Пермский национальный исследовательский политехнический университет ПНИПУ

Отчёт по лабораторной работе на тему “Сортировки”

Выполнил студент группы РИС-23-3Б:

Смирнов Андрей Сергеевич

Проверила:

Доцент кафедры ИТАС

О.А. Полякова

2024

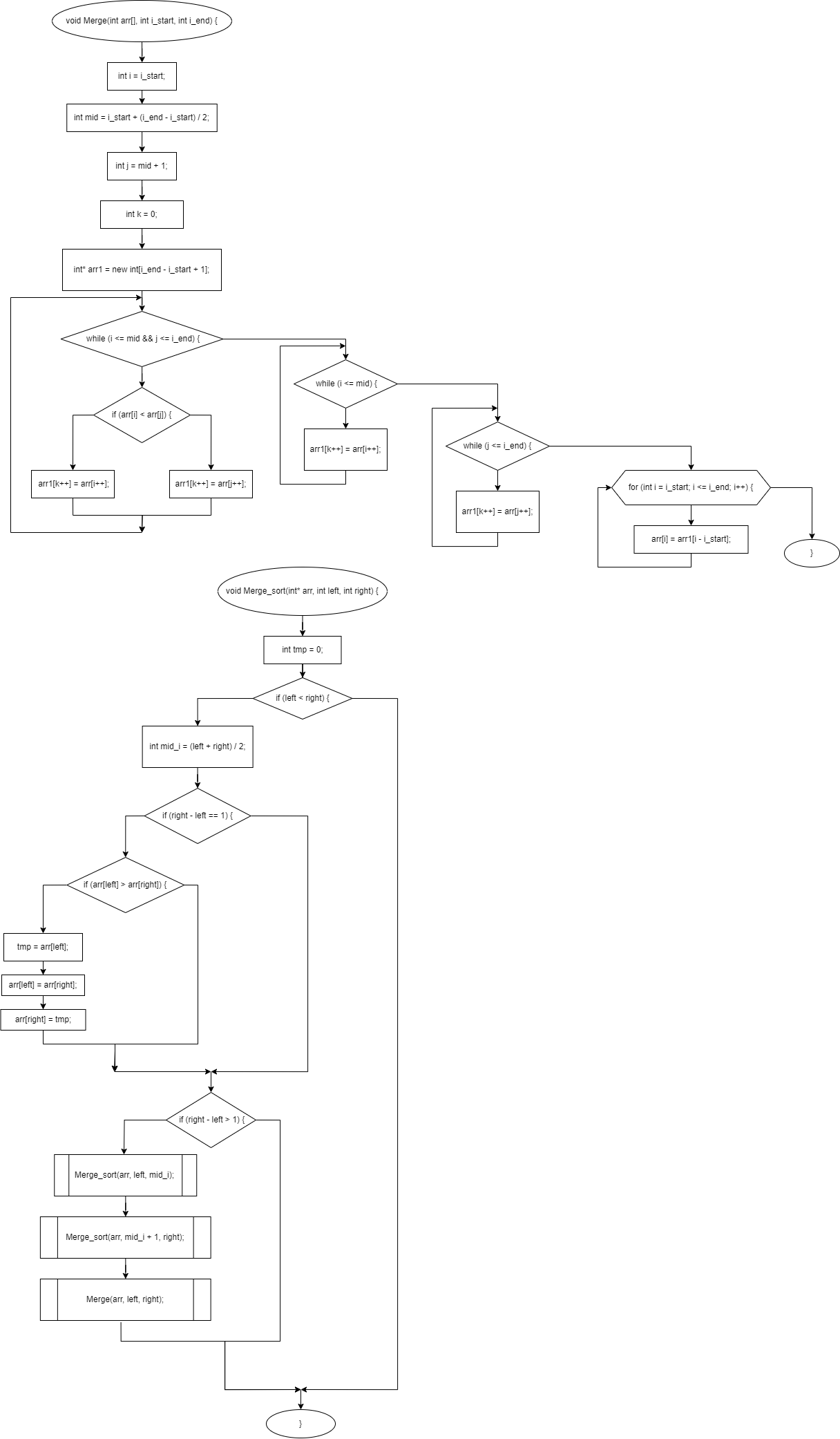
1. Постановка задачи:

Разработать сортировку “Слиянием” и проверить её на массиве размером 25 элементов.

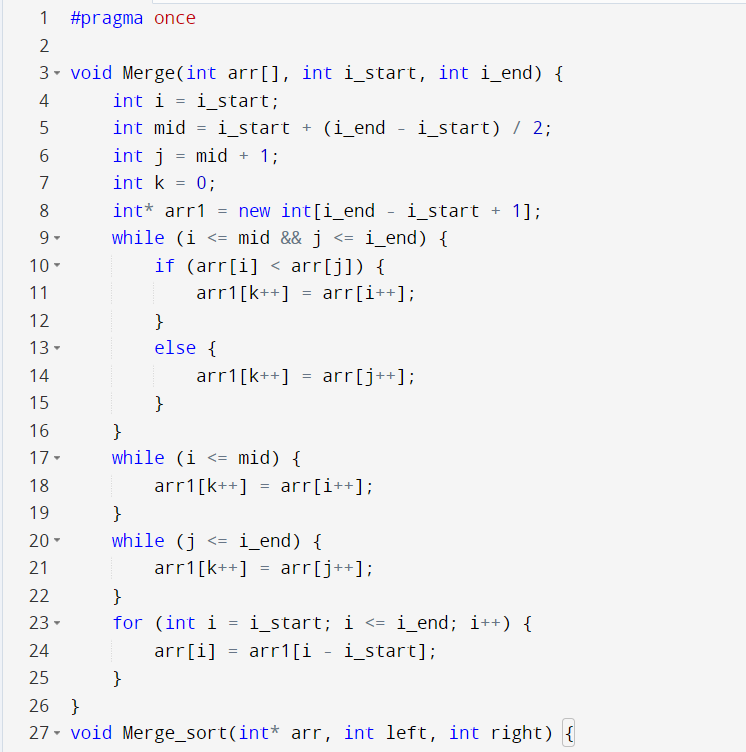
1. Анализ задачи:

