Министерство транспорта РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта» (МИИТ)

Лабораторная работа №9

«Быстрая сортировка»

по дисциплине

«Методы программирования»

Выполнил:

студент группы ТКИ-311,

Гаврилин-Кирьянов И.А.

Проверил:

к.т.н., доц. Логинова Л. Н,

к.т.н., доц. Сафронов А. И.

Москва - 2022 г.

**Цель работы:**

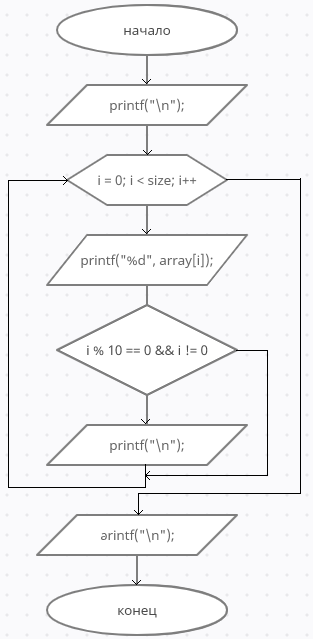
Изучение алгоритма быстрой сортировки.

**Задача:**

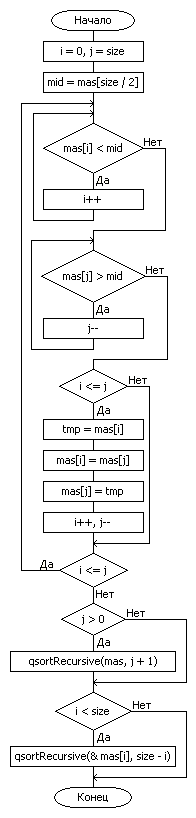
Реализовать алгоритм быстрой сортировки на языке С

**Блок-схемы:**

Функция вывода массива:



Функция быстрой сортировки:



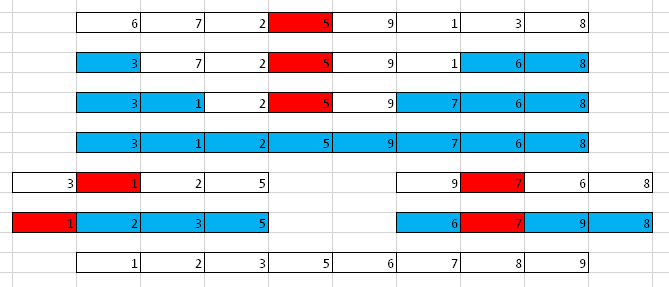
**Тестовые примеры:**

Массив длиной 8, 1000, 10000 и 100000 элементов.

Имеем массив:

6, 7, 2, 5, 9, 1, 3, 8

Проверим алгоритм вручную с помощью Excel:



Сортировка завершена.

Функция qsort рекурсивно получает центральный элемент и границы массива, а затем сортирует массив на две части, в одной из которых элементы меньше центрального, а во второй – больше. Это продолжается до тех пор, пока первый элемент меньше последнего. Если в какой-то из получившихся в результате разбиения массива частей находиться больше одного элемента, то следует произвести рекурсивное упорядочивание этой части, то есть выполнить над ней операцию разбиения, описанную выше.

**Код программы:**

#include <stdio.h>

int n;

void print(int \*array, int n);

void qsort(int \*mas, int size);

int main()

{

printf("Enter n size: ");

scanf("%d", &n);

int array[n];

for (int i = 0; i < n; i++)

{

scanf("%d", &array[i]);

}

int first = 1;

int last = n - 1;

int \*mas = array;

qsort(mas, n);

print(array, n);

}

void qsort(int \*mas, int size)

{

int i = 0, j = size;

int mid = mas[size / 2];

do

{

while(mas[i] < mid)

i++;

while(mas[j] > mid)

j--;

if (i <= j)

{

int tmp = mas[i];

mas[i] = mas[j];

mas[j] = tmp;

i++, j--;

}

} while (i <= j);

if (j > 0)

qsort(mas, j + 1);

if (i < size)

qsort(&mas[i], size - i);

}

void print(int array[n], int n)

{

printf("\n");

for (int i = 0; i <= n; i++)

