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 _____ Мохаммед А.А.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург

2020

Цель работы.

Реализовать сортировку выбором и сортировку выбором с поиском минимума и максимума на языке C++.

Задание.

Вариант 1

1. Сортировка выбором; сортировка выбором с одновременным выбором максимума и минимум.

Формат входных и выходных данных.

На вход программе подается строка, в которой через пробелы указаны элементы сортируемого массива. Программа выводит отсортированный массив чисел.

Описание основных структур данных и функций.

- void select_sort() – функция сортирующая массив алгоритмом выбора.
- void check_string() – функция проверяющая валидность строки
- operator << - перегруженный оператор для вывода вектора

Алгоритм.

Сортировка выбором

Ищется наименьший элемент в массиве и перемещается на первое место. Затем ищется второй наименьший элемент и перемещается уже на второе место после первого наименьшего элемента. Этот процесс продолжается до тех пор, пока в массиве не закончатся не отсортированные элементы. Число проходо внешним циклом по массиву равно N-1, так как последний элемент уже будет отсортирован к моменту завершения обхода. Преимущество данного алгоритма - это простота его реализации. Недостаток - это его эффективность O(n^2).

Сортировка выбором с поиском минимума и максимума

Ищется наименьший и наибольший элементы в массиве, затем наименьший перемещается на первое место, а наибольший на последнее. После чего ищется второй наименьший и наибольший элементы в массиве и перемещаются уже на второе место после первого наименьшего элемента и на предпоследнее место соответственно. Число проходов по массиву внешним циклом равно N/2. Преимущество данного алгоритма - это простота его реализации, а его преимущество над обычной сортировкой, то что не отсортированная часть при каждой итерации внешнего цикла уменьшается сразу на два элемента, но так как количество внутренних циклов, свапов и сравнений равно двум - это лишь незначительно увеличивает скорость. Его главный недостаток - это его эффективность O(n^2).

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 - Результаты тестирования

№	Входные данные	Выходные данные
1	1	Enter string of elements: 1 This is the sorted array [1] They are identical
2	4 6 3	Enter string of elements: 4 6 3 This is 1 iteration of sorting We will swap these elements (4) and (3) [3,6,4] This is 2 iteration of sorting We will swap these elements (6) and (4) [3,4,6] This is the sorted array [3, 4, 6] They are identical
3	7 d 4 2 48 393	Enter string of elements: 7 d 4 2 48 393 This is 1 iteration of sorting We will swap these elements (7) and (2) [2,4,7,48,393] This is 2 iteration of sorting We will swap these elements (4) and (4) [2,4,7,48,393] This is 3 iteration of sorting We will swap these elements (7) and (7) [2,4,7,48,393] This is 4 iteration of sorting We will swap these elements (48) and (48) [2,4,7,48,393] This is the sorted array [2, 4, 7, 48, 393] They are identical

4	45 32 75 385 24 1	Enter string of elements: 45 32 75 385 24 1 This is 1 iteration of sorting We will swap these elements (45) and (1) [1,32,75,385,24,45] This is 2 iteration of sorting We will swap these elements (32) and (24) [1,24,75,385,32,45] This is 3 iteration of sorting We will swap these elements (75) and (32) [1,24,32,385,75,45] This is 4 iteration of sorting We will swap these elements (385) and (45) [1,24,32,45,75,385] This is 5 iteration of sorting We will swap these elements (75) and (75) [1,24,32,45,75,385] This is the sorted array [1, 24, 32, 45, 75, 385] They are identical
5	31 73 14 41 4 2 774 14 41	Enter string of elements: 31 73 14 41 4 2 774 14 41 This is 1 iteration of sorting We will swap these elements (31) and (2) [2,73,14,41,4,31,774,14,41] This is 2 iteration of sorting We will swap these elements (73) and (4) [2,4,14,41,73,31,774,14,41] This is 3 iteration of sorting We will swap these elements (14) and (14) [2,4,14,41,73,31,774,14,41] This is 4 iteration of sorting We will swap these elements (41) and (14) [2,4,14,14,73,31,774,41,41] This is 5 iteration of sorting We will swap these elements (73) and (31)

		[2,4,14,14,31,73,774,41,41] This is 6 iteration of sorting We will swap these elements (73) and (41) [2,4,14,14,31,41,774,73,41] This is 7 iteration of sorting We will swap these elements (774) and (41) [2,4,14,14,31,41,41,73,774] This is 8 iteration of sorting We will swap these elements (73) and (73) [2,4,14,14,31,41,41,73,774] This is the sorted array [2, 4, 14, 14, 31, 41, 41, 73, 774] They are identical

Выводы.

Была успешно реализована на языке C++ сортировка выбором и сортировка выбором с поиском минимума и максимума.

Исходный код программы.

```
#include <iostream>
#include <algorithm>
#include <string>
#include <sstream>
#include <fstream>
#include <vector>

using namespace std;

ostream& operator << ( ostream& out, const  vector<int>& vec) {
  out << '[';
  for (int i = 0; i < vec.size(); i++) {
    if (i == vec.size() - 1) {
      out << vec[i];
      break;
    }
    out << vec[i] << ", ";
  }
  out << ']';
  return out;
}

void select_sort( vector<int>& vec){
  int i,j,loc,temp,min;
  int n = vec.size();
  for(i=0;i<n-1;i++)
  {
    min=vec[i];
    loc=i;
    for(j=i+1;j<n;j++)
    {
      if(min>vec[j])
      {
        min=vec[j];
        loc=j;
      }
    }
      cout << "This is " << i+1 << " iteration of sorting" << std :: endl;
     cout << "We will swap these elements " << '(' << vec[i] << ')' << " and " << '('
<< vec[loc]<< ')' <<  endl;
    temp=vec[i];
    vec[i]=vec[loc];
    vec[loc]=temp;

     cout << '[';
    for(int i = 0 ; i < n ; i++)
    {
      if(i == n-1)
```

```
      {
         cout << vec[i];
      }
      else
         cout << vec[i] << ',';
    }
     cout << ']' <<  endl;
  }
}


void check_string(string& str) {
  //checks the string validity
  for (int i = 0; i < str.size(); i++) {
    if (!isdigit(str[i])&&str[i]!=' ') {
      str.erase(i, 1);
      i -= 1;
    }
  }
}


int main(int argc, char** argv) {

  cout << "____SELECTION SORT ALGORITHM___\n\n";
  cout << "Enter string of elements:\n ";
  string str1;
  vector<int> vec;
  vector<int> vecToCheck;
  int value;

  if (argc < 2)
    getline( cin, str1);
  else
    str1 = argv[1];

  check_string(str1);
  stringstream ss(str1);
  while (ss >> value)
  {
    vec.push_back(value);

    if (ss.peek() == ' '){
      ss.ignore();
    }
  }
  vecToCheck = vec;
  select_sort(vec);
   cout << "This is the sorted array" << '\n';
   cout << vec;
   cout << '\n';
   sort(vecToCheck.begin(), vecToCheck.end());
  if (vec == vecToCheck) {
```

```
      cout << "They are identical" <<  endl;
   }
   else {
      cout << "Incorrect!" <<  endl;
   }
   return 0;
}
```