**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»

Вариант № -

Выполнил студент группы №М3105

*Тросько Виктория Игоревна*

Проверил

*Кулешова Екатерина Дмитриевна*



Санкт-Петербург

2024

**Условие**

Создать программу на языке С, 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 ≤ N ≤ 100).

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

**Текст программы**

#include <iostream>

#include <fstream>

#include <cstdlib>

using namespace std;

extern "C" void bubbleSort(int\*, int);

int main() {

ifstream input("C:/Users/suiremon/Documents/ИТМО/ЭВМ/doptaski/INPUT.TXT");

if (!input.is\_open()) {

cerr << "Failed to open input file." << endl;

return 1;

}

int v\_size;

input >> v\_size;

int\* v = new int[v\_size];

int a;

for (int i = 0; i < v\_size; ++i) {

input >> v[i];

}

bubbleSort(v, v\_size);

ofstream out("C:/Users/suiremon/Documents/ИТМО/ЭВМ/doptaski/OUTPUT.TXT");

for (int i = 0; i < v\_size; ++i) {

out << v[i] << '\n';

}

delete[] v;

return 0;

}

extern "C" void bubbleSort(int\* a, int n) {

\_asm {

xor esi, esi //обнуляем входной регистр

mov ecx, n //есх - счетчик, будет отвечать за длину массива

dec ecx //уменьшили на 1, тк индексация с 0

mov eax, a //записали в А первое значение массива

\_main\_for\_ : //начинаем цикл

cmp esi, ecx //сравниваем индекс текущего элемента и последнего, чтобы не выйти за границы

jge \_end\_main\_for //если esi >= ecx, то выходим из цикла

xor edi, edi //обнуляем приемник, он нужен для внутреннего цикла

mov ebx, ecx //записываем в базу индекс конца массива

sub ebx, esi //ebx = n - i - 1, база отвечает за оставшийся кусок массива

\_inside\_for\_ : //начинаем внутренний цикл

cmp edi, ebx //сравниваем приемник с остатком до конца массива, идем от 0 до оставшегося куска

jge \_end\_inside\_for //если edi >= ebx, то выходим из внутреннего цикла

push ecx //сохраняем в стек индекс конца массива

push edx //сохраняем в стек значение регистра данных

mov ecx, [eax + 4 \* edi] //сохраняем i элемент массива

mov edx, [eax + 4 \* edi + 4] //сохраняем i+1 элемент массива

cmp ecx, edx //сравниваем соседние элементы

jle \_end\_neigh\_if //если a[i] < a[i + 1], то переходим к метке \_end\_if

mov[eax + 4 \* edi + 4], ecx //если же a[i] >= a[i + 1],

mov[eax + 4 \* edi], edx //то свапаем соседние элементы

pop edx //возвращаем значение регистра данных

pop ecx //возвращаем индекс последнего элемента

inc edi //переходим к следующему элементу

jmp \_inside\_for\_ //возвращаемся к началу внутреннего цикла

\_end\_neigh\_if : //метка, куда попадаем, если порядок не нарушен

pop edx //возвращаем значение регистра данных

pop ecx //возвращаем индекс последнего элемента

inc edi //переходим к следующему элементу

jmp \_inside\_for\_ //возвращаемся к началу внутреннего цикла

\_end\_inside\_for : //метка конца внутреннего цикла

inc esi //переходим к следующему элементу внешнего цикла

jmp \_main\_for\_ //заново запускаем цикл

\_end\_if :

\_end\_main\_for: //метка конца внешнего цикла

}

}

\*Псевдокод на Python

arr = list(map(int, input().split()))

for i in range(len(arr)):

for j in range(0, len(arr) - i - 1):

if arr[j] >= arr[j+1]:

arr[j], arr[j+1] = arr[j+1], arr[j]

print(arr)