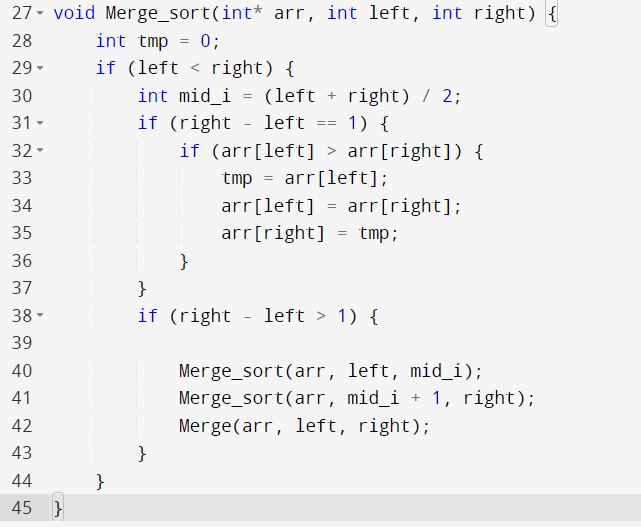
* Создаём отдельный заголовочный файл для функции
* Разделяем сортировку на 2 функции: слияние массивов и саму сортировку перед слиянием.
* На вход функции слияния подаётся массив, индекс первого элемента и индекс последнего элемента массива
* Создаётся новая переменная, означающая серединный индекс массива
* Через циклы while изначальный массив разбивается единичные элементы, которые после “сливаются” в один отсортированный массив.

1. Блок-схема:

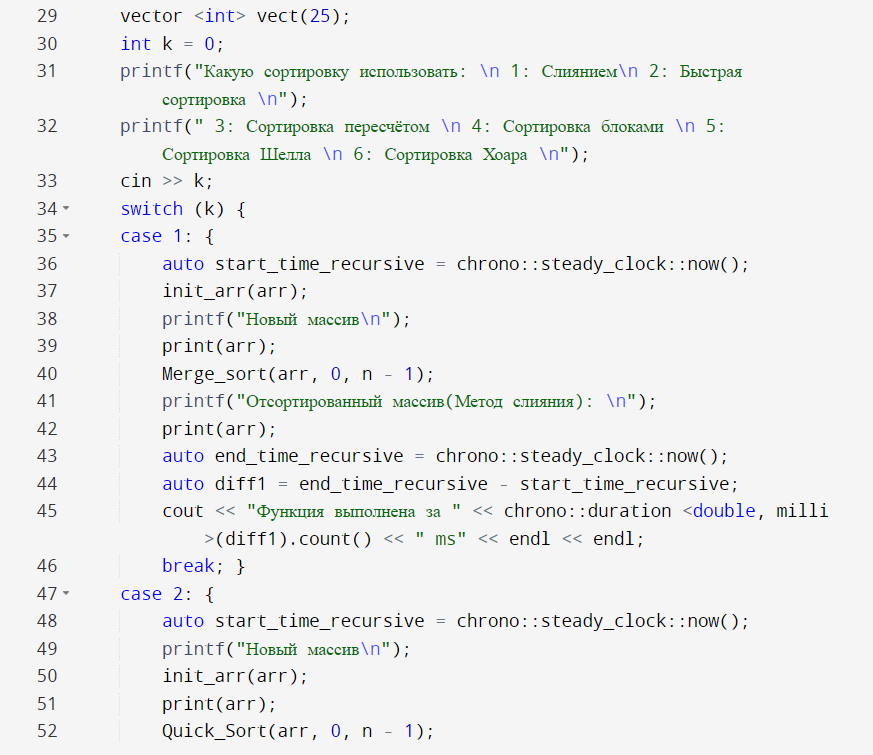
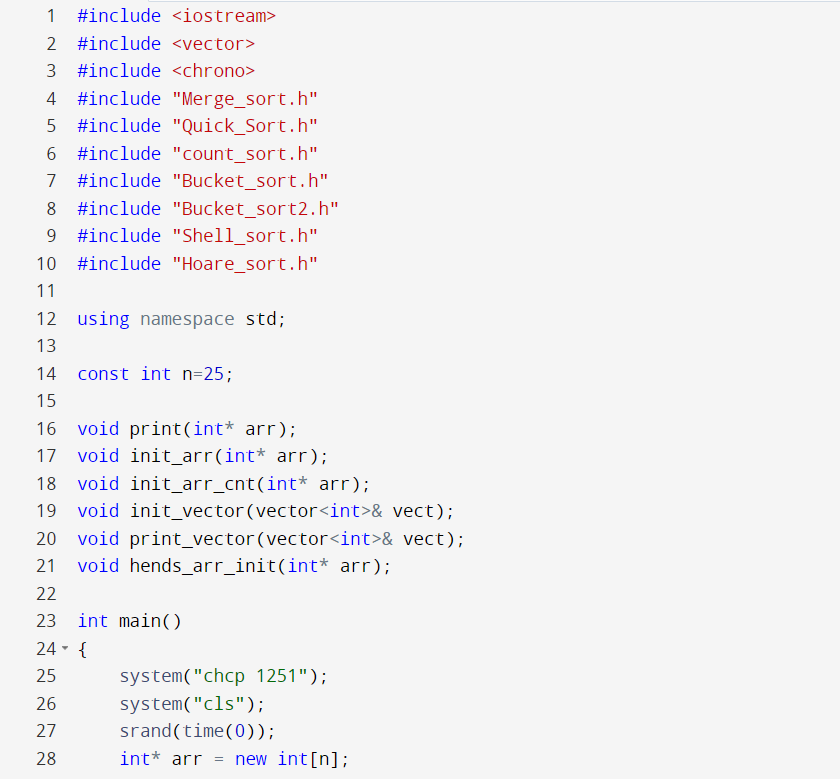


