МИНОБРНАУКИ РОССИИ

Федеральное бюджетное образовательное учреждение высшего профессионального образования

«Ижевский государственный технический университет

имени М.Т. Калашникова»

(ФГБОУ ВПО «ИжГТУ имени М.Т. Калашникова»)

Факультет: «Программное обеспечение»

Кафедра: «Программное обеспечение»

Лабораторная работа №1

Выполнил:

Студент группы Б19-191-1з

Бирюков Филипп Денисович

Проверил:

Старший преподаватель

Еланцев М.О.

Ижевск, 2020

# Оглавление

[Оглавление 2](#_Toc63190236)

[1. ЗАДАНИЕ 3](#_Toc63190237)

[Реализовать два алгоритма сортировки целочисленного массива, согласно варианту: 3](#_Toc63190238)

[2. КОД ПРОГРАММЫ 4](#_Toc63190239)

[ArrayUtil.h: 4](#_Toc63190240)

[SelectionSort.h: 5](#_Toc63190241)

[QuickSort.h: 6](#_Toc63190242)

[Main.cpp: 6](#_Toc63190243)

[3. ПРИМЕР РАБОТЫ 8](#_Toc63190244)

# ЗАДАНИЕ

## Реализовать два алгоритма сортировки целочисленного массива, согласно варианту:

* Сортировка выбором
* Быстрая сортировка

# КОД ПРОГРАММЫ

## ArrayUtil.h:

#pragma *once*

#include <string>

#include <iostream>

#include <cstdlib>

using *namespace* std;

*void* printArray(*int\**, *int*, *char*);

*int\** randomArray(*int*);

*int* getInt();

*void* checkSorted(*int\**, *int*);

*void* swapIndexes(*int\**,*int*,*int*);

*void* swapIndexes(*int\** *array*, *int* *index1*, *int* *index2*) {

*array*[*index1*] = *array*[*index1*] + *array*[*index2*];

*array*[*index2*] = *array*[*index1*] - *array*[*index2*];

*array*[*index1*] = *array*[*index1*] - *array*[*index2*];

}

*void* printArray(*int\** *array*, *int* *size*, *char* *separator*) {

    cout << "Printing array of size " << *size* << ":" << endl;

    for (*int* i = 0; i<*size*; i++) {

        cout << "['" << i << "'=" << *array*[i] << "]" << *separator*;

    }

    cout << endl;

}

*int\** randomArray(*int* *size*) {

    cout << "Generating array of size " << *size* << endl;

*int*\* array = new *int*[*size*];

    for (*int* i = 0; i<*size*; i++) {

        array[i] = rand()%200-100;

    }

    cout << "... generated." << endl;

    return array;

}

*int* getInt() {

*int* value = 0;

    cin >> value;

    return value;

}

*void* checkSorted(*int\** *array*, *int* *size*) {

    cout << "Check is array sorted. Array size: " << *size* << endl;

*int* corruptedElement = -1;

    for (*int* i = 0; i<*size*-1; i++) {

        if (*array*[i] > *array*[i+1]) {

            corruptedElement = i;

            break;

        }

    }

    if (corruptedElement != -1) {

        cout << "Array element at index[" << corruptedElement

            <<"] bigger than element at index[" << corruptedElement+1 << "]" << endl;

        cout << "['" << corruptedElement << "'=" << *array*[corruptedElement] << "] ['"

            << corruptedElement+1 << "'=" << *array*[corruptedElement+1] << "]" << endl;

    } else {

        cout << "Array is sorted properly." << endl << endl;

    }

}

## SelectionSort.h:

#pragma *once*

#include <iostream>

#include "ArrayUtil.h"

using *namespace* std;

*void* sortBySelection(*int\**,*int*);

*void* swapIndexes(*int\**,*int*,*int*);

*int* findMin(*int\**,*int*,*int*);

