МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №4
по дисциплине «Алгоритмы и структуры данных»
Тема:Сортировки

Студент гр. 9304 Прокофьев М.Д.

Преподаватель Филатов А.Ю.

Санкт-Петербург
2020

Цель работы.

Узнать о сортировках и о их использовании в практике

Задание.

11. Быстрая сортировка, рекурсивная реализация–отсечение малых подмассивов.Быстрая сортировка выполняется не до конца. Когда сегменты становятся достаточно маленькими, они окончательно сортируются другим методом.Параметр, определяющий размер малого подмассива,задаётся пользователем

Выполнение работы.

Для выполнения работы было создано две функции: *sVstavkami()* – функция для неполной сортировки вставками и *sFast()* – функция неполной быстрой сортировки.

Функция *sFast()* работает следующим образом: изначально определяется опорный элемент *pivot*, в данном случае он всегда является серединой сортируемого массива/подмассива. После выбора опорного элемента ставятся “маяки” на левый и правый конец массива/подмассива, которые обозначаются, соответственно, за *l* и *r*. *l* итерируется по возрастанию, пока элемент находящийся в позиции *l* будет меньше опорного, *r* же итерируется по убыванию, пока элемент в ячейке *r* больше *pivot*. После итераций элементы меняются местами с помощью функции *swap().* После чего массив делится на два подмассива, после чего функция вызывается рекурсивно для подмассивов в том случае, если длина подмассива больше входной переменной min, в противном случае для этого подмассива выполняется функция *sVstavkami().*

Функция *sVstavkami()* представляет из себя неполную сортировку вставками. Работает следующим образом: происходит итерация от левого исследуемого края до правого. При итерации элемент сравнивается со всеми остальными элементами позади его до тех пор, пока не попадется элемент, который меньше чем исследуемый элемент, затем исследуемый элемент помещается в ту найденную позицию.

Переменная *min* вводится пользователем.