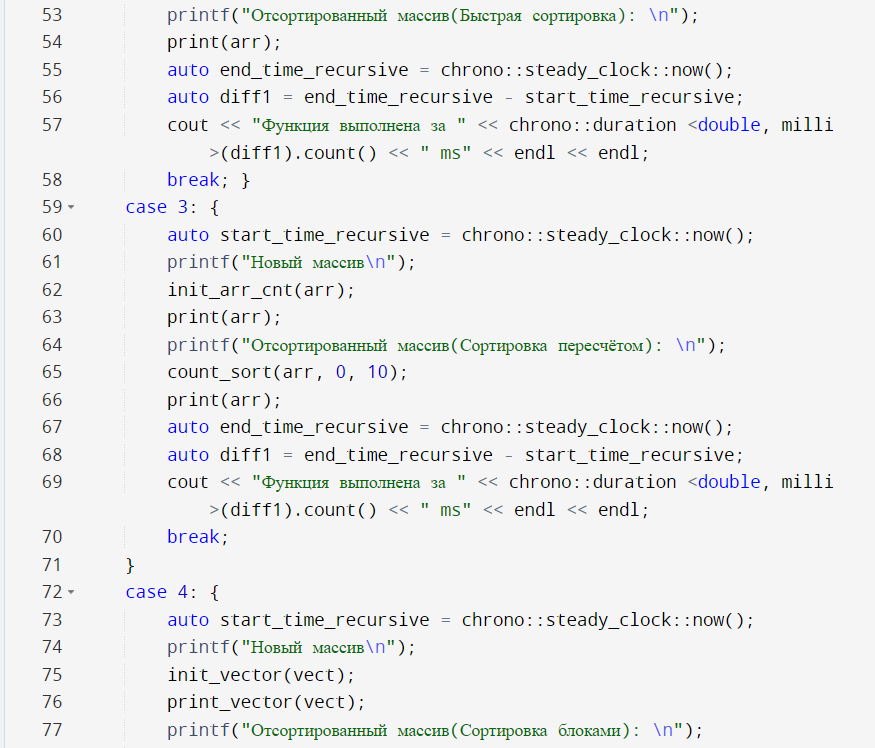
1. Код:

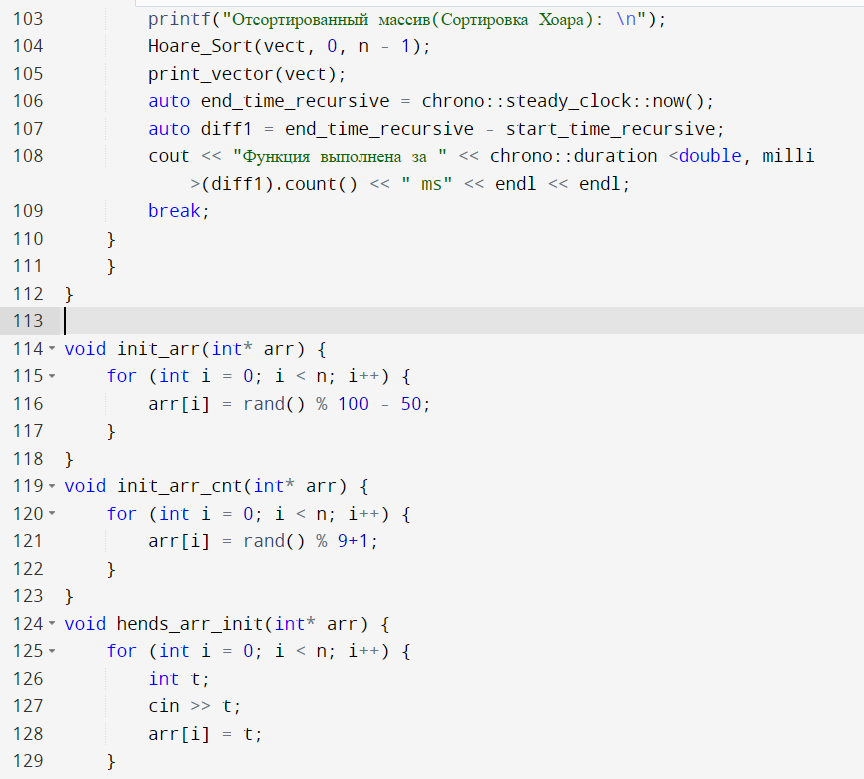
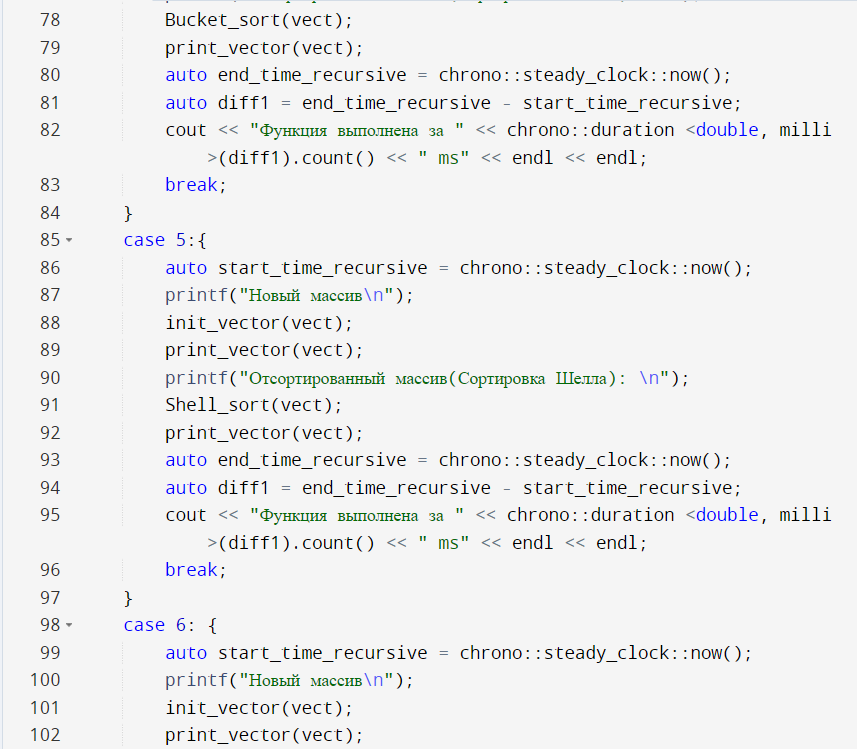


