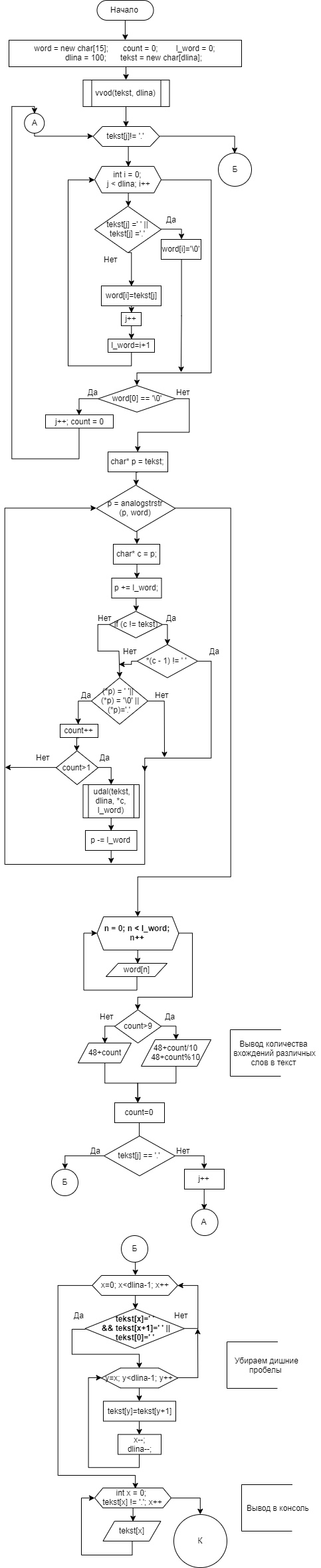
void vvod(char\* arr, int n) – функция, контролирующая ввод текста

void udal(char\* arr, int& dlina, char& nom, int l\_word) – удаление слова из текста

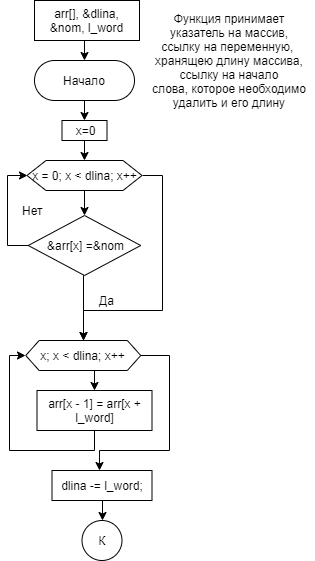
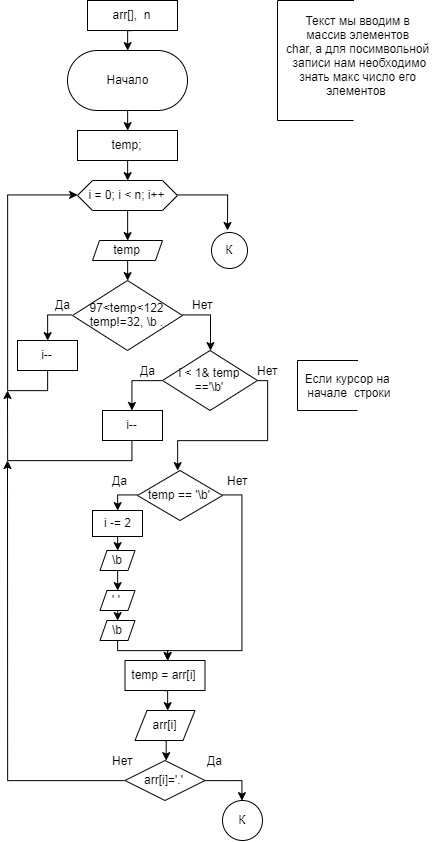
char\* analogstrstr(char\* string, char\* podstring) поиск подстроки в строке. Возвращает указатель



