

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет ИТМО

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

**Дополнительная работа №1**


По дисциплине «Аппаратное обеспечение вычислительных систем»

Выполнил студент группы №М3113

*Балакирева Виктория Валерьевна*

Проверил

*Шевчик Софья Владимировна*



**УНИВЕРСИТЕТ ИТМО**

Санкт-Петербург

2024

### **Условие**

Создать программу на языке C, которая считывает из файла целые числа.

Необходимо написать ассемблерную вставку, осуществляющую сортировку массива. Метод сортировки определяет разработчик. Вставка должна быть вынесена в отдельную функцию. Использовать глобальные переменные для передачи данных в указанную функцию запрещено.

Полученный массив записать в файл вывода.

### **Входные данные**

Первая строка входного файла INPUT.TXT содержит одно целое число  $N$  ( $1 \leq N \leq 100$ ).

В следующих  $N$  строках содержатся числа, по модулю не превышающие  $10^9$ .

На следующей странице представлена демонстрация реализации ассемблерной вставки, осуществляющая сортировку пузырьком массива из  $N$  элементов

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // Функция пузырьковой сортировки, использующая ассемблерные вставки
5  void bubble_sort(int *array, int n) {
6      __asm__ (
7          "subs %w1, %w1, #1\n"      // Декрементируем n уменьшаем значение регистра на 1 слож в w1
8          "1:\n"                    // Метка внешнего цикла
9          "mov w5, #0\n"            // Обнуляем w5 (для отслеживания обменов) конст
10         "mov x3, #0\n"             // Копируем указатель на массив в регистр (замены параметров на регистры или значения)
11         "mov w4, %w1\n"           // Загружаем w1(н) в w4 для внутреннего цикла копируем значение регистра w1 (модифицированного n) в регистр w4.
12         "2:\n"                    // Метка внутреннего цикла
13         "ldn w1, [x3]\n"          // Загружаем адрес текущего значения x3 в w1
14         "ldn w2, [x3, #4]\n"      // Адрес следующего элемента массива (указатель в x3 плюс 4 байта)
15         "cmp w1, w2\n"            // Сравниваем текущий и следующий элементы взаависимости от этого меняются флаги
16         "ble 3f\n"               // Переходим 3 метке, если текущий элемент <= следующего если w1 мен тогда
17         "str w2, [x3]\n"         // Меняем элементы местами записываем элемент w2 след по адресу тек элемента
18         "str w1, [x3, #4]\n"      // записываем значение следующего
19         "mov w5, #1\n"           // Указываем, что был произведен обмен поэтому в 1 конст
20         "3:\n"
21         "add x3, x3, #4\n"       // Переходим к следующему элементу (int = 4 байта) сложение того что лежит в 4 байт
22         "subs w4, w4, #1\n"      // Уменьшаем счетчик внутреннего цикла
23         "bne 2b\n"              // Продолжаем внутренний цикл к 2, если счетчик не равен нулю
24         "cbz w5, 4f\n"          // Проверим, были ли обмены, если нет, выходим, если w5=0 перход к 4
25         "subs %w1, %w1, #1\n"    // Уменьшаем n для следующей итерации внешнего цикла
26         "bne 1b\n"              // Продолжаем внешний цикл, если счетчик не равен нулю
27         "4:\n"
28         :
29         : "r"(array), "r"(n)
30         : "x1", "x2", "x3", "x4", "x5", "memory"
31       );
32   }
33