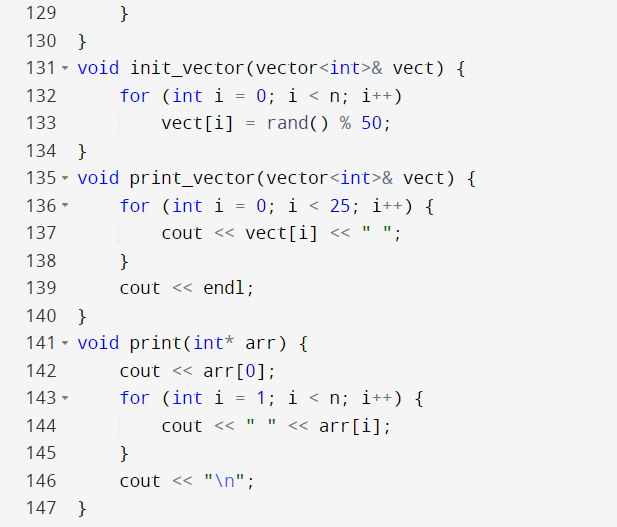


Main:

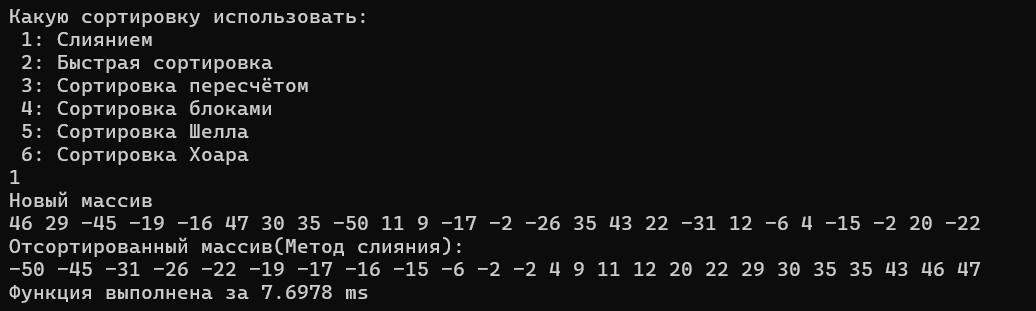








1. Результат работы:



1. Вывод:

Сортировка слиянием удобна для небольших массивов с неповторяющимися числами.

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>

Быстрая сортировка

1. Постановка задачи:

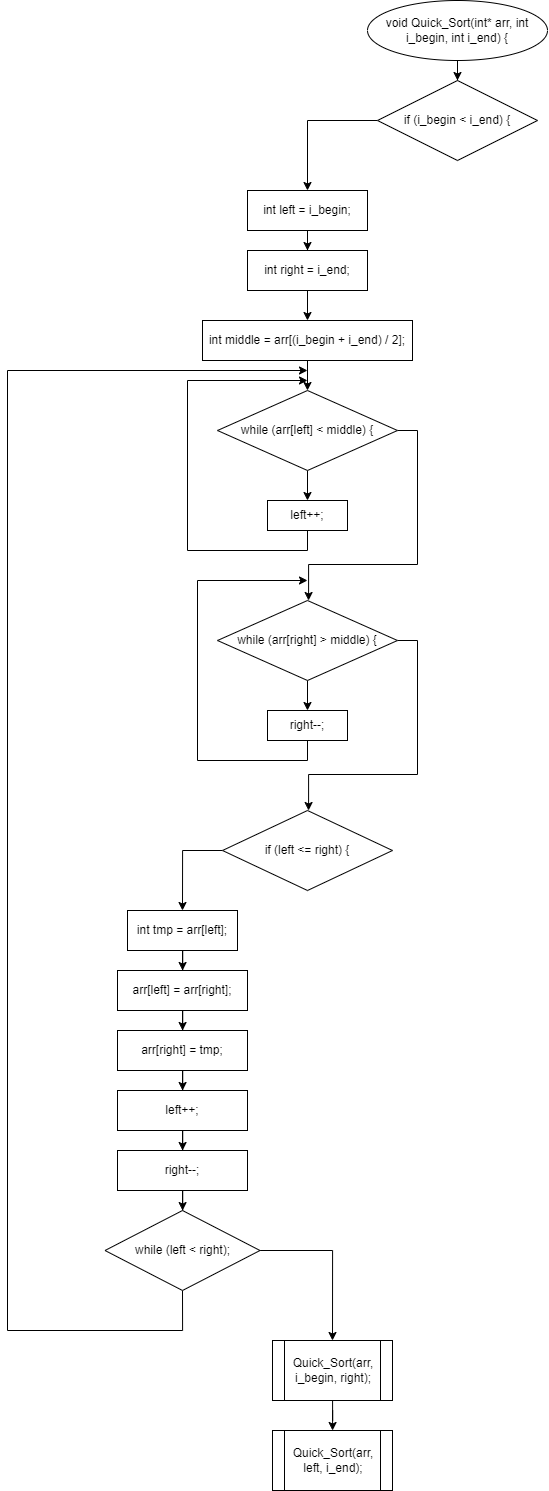
Разработать “Быструю” сортировку и проверить её на массиве размером 25 элементов.

