|  |  |
| --- | --- |
| 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 *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ**

по дисциплине: «Основы программирования»

|  |  |  |
| --- | --- | --- |
| Студент |  | Камалов Антон Павлович |
| Группа |  | РК6-15Б |
| Тип задания |  | Лабораторная работа №10 |
| Тема лабораторной работы |  | Динамические массивы |

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

*подпись, дата фамилия, и.о.*

Преподаватель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_ \_\_**

*подпись, дата фамилия, и.о.*

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

*Москва, 2022 г.*

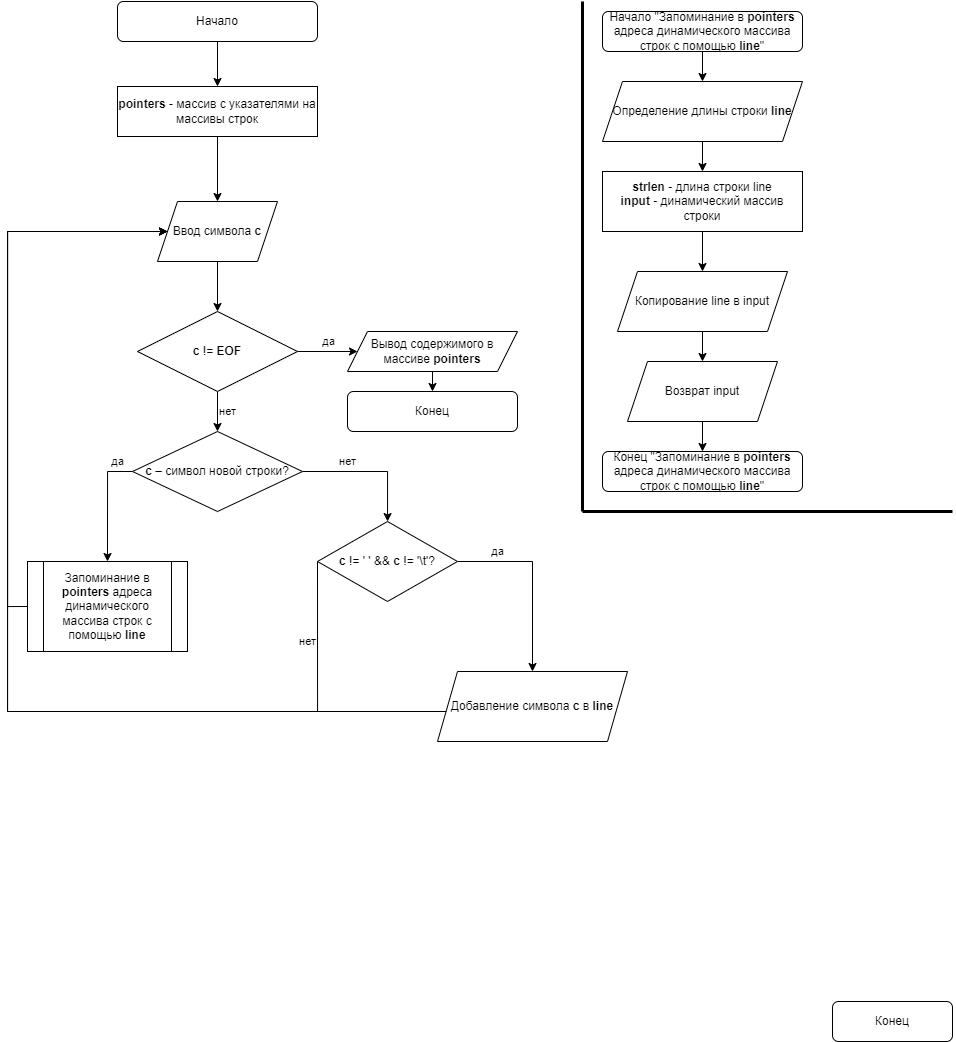
## Цель работы

## Изучение функций управления динамической памятью. Применение этих функций для организации динамических массивов.

## Ответы на контрольные вопросы

1. Указатель - переменная, содержащая адрес.
2. Указателю можно присвоить значение указателя того же типа или адрес объекта того же типа; целое число можно сложить с указателем или вычесть из него; указатель можно сравнивать с другим указателем; указатель можно вычесть из другого указателя; указателю можно присвоить значение нулевого указателя - константы NULL; указатель можно сравнивать с нулевым указателем.
3. Адресная арифметика - арифметические операции над указателями.
4. Тип указателя надо объявлять правильно, чтобы корректно использовать адресную арифметику.
5. Malloc (выделяет память и возвращает адрес начала области памяти, при этом выделенная память не инициируется нулями) calloc (выделяет блок, который вмещает N элементов размером K байтов каждый, возвращает адрес начала области памяти, при этом выделенная память инициируется нулями) и realloc (размер блока по указателю B равным K байтов и возвращает указатель на блок).
