**Министерство Образования, Культуры и Исследований**

**Молдавский Государственный Университет**

**Факультет Математики и Информатики**

**Департамент Информатики**

**Отчет по лабораторной работе №2**

**по предмету:**

**«Алгоритмы, Структуры Данных»**

**Выполнила:** Спыну Екатерина,

группа IA2102 (группа:дизайна)

**Проверяющий:** Mихаил Кроитор

**Кишинев, 2023**

Требуется реализовать программу на любом удобном вам языке программирования, в которой реализовано **3 способа сортировки** массивов данных из созданного файла по ключу. Для каждого метода сортировки проанализировать теоретическую и практическую сложности.

Для каждого алгоритма вывести следующую информацию:

* теоретичекая оценка сложности
* количество сравнений
* количество перестановок
* время выполнения алгоритма

## **Сортировка пузырьком (Bubble sort)**

Сортировка пузырьком – это метод сортировки массивов и списков путем последовательного сравнения и обмена соседних элементов, если предшествующий оказывается больше последующего.

В процессе выполнения данного алгоритма элементы с большими значениями оказываются в конце списка, а элементы с меньшими значениями постепенно перемещаются по направлению к началу списка. Образно говоря, тяжелые элементы падают на дно, а легкие медленно всплывают подобно пузырькам воздуха.

В сортировке методом пузырька количество итераций внешнего цикла определяется длинной списка минус единица, так как когда второй элемент становится на свое место, то первый уже однозначно минимальный и находится на своем месте.

Количество итераций внутреннего цикла зависит от номера итерации внешнего цикла, так как конец списка уже отсортирован, и выполнять проход по этим элементам смысла нет.

## **Быстрая сортировка (Quick sort)**

Быстрая сортировка, сортировка Хоара, часто называемая qsort (по имени в стандартной библиотеке языка Си) — алгоритм сортировки, разработанный английским информатикам Тони Хоаром во время своей работы в МГУ в 1960 году.

Алгоритм быстрой сортировки является рекурсивным, поэтому для простоты процедура на вход будет принимать границы участка массива от l включительно и до r не включительно. Понятно, что для того, чтобы отсортировать весь массив, в качестве параметра l надо передать 0, а в качестве r — n, где по традиции n обозначает длину массива.

В основе алгоритма быстрой сортировке лежит процедура partition. Partition выбирает некоторый элемент массива и переставляет элементы участка массива таким образом, чтобы массив разбился на 2 части: левая часть содержит элементы, которые меньше этого элемента, а правая часть содержит элементы, которые больше или равны этого элемента

## **Сортировка подсчетом (Counting sort)**

Сортировка подсчётом — это сортировочный алгоритм, применяющий набор чисел массива, подлежащего сортировке, для определения количества одинаковых элементов.

Идея этого алгоритма заключается в том, чтобы для каждого элемента х предварительно подсчитать, сколько элементов входной последовательности меньше х. Далее х записывается напрямую в выходной массив в соответствии с этим числом (если, скажем, 5 элементов входного массива меньше х, то в выходном массиве х должен быть записан на место номер 6). Если предположить, что в сортируемой последовательности могут присутствовать равные числа, то представленная схема потребует небольшой модификации, позволяющей избежать записи нескольких чисел на одно место.

**Программа:**

#include <iostream>

#include <fstream>

#include <string>

const int size = 49;

using namespace std;

struct base

{

char name[20];

char description[25];

double price;

int count;

char unit[5];

long long int id;

};

void bubbleSort(struct base b[], int length)

{

base temp;

int i;

int index;

for (i = 1; i < length; i++)

{

for (index = 0; index < length - i; index++)

if (b[index].unit > b[index + 1].unit)

{

temp = b[index];

b[index] = b[index + 1];

b[index + 1] = temp;

}

}

}

void qs(struct base b[], int left, int right)

{

register int i, j;

char\* x;

struct base temp;

i = left; j = right;

x = b[(left + right) / 2].name;

do {

while ((strcmp(b[i].name, x) < 0) && (i < right)) i++;

while ((strcmp(b[j].name, x) > 0) && (j > left)) j--;

if (i <= j) {

temp = b[i];

b[i] = b[j];

b[j] = temp;

i++; j--;

}

} while (i <= j);

if (left < j) qs(b, left, j);

if (i < right) qs(b, i, right);

}

void quickSort(struct base items[], int count)

{

qs(items, 0, count - 1);

}

void countingSort(struct base b[], int sz) {

int i, j, k;

int index;

int idx = 0;

int min, max;

min = max =0;

for (i = 1; i < sz; i++) {

min = (b[i].count < min) ? b[i].count : min;

max = (b[i].count > max) ? b[i].count : max;

}

k = max - min + 1;

int\* B = new int[k];

for (i = 0; i < k; i++) B[i] = 0;

for (index = 0; i < sz; i++) B[b[i].count - min]++;

for (index = min; i <= max; i++)

for (j = 0; j < B[i - min]; j++) b[idx++].count = i;

delete[] B;

}

int main()

{

base b[50];

ifstream fin("mebeli1.txt");

int i = 0;

if (fin)

{

for (i = 0; i < 50; i++){

fin >> b[i].name >> b[i].description >> b[i].price >> b[i].count >> b[i].unit >> b[i].id;

}

fin.close();

}

else

cout << "Error!" << endl;

void bubbleSort(struct base b[], int size);

void quickSort(struct base b[], int size);

void countingSort(struct base b[], int size);

system("pause");

return 0;

cout <<"count of if, perestanovok: " <<ifc<<endl;

cout <<"Count of comparisons(sravnenii): " << iter <<endl;

cout <<"Count of itterations: " << iter <<endl;

cout <<"Count of if: " <<ifc <<endl;

}