1. Анализ задачи:

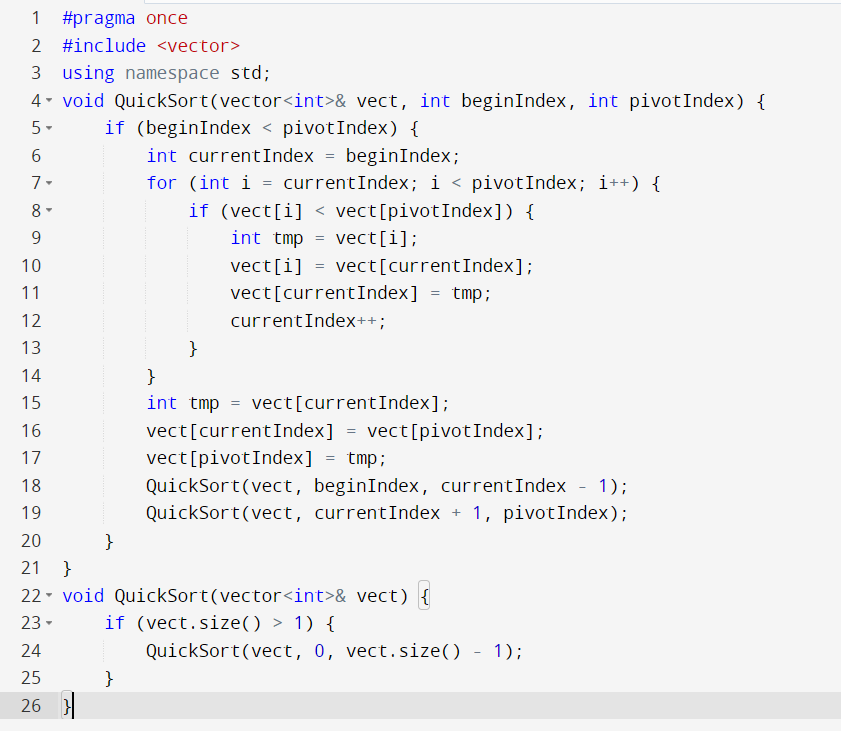
* Создаём отдельный заголовочный файл для функции
* В функцию передаём массив, у которого в начале и середине стоят два указателя
* Все числа меньше центрального перемещаются влево, остальные остаются на месте
* В последующем массив рекурсивно разбивается на два и также сортируется
* В результате получаем отсортированный массив

1. Блок-схема:

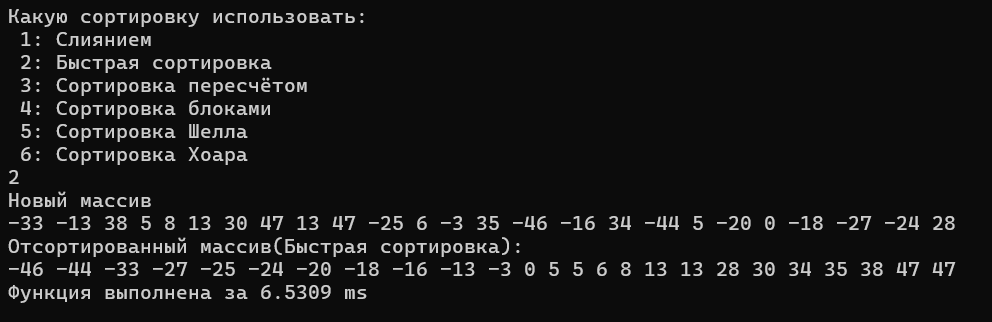
(Блок main не меняется, поэтому блок схема прикреплена выше по отчёту)



1. Код:



1. Результат работы:



1. Вывод:

Быстрая сортировка работает за меньшее время по сравнению с другими, но неэффективна на больших массивах.

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>

Сортировка подсчётом

1. Постановка задачи:

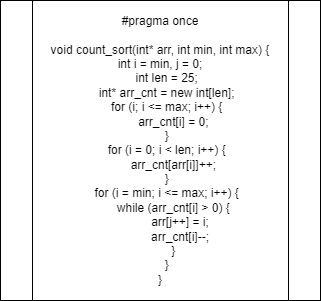
Разработать сортировку “Подсчётом” и проверить её на массиве размером 25 элементов.