6. С помощью функции free освобождается блок памяти и делает его доступным для последующих выделений памяти.

## Блок-схема программы



**Текст программы**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define POINT\_LEN 2048

#define LINE\_LEN 2048

void in (char \* []);

char \* str(char \* );

void out(char \* []);

void clear(char \* []);

int main() {

char \* pointers[POINT\_LEN];

in (pointers);

out(pointers);

clear(pointers);

}

void in (char \* pointers[]) {

char line[LINE\_LEN], c;

int i, count;

i = 0, count = 0;

while ((c = getchar()) != EOF) {

if (c == '\n') {

line[i] = '\0';

i = 0;

pointers[count++] = str(line);

} else {

if (c != '\t' && c != ' ') {

line[i++] = c;

}

}

}

if (i > 0 && strlen(line) != 0) {

line[i] = '\0';

i = 0;

pointers[count++] = str(line);

}

}

char \* str(char \* line) {

char \* input;

int stl;

stl = strlen(line);

if ((input = (char \* ) malloc(stl + 1)) == NULL) {

printf("MEMORY OFF");

exit(-1);

}

strcpy(input, line);

return input;

}

void out(char \* pointers[]) {

printf("\n\nOutput:\n");

for (int i = 0; i < POINT\_LEN; i++) {

if (pointers[i] == NULL) break;

printf("%s\n", pointers[i]);

}

}

void clear(char \* pointers[]) {

for (int i = 0; i < POINT\_LEN; i++) {

if (pointers[i] == NULL) break;

free(pointers[i]);

}

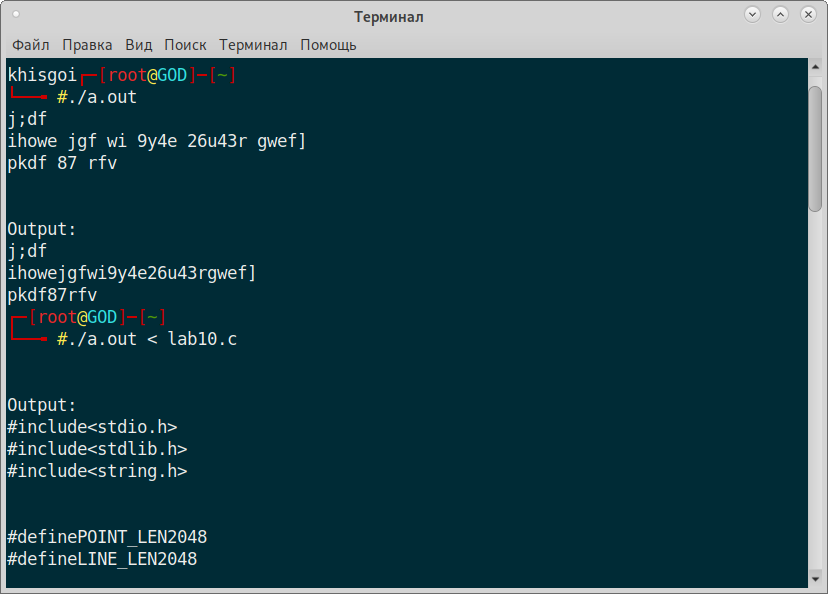
}

## 

## Результаты отладки

После первой компиляции была найдена синтаксическая ошибка: пропущена «;» после инициализации переменной ***stl***.

## Результаты работы программы



## Выводы по работе

Я изучил функции управления динамической памятью и применил их для организации динамических массивов.