Отчет по лабораторной работе

по дисциплине «Структура и алгоритмы и обработки данных»

на тему:

«Методы сортировки»

Выполнил:

Студент группы БСТ1902

Бадмаев А.Д

Вариант №1

Москва 2021

Оглавление

Задания на лабораторную работу

1. Вывести в консоль «Hello World!»

2. Написать генератор случайных матриц(многомерных), который принимает опциональные параметры m, n, min_limit, max_limit, где m и n указывают размер матрицы, а min_lim и max_lim - минимальное и максимальное значение для генерируемого числа. По умолчанию при отсутствии параметров принимать следующие значения:

3. Реализовать методы сортировки строк числовой матрицы в соответствии с заданием. Оценить время работы каждого алгоритма сортировки и сравнить его со временем стандартной функции сортировки. Испытания проводить на сгенерированных матрицах.

4. Создать публичный репозиторий на GitHub.

Ход работы

Задание 1

Выведем в консоль сообщение:

```
console.log("Hello, World!");
```

Результат команды:

```
[nodemon] starting `node .\Lab1_BST1902_Badmaev.js`
Hello,World!
```

Задание 2

Реализуем функцию генерации матрицы:

```
function matrixArray(m, n, min_limit, max_limit) {
    var arr = new Array();
    for (var i = 0; i < m; i++) {
        arr[i] = new Array();
        for (var j = 0; j < n; j++) {
            arr[i][j] = Math.floor(Math.random() * (max_limit -
 min_limit + 1)) + min_limit;
        }
    }
    return arr;
```

Вызов функции матрицы:

```
const matrix = matrixArray(1000, 1000, -250, 1001)
```

Результат вывода:

```
  532,   94,  187,  12,  688,  979, 760,  259,  793, -175, -245,
  100,  819,  344, 705,  201,  -74, 479, -174,  916,  347,  150,
  692,   48,  720, 428, -235,  958, 388,  120,  787,  772,  429,
  597,  598,   48, 758,   46,  322, 750, -118,  539,  890,  105,
  417,  986,  463, 769,  653,  404, 214,  958,  274,  899, -169,
  792, -108, -192, 990,   -7,  810, 873,  297,  489,  854,   -5,
  672,  845,  262, 211,   63, -215, 240,  584, -115,  871,  590,
  -69,  367,  797, 183, -125, -191, 358,  834,  731,  450,  402,
  169,
  ... 900 more items
],
[
  956,  659,  398,  539,  15,  785,  886,  -10, 258, 607,  373,
   66,  778,  348,  191, 291,   67,  504, -117, 965, -47,  577,
  109,   57,   79,  628, 111,  174,  936,  -23, -76, 497, -118,
  104,  567, -190, -230,  88,  939,  660,  266, 525, 195,  180,
  521,  -77,  293,  341, 958,  168, -153,  780,  -8, 954,  307,
  465, -203,   66,  167, 433,  566,  461,  618, 470, 343,  434,
```

Задание 3

Сортировка выбором

```
function SelectionSort(arr) {
    var n = arr.length;
    for (var i = 0; i < n - 1; i++) {
        var min = i;
        for (var j = i + 1; j < n; j++) { if (arr[j] < arr[min]) min = j; }
        var t = arr[min];
        arr[min] = arr[i];
        arr[i] = t;
    }
    return arr;
}
```

Сортировка вставками

```
const InsertionSort = arr => {
    for (let i = 1, l = arr.length; i < l; i++) {
        const current = arr[i];
        let j = i;
        while (j > 0 && arr[j - 1] > current) {
            arr[j] = arr[j - 1];
            j--;
        }
        arr[j] = current;
    }
    return arr;
}
```

Сортировка обменом

```
function BubbleSort(arr) {
    var n = arr.length;
    for (var i = 0; i < n - 1; i++) {
        var swap = false
        for (var j = 0; j < n - 1 - i; j++) {

            if (arr[j + 1] < arr[j]) {
                var t = arr[j + 1];
                arr[j + 1] = arr[j];
                arr[j] = t;
                swap = true
            }
        }
        if (swap == false) {
            return arr
        }
    }
    return arr;
}
```

Сортировка Шелла

```
const ShellSort = arr => {
    const l = arr.length;
    let gap = Math.floor(l / 2);
    while (gap >= 1) {
        for (let i = gap; i < l; i++) {
            const current = arr[i];
            let j = i;
            while (j > 0 && arr[j - gap] > current) {
                arr[j] = arr[j - gap];
                j -= gap;
            }
            arr[j] = current;
        }
        gap = Math.floor(gap / 2);
    }
    return arr;
}
```

Быстрая сортировка

```
function QuickSort(arr, left, right) {
    var index;
    if (arr.length > 1) {
        left = typeof left != "number" ? 0 : left;
        right = typeof right != "number" ? arr.length - 1 : right;
        index = partition(arr, left, right);
        if (left < index - 1) {
            QuickSort(arr, left, index - 1);
        }
        if (index < right) {
            QuickSort(arr, index, right);
        }
    }
    return arr;
}
```

```
    return quickSort(less).concat(pivot, quickSort(more));
}
```

Пирамидальная сортировка

```
function HeapSort(arr) {
    if (arr.length == 0) return [];
    var n = arr.length, i = Math.floor(n / 2), j, k, t;
    while (true) {
        if (i > 0) t = arr[--i];
        else {
            n--;
            if (n == 0) return arr;
            t = arr[n]; arr[n] = arr[0];
        }
        j = i; k = j * 2 + 1;
        while (k < n) {
            if (k + 1 < n && arr[k + 1] > arr[k]) k++;
            if (arr[k] > t) { arr[j] = arr[k]; j = k; k = j * 2 + 1; }
            else break;
        }
        arr[j] = t;
    }
}
```

Встроенная сортировка

```
function IncludeSort(arr) {
    return arr.sort((a, b) => a - b)
}
```

Турнирная сортировка

```
class Cell {
    constructor(id, value) {
        this.id = id;
        this.value = value;
    }
}

const getWinnerIndex = (arr) => {
    const cellArr = []
    for (let index = 0; index < arr.length; index += 2) {
    if (index + 1 >= arr.length) {
        cellArr.push(new Cell(index, arr[index]))
    } else if (arr[index] < arr[index + 1]) {
        cellArr.push(new Cell(index, arr[index]))
    } else {
        cellArr.push(new Cell(index + 1, arr[index + 1]))
    }
    }
    return tournament(cellArr);
}

function tournament(arr) {
    const cellArr = [];
    for (let index = 0; index < arr.length; index += 2) {
    if (index + 1 >= arr.length) {
        cellArr.push(arr[index])
    } else if (arr[index].value < arr[index + 1].value) {
        cellArr.push(arr[index])
    } else {
        cellArr.push(arr[index + 1])
    }
    }
    if (cellArr.length > 2) {
        return tournament(cellArr)
    } else if(cellArr.length === 1) {
        return cellArr[0].id;
    } else {
        return cellArr[0].value < cellArr[1].value ? cellArr[0].id : cellArr[1].id
    }
}

function TournamentSort(arr) {
    const sortedArr = []
    while (arr.length) {
        let idx = getWinnerIndex(arr)
        sortedArr.push(arr.splice(idx, 1)[0]);
    }
    return sortedArr;
}
```

Результат выполнения сортировок