1. Анализ задачи:

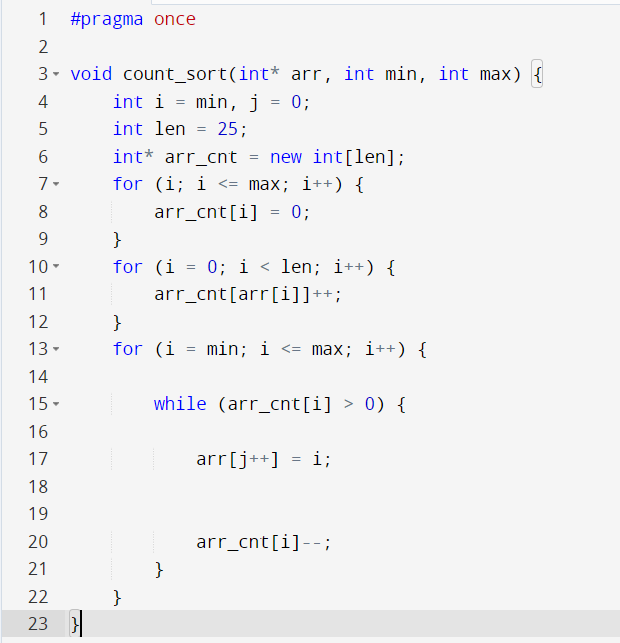
* Создаём отдельный заголовочный файл для функции
* Создаётся цикл прохождения по массиву, где при нахождении числа в этом индексе число увеличивается на 1
* В результате получаем отсортированный массив

1. Блок-схема:

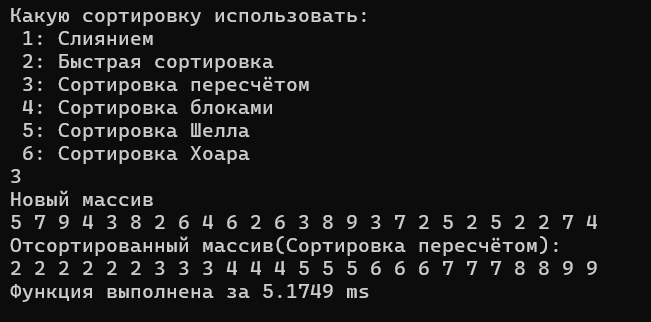
(Блок main не меняется, поэтому блок схема прикреплена выше по отчёту)



1. Код:



1. Результат работы:



1. Вывод:

Сортировка эффективна при небольших диапазонах чисел в массиве, но работает очень медленно по сравнению с остальными сортировками на стандартных диапазонах чисел.

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>

Сортировка подсчётом

1. Постановка задачи:

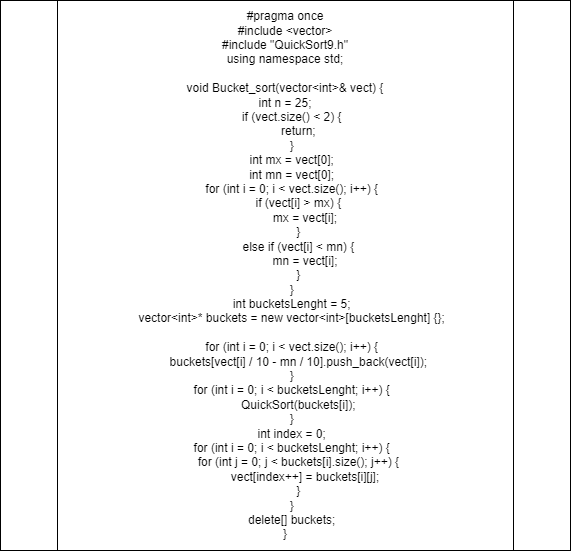
Разработать Блочную сортировку и проверить её на массиве размером 25 элементов.

1. Анализ задачи:

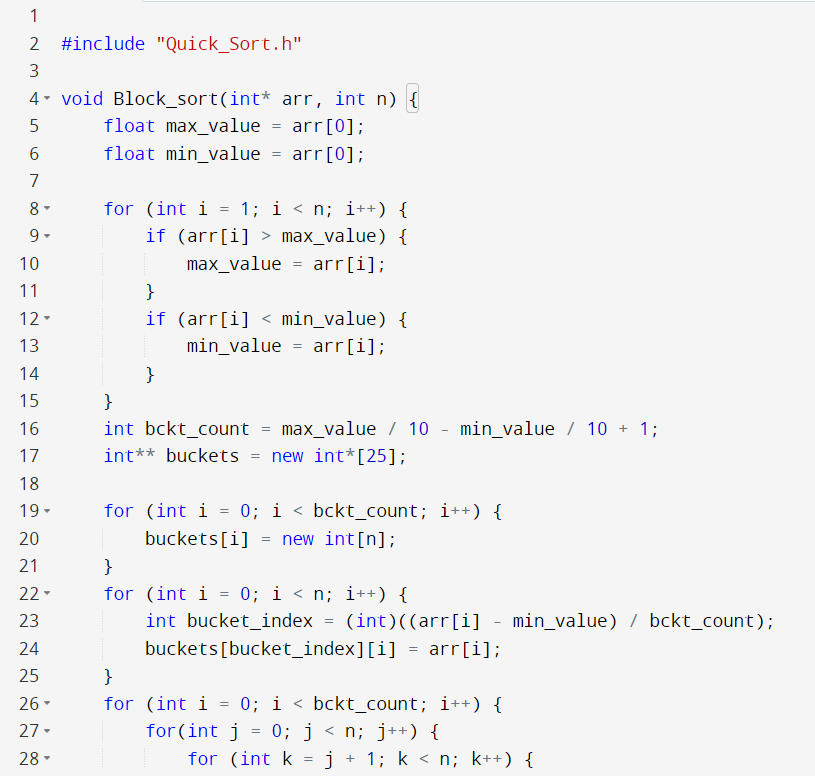
* Создаём отдельный заголовочный файл для функции
* В данной сортировке используется быстрая сортировка, т.к. изначальный массив разбивается на под массивы, которые отдельно сортируются
* Диапазон чисел массива разделяется на вычисленные по формуле блоки, в которых и записываются числа, содержащиеся в изначальном массиве.
* В этих блоках при поступлении числа сортируются
* После разбиения всех чисел массива по блокам, блоки по очереди записываются в изначальный массив.
* В результате получаем отсортированный массив

