МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ЭЛЕКТРОТЕХНИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА «ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ»

ОТЧЁТ
«СОРТИРОВКА ПУЗЫРЬКОМ»

Дисциплина: «Программирование»

Выполнил:
Студент группы ИВТ-21-2б
Безух Владимир Сергеевич

Проверил:
Доцент кафедры ИТАС
Полякова Ольга Андреевна

Пермь, 2022

Содержание

1. Постановка задачи

Написать функцию для сортировки массива с помощью пузырькового метода.

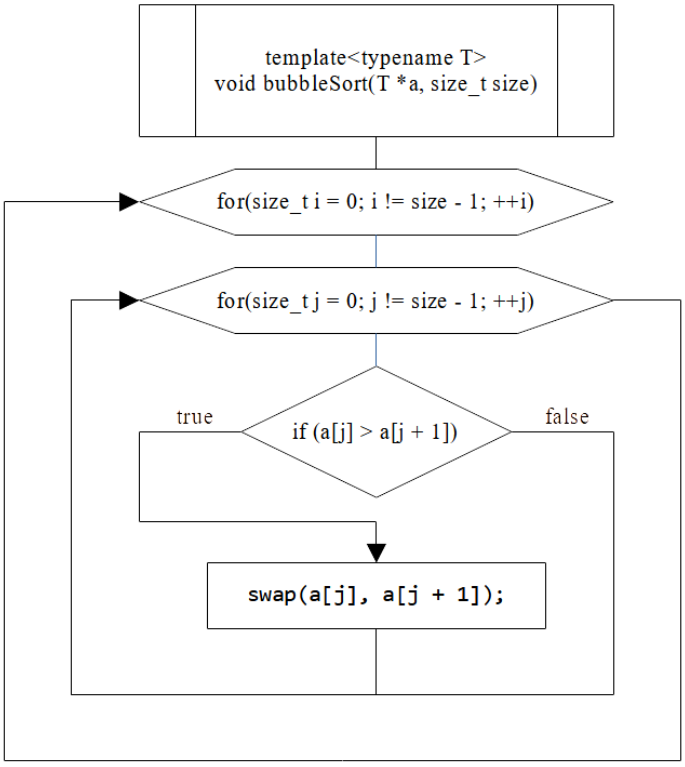
2. Анализ задачи

Алгоритм состоит в повторяющихся проходах по сортируемому массиву. На каждой итерации последовательно сравниваются соседние элементы, и, если порядок в паре неверный, то элементы меняют местами. За каждый проход по массиву как минимум один элемент встаёт на своё место, поэтому необходимо совершить не более n - 1 проходов, где n — размер массива, чтобы отсортировать массив.

3. Описание переменных

void bubbleSort(T *a, size_t size) { … } — сортируемый массив и количество элементов в массиве.

4. Блок-схемы


template<typename T>
void bubbleSort(T *a, size_t size)
for(size_t i = 0; i != size - 1; ++i)
for(size_t j = 0; j != size - 1; ++j)
true
if (a[j] > a[j + 1])
false
swap(a[j], a[j + 1]);

5. Исходный код

```
#include <iostream>

using namespace std;

template<typename T>
void bubbleSort(T* a, size_t size)
{
    for (size_t i = 0; i != size - 1; ++i)
        for (size_t j = 0; j != size - 1; ++j)
            if (a[j] > a[j + 1])
                swap(a[j], a[j + 1]);
}

int main()
{
    size_t n; cin >> n;
    int* array = new int[n];
    for (size_t i = 0; i != n; ++i)
        cin >> array[i];

    bubbleSort(array, n);

    for (size_t i = 0; i != n; ++i)
        cout << array[i] << ' ';

    return 0;
}
```

6. Консольный интерфейс программы

```
13
14  int main()
15  {
16      size_t n; cin >> n;
17      int *array = new int[n];
18      for (size_t i = 0; i != n; ++i)
19          cin >> array[i];
20
21      bubbleSort(array, n);
22
23      for (size_t i = 0; i != n; ++i)
24          cout << array[i] << ' ';
25
26      return 0;
27  }
```

```
7
21 54 77 21 5 -5 -750
-750 -5 5 21 21 54 77
```

7. Анализ результатов

Можно заметить, что после i-ой итерации внешнего цикла i последних элементов уже находятся на своих местах в отсортированном порядке, поэтому нет необходимости производить их сравнения друг с другом. Следовательно, внутренний цикл можно выполнять не до n - 2, а до n - i - 2.

Также заметим, что если после выполнения внутреннего цикла не произошло ни одного обмена, то массив уже отсортирован, и продолжать что-то дальше бессмысленно. Поэтому внутренний цикл можно выполнять не n - 1 раз, а до тех пор, пока во внутреннем цикле происходят обмены.

Эти оптимизации не влияют на итоговую сложность алгоритма.