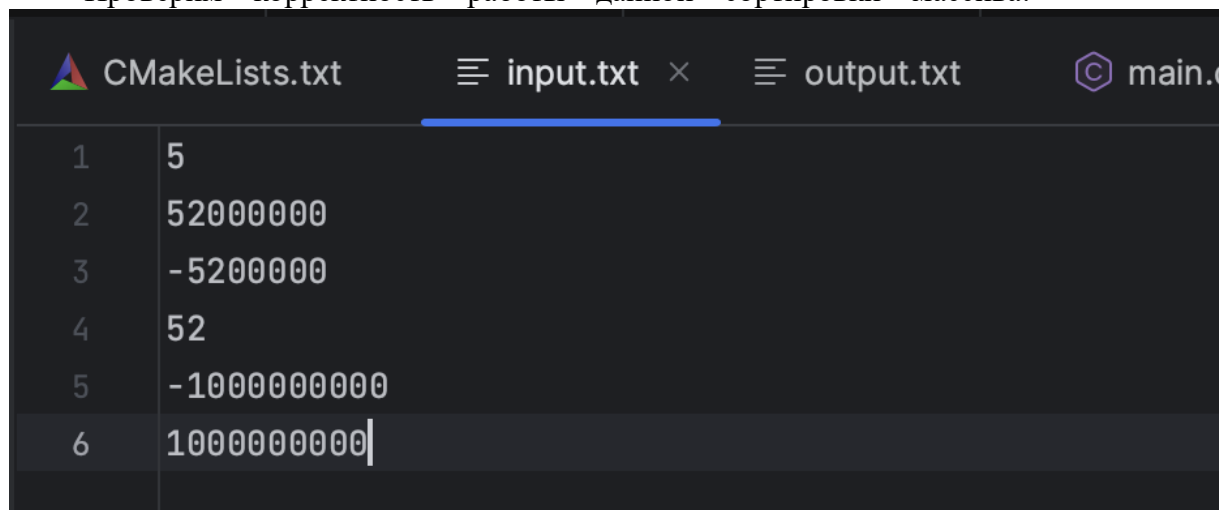
```

```

34 int main() {
35     FILE *input_file = fopen(filename "input.txt", "r"); // Открываем файл input.txt для чтения
36     if (input_file == NULL) { // Проверим, удалось ли открыть файл
37         printf("can't open input.txt\n");
38         return 1; // Выходим с ошибкой, если файл не удалось открыть
39     }
40     FILE *output_file = fopen(filename "output.txt", "w"); // Открываем файл output.txt для записи
41     if (output_file == NULL) { // Проверим, удалось ли открыть файл
42         printf("can't open output.txt\n");
43         fclose(input_file); // Закрываем входной файл перед выходом
44         return 1; // Выходим с ошибкой, если файл не удалось открыть
45     }
46
47     int n;
48     fscanf(input_file, "%d", &n); // Считываем количество элементов из входного файла
49
50     int *array = malloc(sizeof(n) * sizeof(int)); // Выделяем память для массива
51     if (array == NULL) {
52         printf("memory allocation failed\n");
53         fclose(input_file);
54         fclose(output_file);
55         return 1;
56     }
57
58     // Считываем элементы массива из входного файла
59     for (int i = 0; i < n; i++) {
60         fscanf(input_file, "%d", &array[i]);
61     }
62
63     bubble_sort(array, n); // Сортируем массив
64
65     // Записываем отсортированные элементы в выходной файл
66     for (int i = 0; i < n; i++) {
67         printf("%d ", array[i]);
68     }
69
70     int n;
71     fscanf(input_file, "%d", &n); // Считываем количество элементов из входного файла
72
73     int *array = malloc(sizeof(n) * sizeof(int)); // Выделяем память для массива
74     if (array == NULL) {
75         printf("memory allocation failed\n");
76         fclose(input_file);
77         fclose(output_file);
78         return 1;
79     }
80
81     // Считываем элементы массива из входного файла
82     for (int i = 0; i < n; i++) {
83         fscanf(input_file, "%d", &array[i]);
84     }
85
86     bubble_sort(array, n); // Сортируем массив
87
88     // Записываем отсортированные элементы в выходной файл
89     for (int i = 0; i < n; i++) {
90         printf("%d ", array[i]);
91     }
92
93     // Освобождаем память, выделенную для массива
94     free(array);
95
96     // Закрываем файлы
97     fclose(output_file);
98     fclose(input_file);
99
100    return 0; // Завершаем программу
101 }

```

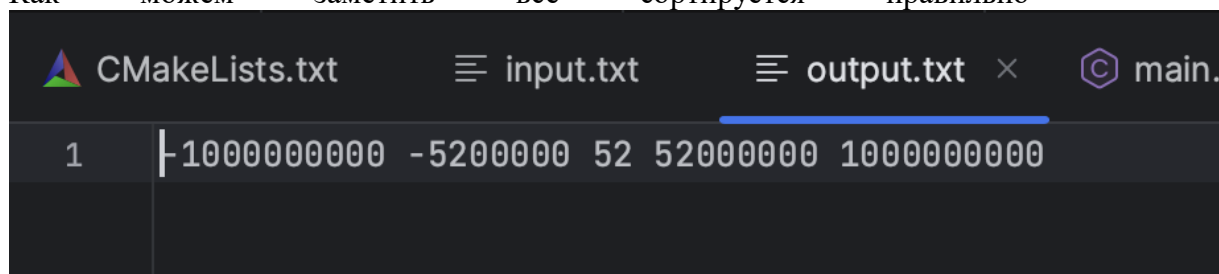
Проверим корректность работы данной сортировки массива:



The screenshot shows a code editor with three tabs: 'CMakeLists.txt', 'input.txt', and 'output.txt'. The 'input.txt' tab is active and contains the following text:

Line	Value
1	5
2	52000000
3	-5200000
4	52
5	-1000000000
6	1000000000

Как можем заметить все сортируется правильно



The screenshot shows the same code editor with the 'output.txt' tab active. It contains the following text:

Line	Value
1	1000000000 -5200000 52 52000000 1000000000