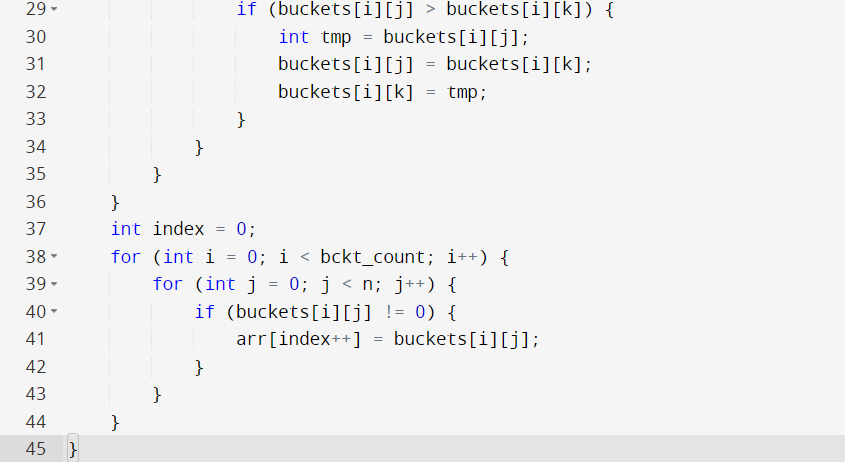
1. Блок-схема:

(Блок main не меняется, поэтому блок схема прикреплена выше по отчёту)

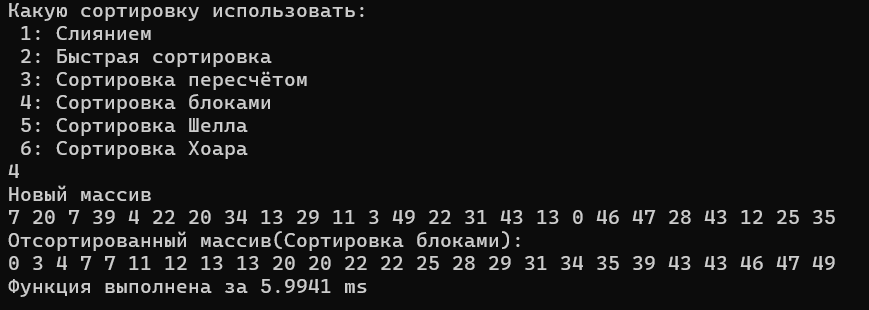


1. Код:





1. Результат работы:



1. Вывод:

В сравнении с другими сортировками, блочная сортировка работает быстрее, но имеет ограничение в диапазоне чисел изначального массива

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>

Сортировка Хоара

1. Постановка задачи:

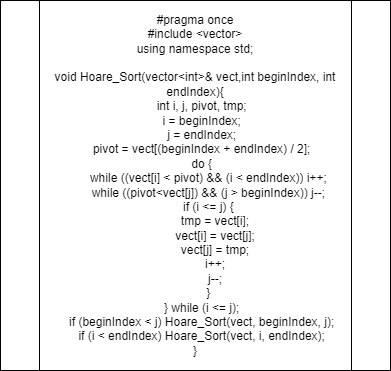
Разработать сортировку “Хоара” и проверить её на массиве размером 25 элементов.

1. Анализ задачи:

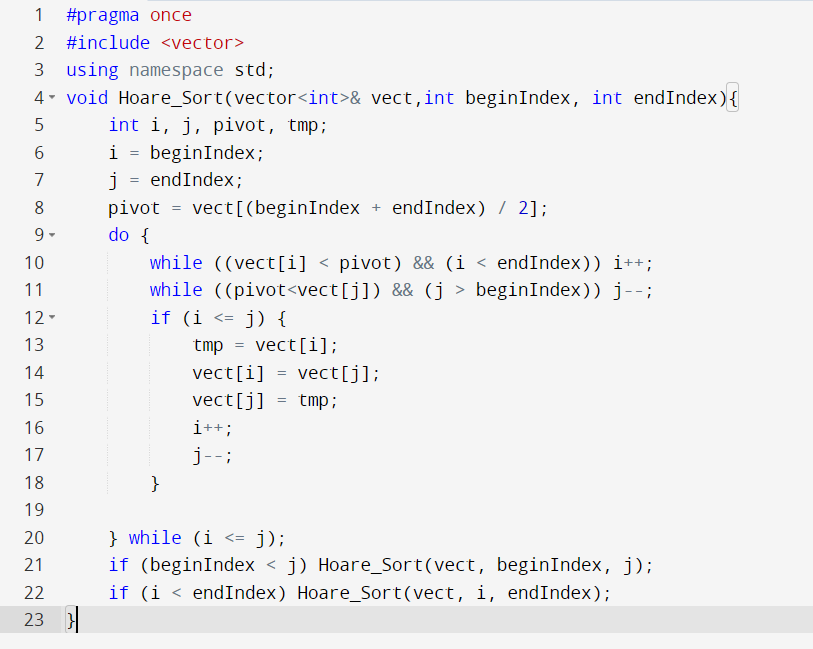
* Создаём отдельный заголовочный файл для функции
* Берётся массив и создаются указатели на первый и последний элемент массива
* Вычисляется средний индекс и все числа, что меньше перемещаются в правую часть массива, все числа что меньше остаются в левой части массива
* Рекурсивно массив делится на всё меньшие части, где также вычисляются новые средние индексы и числа перемещаются в правую и левую части новых массивов меньших размеров
* Так происходит до тех пор, пока изначальный массив не получается отсортированным
* В результате получаем отсортированный массив