*void* sortBySelection(*int\** *array*, *int* *size*) {

    cout << "Sorting by selection sort..." << endl;

    for (*int* i = 0; i<*size*; i++) {

*int* min = findMin(*array*,i,*size*);

        if (min != i) {

            swapIndexes(*array*, i, min);

        }

    }

    cout << "Sorted." << endl << endl;

}

*int* findMin(*int\** *array*, *int* *start*, *int* *size*) {

*int* min = *start*;

    for (*int* i = *start*; i<*size*; i++) {

        if (*array*[min] > *array*[i]) {

            min = i;

        }

    }

    return min;

}

## QuickSort.h:

#pragma *once*

#include <ctime>

#include <iostream>

#include "ArrayUtil.h"

using *namespace* std;

*void* quickSort(*int\**,*int*);

*void* quickSorting(*int\**,*int*,*int*);

*void* quickSort(*int\** *array*, *int* *size*) {

    cout << "Sorting by quick sort..." << endl;

*long* startSeconds = time(0);

    quickSorting(*array*, 0, *size*-1);

*long* endSeconds = time(0);

    cout << "Sorted for " << (endSeconds-startSeconds) << " seconds." << endl << endl;

}

*void* quickSorting(*int\** *array*, *int* *start*, *int* *end*) {

    if (*start* >= *end*) return;

*int* mid = (*start*+*end*)/2;

*int* midValue = *array*[mid];

*int* pointer = *start*;

*int* swaps = *start*;

    while (pointer <= *end*) {

        if (*array*[pointer] < midValue) {

            if (swaps == mid) {

                mid = pointer;

            }

*int* temp = *array*[pointer];

*array*[pointer] = *array*[swaps];

*array*[swaps] = temp;

            swaps++;

        }

        pointer++;

    }

    if (mid != swaps) {

        swapIndexes(*array*, mid, swaps);

        mid = swaps;

    }

    quickSorting(*array*, *start*, mid-1);

    quickSorting(*array*, mid+1, *end*);

}

## Main.cpp:

#include <string>

#include <iostream>

#include "SelectionSort.h"

#include "ArrayUtil.h"

#include "QuickSort.h"

#include <ctime>

using *namespace* std;

*int* main() {

    srand(time(0));

    cout << "Print array size:";

*int* size = getInt();

*int*\* array = randomArray(size);

    checkSorted(array, size);

    sortBySelection(array, size);

    printArray(array, size, ' ');

*int* quickSortArraySize = 700000;

    cout << "Create array with random values of size " << quickSortArraySize << endl;

*int*\* quickSortArray = randomArray(quickSortArraySize);

    checkSorted(quickSortArray, quickSortArraySize);

    quickSort(quickSortArray, quickSortArraySize);

    checkSorted(quickSortArray, quickSortArraySize);

    return 0;

}

# ПРИМЕР РАБОТЫ

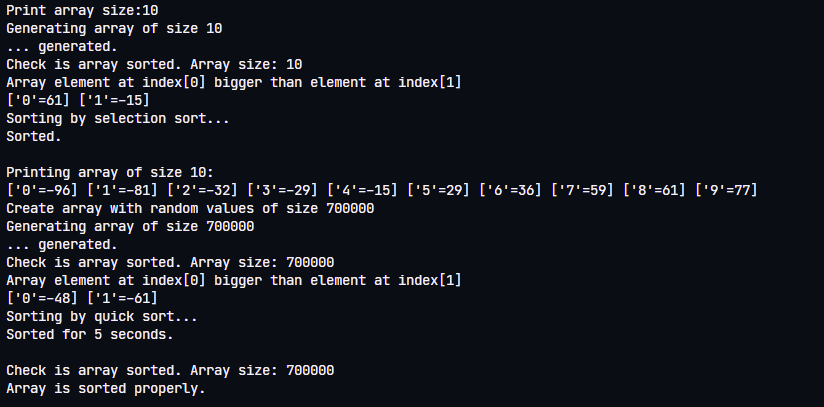


Рисунок 1