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

ОТЧЕТ

по лабораторной работе №4

по дисциплине «АиСД»

Тема: Методы сортировок

Студент гр. 9303 _____ Емельянов С.А.

Преподаватель _____ Филатов А.Ю.

Санкт-Петербург

2020

Цель работы.

Создание программы сортировки данных методом Бинго.

Задание.

Вариант 6

Написать программу для сортировки данных методом Бинго.

Основные теоретические положения.

Интересной особенностью сортировки выбором является независимость скорости от характера сортируемых данных.

Например, если массив почти отсортирован, то, как известно, сортировка вставками его обработает гораздо быстрее (даже быстрее чем быстрая сортировка). А реверсно упорядоченный массив для сортировки вставками является вырожденным случаем, она будет его сортировать максимально долго.

А для сортировки выбором частичная или реверсная упорядоченность массива роли не играет — она обработает его примерно с той же скоростью что и обычный рандом. Также для классической сортировки выбором неважно, состоит ли массив из уникальных или повторяющихся элементов — на скорость это практически не влияет.

Но в принципе, можно исхитриться и модифицировать алгоритм так, чтобы при некоторых наборах данных работало быстрее. Например, бинго-сортировка учитывает, если массив состоит из повторяющихся элементов.

Здесь фокус в том, что в неупорядоченной части запоминается не только максимальный элемент, но и определяется максимум для следующей итерации. Это позволяет при повторяющихся максимумах не искать их заново каждый раз, а ставить на своё место сразу как только этот максимум в очередной раз встретили в массиве.

Алгоритмическая сложность осталась та же. Но если массив состоит из повторяющихся чисел, то бинго-сортировка справится в десятки раз быстрее, чем обычная сортировка выбором.

Выполнение работы.

Для решения поставленной задачи была разработана программа в среде Visual Studio Code на языке C++.

void bingo(T* data, int max, ofstream &fout) – основная функция. На вход принимает массив сортируемых данных, их количество в массиве и файл, в который запишется результат обработки.

Для выполнения дополнительных требований, тип сортируемых данных был сделан шаблонным (template<typename T>)

Исходный код программы представлен в приложении А. Результаты тестирования приведены в файле Result.txt , который прилагается к отчёту.

Выводы.

Были изучены теоретические материалы по методам сортировки данных. Был реализован метод сортировки Бинго, тип сортируемых данных был сделан шаблонным.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <fstream>
#include <unistd.h>
using namespace std;

template<typename T>

void bingo( T* data, int max, ofstream &fout)
{
   T nextValue = data[max];
   int max_arr = max;
   for(int i = max - 1; i != -1; i--){
      if(data[i]>data[max]) nextValue = data[max];
   }

   while (max && (data[max] == nextValue) && max--);

   while (max)
   {
      T value = nextValue;
      nextValue = data[max];
      for (int i = max - 1; i != -1; i--)
         {
            if (data[i] == value)
            {
```

```
                std::swap(data[i], data[max]);
                max--;
                for (int j = 0; j < max_arr; j++)
                {
                    fout << data[j] << " ";
                }
                fout<<" value: "<<value<<"\n";

            }
            else if (data[i] > nextValue)
                nextValue = data[i];
        }
    while (max && (data[max] == nextValue) && max--);
  }
  //return data;
}

int main()
{
  ifstream fin;
  ofstream fout;
  fin.open("test.txt");
  fout.open("Result.txt");
  int n, m;
  fin >> n;
  for (int i = 0; i < n; i++)
  {
    fin >> m;
    int *arr = new int[m+1];
```

```
        for (int j = 0; j < m; j++)
            {
            fin >> arr[j];
        }
        fout << "\n\nВходные данные [ ";
        for (int j = 0; j < m; j++)
        {
            fout << arr[j] << " ";
        }
        fout << "]\n";

        bingo(arr, m,fout);
        fout << "Результат: [ ";
        for (int j = 0; j < m; j++)
        {
            fout << arr[j] << " ";
        }
        fout << "]\n";
        // delete res;
        delete arr;
    }
    fin.close();
    fout.close();


    return 0;

}
```