1. Блок-схема:

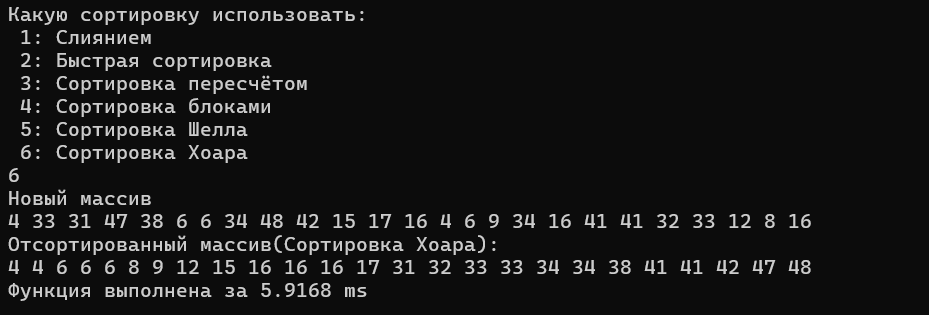
(Блок main не меняется, поэтому блок схема прикреплена выше по отчёту)



1. Код:



1. Результат работы:



1. Вывод:

Время выполнения работы схоже с быстрой сортировкой, т.к. принцип работы очень схож.

1. <https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>

Сортировка Шелла

1. Постановка задачи:

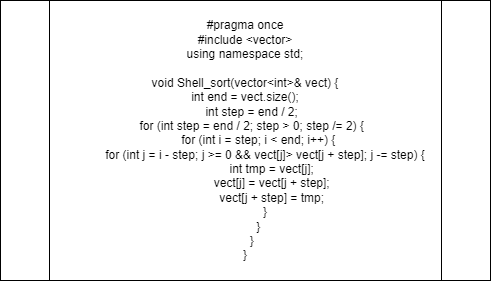
Разработать сортировку “Шелла” и проверить её на массиве размером 25 элементов.

1. Анализ задачи:

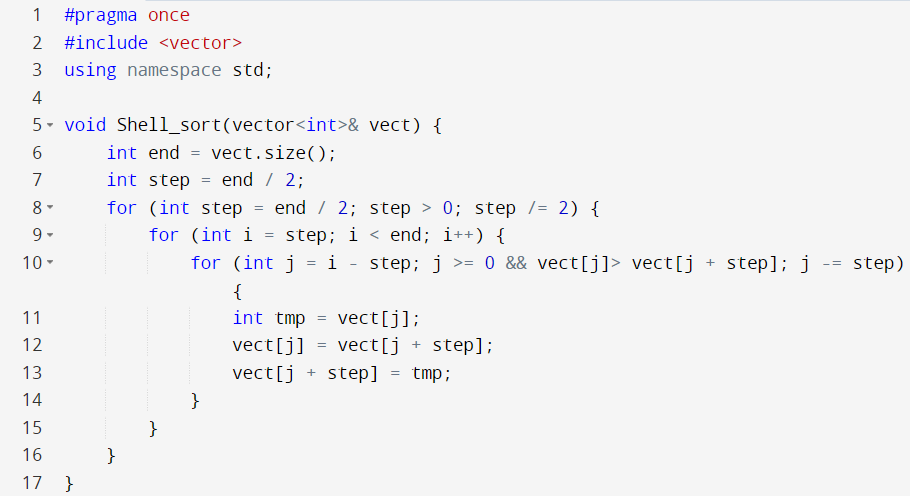
* Создаём отдельный заголовочный файл для функции
* Вычисляется шаг равный половине длины массива
* Два числа, стоящие на начале и конце шага, сортируются и шаг уменьшается на 1
* Так происходит до тех пор, пока длина шага не становится равна 1
* После прохода шага равного 2 массива становится отсортированным
* В результате получаем отсортированный массив

1. Блок-схема:

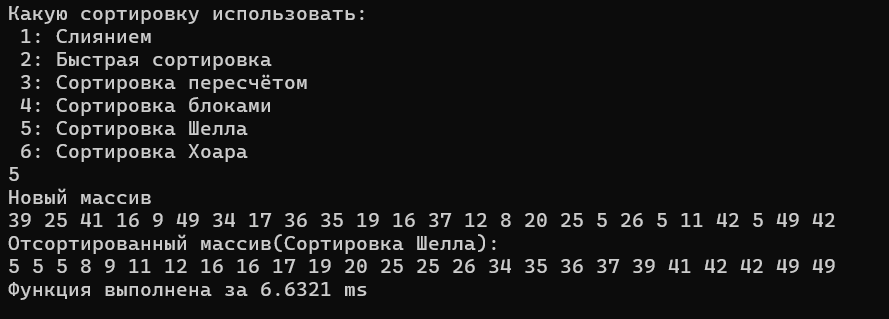
(Блок main не меняется, поэтому блок схема прикреплена выше по отчёту)



1. Код:



1. Результат работы:



1. Вывод:

Время выполнения сортировки такое же как и у быстрой сортировки, так и сортировки Хоара одинаково.

1. GitHub:

<https://github.com/Andr0medA007/Labs/tree/main/Labs_1-12>