**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»**

Московский институт электроники и математики им. А.Н. Тихонова

Департамент прикладной математики

# Отчёт
# по лабораторной работе №6
# по курсу «Алгоритмизация и программирование»

| ФИО студента | Номер группы | Дата |
|---|---|---|
| Вязов Глеб Дмитриевич | БПМ-231 | 16.12.2023 |

**Москва, 2023**

# Задание (вариант №7)

Размер динамического массива вводится пользователем на этапе выполнения. Тип массива указан в задании. Элементы массива вводятся с клавиатуры. Написать функции заполнения массива и вывода массива. Написать функцию модификации массива указанных элементов. Вспомогательные массивы не использовать.

Тип массива – double. Удалить все положительные элементы, расположенные между первым максимальным и последним минимальным элементами.

# Решение

Листинг 1: C

```c
#include <stdio.h>
#include <stdlib.h>

// Функция возвращает индекс первого максимального элемента
int first_max_index(double *array, int k) {
    double max = array[0];
    int max_index = 0;

    for (int i=1; i<k; i++) {
        if (array[i] > max) {
            max = array[i];
            max_index = i;
        }
    }

    return max_index;
}

// Функция возвращает индекс последнего минимального элемента
int last_min_index(double *array, int k) {
    double min = array[0];
    int min_index = 0;

    for (int i=1; i<k; i++) {
        if (array[i] <= min) {
            min = array[i];
            min_index = i;
        }
    }

    return min_index;
}

// Функция сдвигает элементы на один влево, начиная с index
void move_elements(double *array, int k, int index) {
    for(int i=index; i<k-1; i++) {
```

```
37          array[i] = array[i + 1];
38      }
39  }
40
41  // Функция удаляет элементы
42  // Возвращает указатель на новое место в памяти
43  double* delete(double *array, int k, int* new_size) {
44      int first = first_max_index(array, k);
45      int last = last_min_index(array, k);
46      int count_delete_elements = 0;
47      // Если первый индекс указывает на один из двух последних элементов
            (k-1 или k-2)
48      // Или последний индекс указывает на один из двух первыв элементов (0
            или 1)
49      // То в этом массиве ничего удалять не нужно
50      if (first >= k-2 || last <= 1) {
51          *new_size = k;
52          return array;
53      }
54
55      int index = first+1;
56      while (index < last) {
57          if (array[index] <= 0) {
58              index++;
59              continue;
60          }
61
62          move_elements(array, k, index);
63          last--;
64          count_delete_elements++;
65      }
66
67      *new_size = k - count_delete_elements;
68      array = realloc(array, (*new_size) * sizeof(double));
69      return array;
70  }
71
72  // Выделение памяти для массива из k элементов
73  double* get_memory(int* k) {
74      printf("Введите длинумассива : ");
```

```
75      scanf("%d", k);
76
77      // Выделение памяти для k элементов, размерности sizeof(double)
78      // И инициализируем всё нулями
79      double *array = calloc(*k, sizeof(double));
80      return array;
81  }
82
83  // Заполнение массива, длиной k, вещественными элементами
84  double * create_array(double *array, int k) {
85      for (int i=0; i<k; i++) {
86          scanf("%lf", &array[i]);
87      }
88      return array;
89  }
90
91  // Вывод массива длиной k с вещественными элементами
92  void print_array(double *array, int k) {
93      for (int i=0; i<k; i++) {
94          printf("%10.4lf\t", array[i]);
95      }
96  }
97
98  int main() {
99      // Меняем кодировку на UTF-8, чтобы можно было писать на русском
100     system("chcp 65001");
101     // Ввод переменных. Дружественный интерфейс
102     printf("Выполнил задание: ВязовГлеб . Группа: БПМ231\n");
103
104     unsigned int k;
105     double *array = get_memory(&k);
106
107     // Если память не выделилась – возвращаем ошибку
108     if (array == NULL) {
109         return 1;
110     }
111     create_array(array, k);
112     printf("Вы создалимассив : ");
113     print_array(array, k);
114
```

```
115     int new_size = k;
116     double* new_array = delete(array, k, &new_size);
117 
118     // После realloc память может выделиться некорректно
119     if (array == NULL) {
120         return 1;
121     }
122 
123     printf("\Преобразованныйn массив: ");
124     print_array(new_array, new_size);
125 
126     return 0;
127 }
```

# Тестирование

1. **Тест №1.**

   *Ввод:* Длина массива: 5 Элементы массива: 100, 1, 2, 3, -100

   *Вывод:*

```
Вы создали массив:   100.0000       1.0000          2.0000          3.0000       -100.0000
Преобразованный массив:   100.0000       -100.0000
Process finished with exit code 0
```

2. **Тест №2.**

   *Ввод:* Длина массива: 5 Элементы массива: -100, 1, 2, 3, 100

   *Вывод:*

```
Вы создали массив:  -100.0000       1.0000          2.0000          3.0000        100.0000
Преобразованный массив:  -100.0000          1.0000          2.0000          3.0000        100.0000
Process finished with exit code 0
```

3. **Тест №3.**

   *Ввод:* Длина массива: 5 Элементы массива: 100, 100, 1, -1, -100

   *Вывод:*

```
Вы создали массив:   100.0000     100.0000          1.0000         -1.0000       -100.0000
Преобразованный массив:   100.0000         -1.0000       -100.0000
Process finished with exit code 0
```