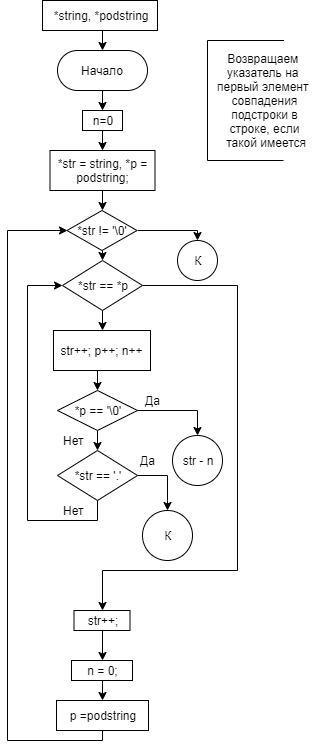




|  |  |
| --- | --- |
| **vvod** | **analogstrstr** |



**Udal**



**Текст программы**

#include <conio.h>

void vvod(char\* arr, int n);

void udal(char\* arr, int& dlina, char& nom, int l\_word);

char\* analogstrstr(char\* string, char\* podstring);

int main()

{

char\* word = new char[15];

int count = 0;

int l\_word = 0;

int dlina = 100;

char\* tekst = new char[dlina];

vvod(tekst, dlina);

int j = 0;

while (tekst[j] != '.')

{ //berem slovo

for (int i = 0; j < dlina; i++)

{

if (tekst[j] == ' ' || tekst[j] == '.') { word[i] = '\0'; break; }

word[i] = tekst[j];

j++;

l\_word = i + 1;

}

char\* p = tekst; //

while (p = analogstrstr(p, word)) {

char\* c = p; //с-начало подстроки совпадения

p += l\_word; //р-конец подстроки совпадения

// подстрока не в начале строки?

if (c != tekst)

// символ перед подстрокой совпадения не разделитель?

if (\*(c - 1) != ' ') continue;

// символ после слова разделитель?

if ((\*p) == ' ' || (\*p) == '\0' || (\*p)=='.')////

count++; //подстрока – отдельное слово

else continue;/////////

if (count > 1) { udal(tekst, dlina, \*c, l\_word); p -= l\_word;}

}

if (word[0] == '\0') { j++; count = 0; continue; }

//vivod

for (int n = 0; n < l\_word; n++) {

\_putch(word[n]);

}

if (count>9){ \_putch('-'); \_putch(48+(int)count/10); \_putch(48+(int)count%10) ; \_putch('\n');}

else{

\_putch('-'); \_putch(48 + (int)count); \_putch('\n');}

count = 0;

if (tekst[j] == '.') { break; }

j++;

}

//ubirayem probely

for (int x=0; x<dlina-1; x++)

{

if (tekst[x]==' ' && tekst[x+1]==' ' || tekst[0]==' ') {

for (int y=x; y<dlina-1; y++)

{tekst[y]=tekst[y+1];}

x--;

dlina--;}

if(tekst[x]=='.'&&tekst[x-1]==' '){tekst[x-1]=tekst[x];dlina--;}

else continue;

}

const char a[] = "The text is:";

\_putch('\n');

for (int x = 0; a[x] != ':'; x++) { \_putch(a[x]); };

\_putch('\n');

for (int x = 0; tekst[x] != '.'; x++) { \_putch(tekst[x]); }; \_putch('.');

\_getch();

return 0;

}

void vvod(char\* arr, int n)

{

for (int i = 0; i < n; i++)

{

char temp;

temp = \_getch();

if ( (int)temp > 122 && (int)temp != 32 && temp != '\b' && temp!='.' ) {i--; continue;}//iniye simboly

if ( (int)temp < 97 && (int)temp != 32 && temp != '\b' && temp != '.') { i--; continue; }

if (i < 1 && temp == '\b') { i--; continue; }// ne vihodil za predeli teksta

if (temp == '\b' ) { i -= 2; \_putch('\b'); \_putch(' '); \_putch('\b'); continue; } //backspace

arr[i] = temp;

\_putch(arr[i]);

if (arr[i] == '.') { break; }

}

\_putch('\n');

}

void udal(char\* arr, int& dlina, char& nom, int l\_word)

{

int x = 0;

for (x = 0; x < dlina; x++)

{

if (&arr[x] == &nom) { break; };

}

for (x; x < dlina; x++)

{

arr[x - 1] = arr[x + l\_word];// +0 +1

}

dlina -= l\_word;

}

char\* analogstrstr(char\* string, char\* podstring)

{

int n = 0;

char\* str = string, \* p = podstring;

while (\*str != '\0') {

while (\*str == \*p)

{

str++; p++; n++;

if (\*p == '\0') return str - n;

if (\*str == '.')return 0;

}

str++;

n = 0;

p = podstring;

}

return 0;}

**Тестирование**