{

printf("%d ", array[i]);

if (i % 10 == 0)

{

printf("\n");

}

}

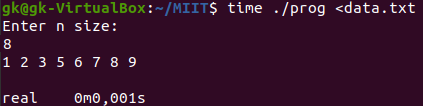
printf("\n");

return;

}

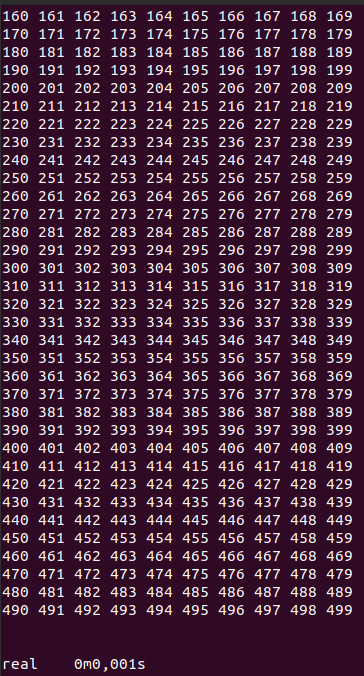
**Результаты программы:**

8 элементов:

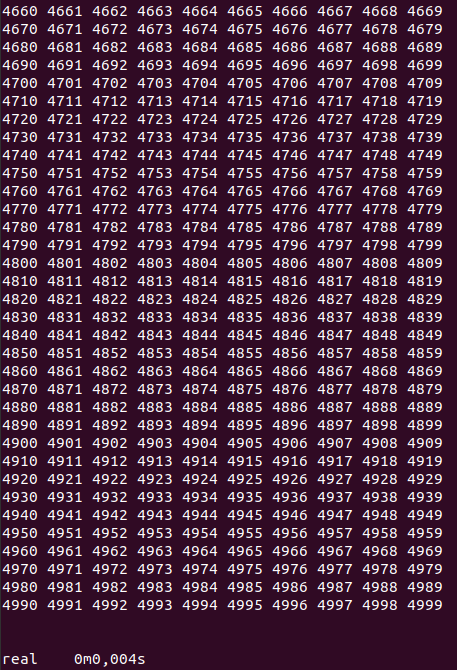
****

Программа выполняется за 0,001 секунды

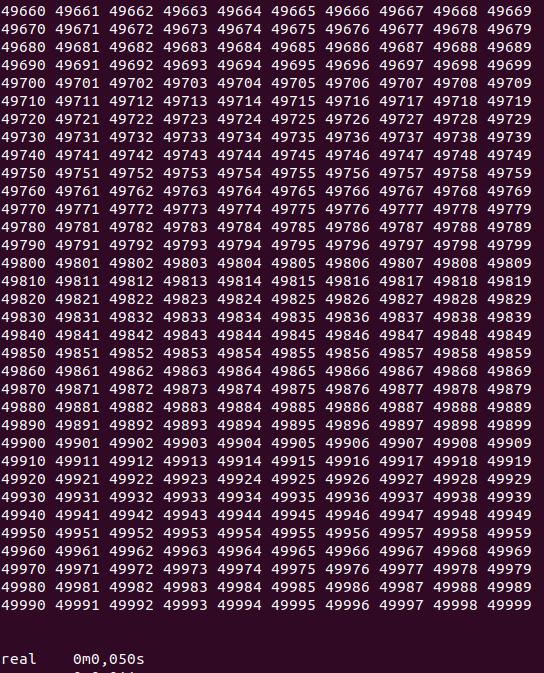
1000 элементов:



10000 элементов:



100000 элементов:



**Вывод:**

Я изучил и реализовал алгоритм быстрой сортировки. Это один из самых быстрых алгоритмов сортировки, что нам и показали тесты на разных массивах. Код является простым, алгоритм понятен для понимания. Однако, поскольку этот алгоритм рекурсивен, в случае ошибки он может выйти за границы установленной памяти, что приведет к ошибке. Так же на многих машинах есть ограничение на количество рекурсивных вызовов функций, что тоже приведет к ошибке. Но, не смотря на возможное переполнение памяти, при грамотном написании кода алгоритм требует в худшем случае O(n) дополнительной памяти, что, несомненно, является плюсом.