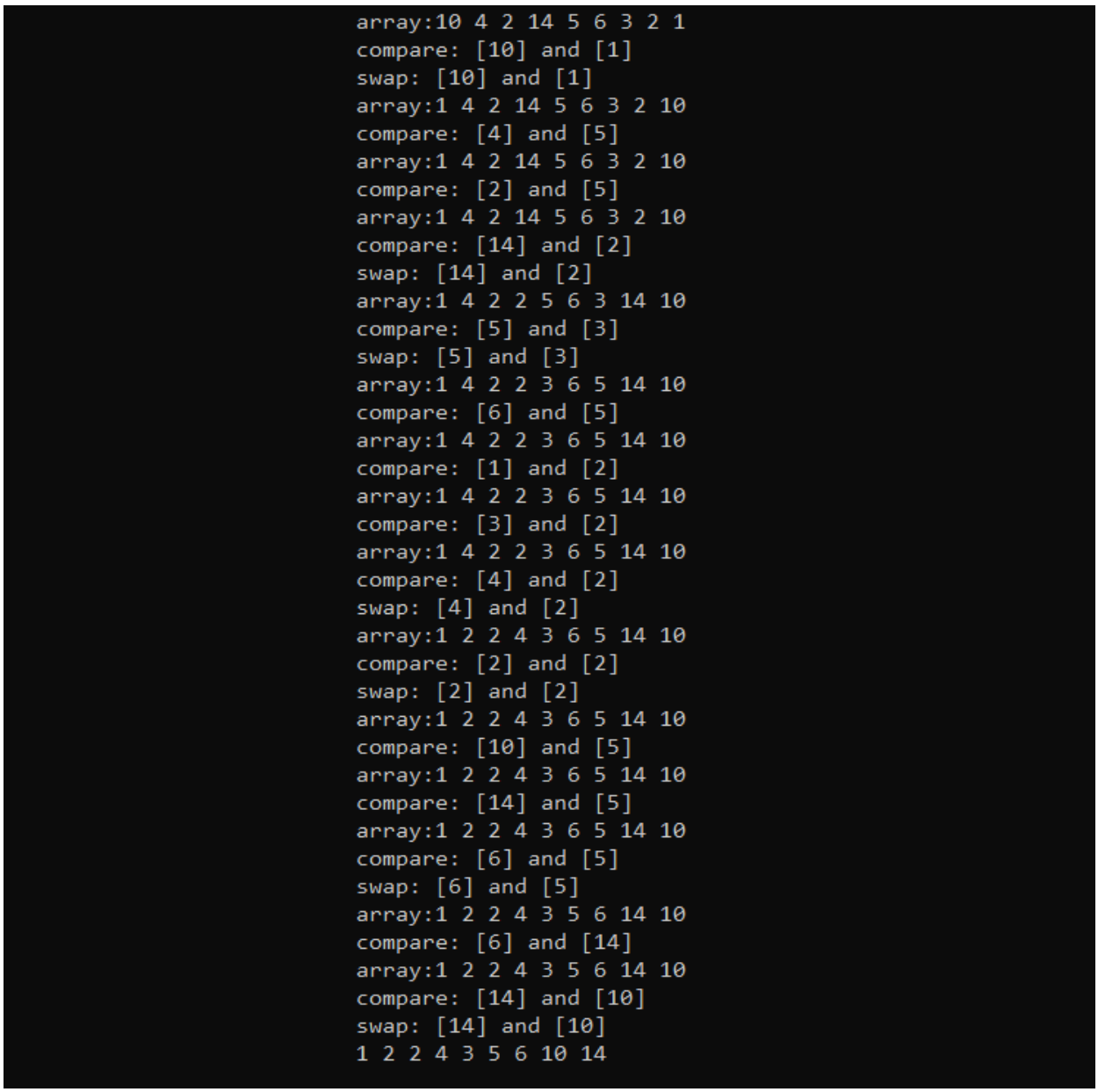

Рисунок 1 - Пример запуска программы

Тестирование.

Был написан python-скрипт для проведения тестов(testing.py). Посредством testing.py, (“входной” строке в программе) последовательно передаются строки из файлов SuccessTests.txt и ErrorTests.txt. Таким образом, при “запуске” Makefile, программа обрабатывает 2 строки из вышеуказанных файлов(SuccessTests.txt и ErrorTests.txt), после чего выдает ответы в консоли.

Результаты тестирования изложены в приложении В.

Выводы.

Написана программа быстрой сортировки с отсечением малых подмассивов. Использование иных методов сортировки при написании программы оправдано, поскольку об этом сказано в условии задания.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Файл: main.cpp

```
#include "stdafx.h"
#include <iostream>
#include <fstream>
#include "string.h"
#include <memory>


void TextToInt(std::string c, int *a_local, bool *warning)
{
	int j=0;
	int znak=1;
	int result=0;
	j=0;
	for(int i=0; i<c.length(); i++)
	{
		if((c[i] == '-')&&(znak==1)&&(result==0)) znak*=(-1);
		if((c[i] >= '0') && (c[i] <= '9'))
		{
			result=result*10+c[i]-'0';
		} else if((c[i] != '-')&&(c[i] != ' ')) *warning=1;
		if(c[i] == ' ')
		{
			result*=znak;
			a_local[j]=result;
			result=0;
			znak=1;
```

```
                    j++;
                }
            }
}



void swap_int(int *first, int *second) {
    int time_int = *first;
    *first = *second;
    *second = time_int;
}

void sVstavkami(int *a_local, int left, int right)
{
    int time_int,
        time_iter;
    for (int i = left+1; i < right; i++)
    {
        time_int = a_local[i];
        time_iter = i-1;
        while(time_iter >= left && a_local[time_iter] > time_int)
        {
            a_local[time_iter + 1] = a_local[time_iter];
            a_local[time_iter] = time_int;
            time_iter--;
        }
    }
}
```

```
void sFast(int* a_local, int left, int right, int min)
{
   int l = left, r = right;
   int v = a_local[l+(r-l)/2];
   while(l <= r)
   {
                for(;a_local[l] < v; l++);
                for(;a_local[r] > v; r--);
      if(l <= r){
         swap_int(&a_local[l], &a_local[r]);
         l++,
                        r--;
      }
   }
   if(left < r)
                if(r-left>min)
                        sFast(a_local, left, r, min);
                else
                        sVstavkami(a_local, left, r);
   if(l < right)
                if(right-l>min)
                         sFast(a_local, l, right, min);
                else
                        sVstavkami(a_local, left, r);
}

int main(int argc, char* argv[])
{
        std::string input1, input2;
   std::ifstream in("input.txt");
```

```
    getline(in, input1);
    getline(in, input2);
        int size=0;
        input1 = "1";
        input2 = "10 4 2 14 5 4 3 2 1 ";
        for(int i=0; i<input2.length(); i++)
                if(input2[i]==' ')
                        size++;
        bool warning_global=0;
        int *a = NULL;
        a = (int*)malloc(size * sizeof(int));
        TextToInt(input2, a, &warning_global);
        int min = atoi(input1.c_str());

        if((min>size)||(warning_global)) std::cout << "Incorrect!";
        else
        {
        sFast(a, 0, size-1, min);
        for(int i=0; i<size; i++) std::cout << a[i] << " ";
        }
        free(a);
        return 0;
}
```