

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**"МИРЭА – Российский технологический университет"**

# РТУ МИРЭА

Институт информационных технологий (ИТ)

Кафедра МОСИТ

**ОТЧЕТ**

**ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

Исследование сортировок 2

**«АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ»**

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил студент группы ИКБО-16-19 . | |  | Павлов С.П. |
| Принял  д.т.н. , проф |  |  | Кораблин Ю.П. |
| Лабораторные работы выполнены | «\_\_» \_\_\_\_\_\_\_ 2020 г. |  |
|  |  |  |  |
| «Зачтено» | «\_\_» \_\_\_\_\_\_\_ 2020 г.  Москва 2019 |  |  |
|

**CОДЕРЖАНИЕ**

|  |  |  |
| --- | --- | --- |
| 1. | Программа сортировки реализованная на с++ | 3 |
| 2. | Результаты выполнения программы | 4 |
|  |
|  |  |  |
|  |  |  |
|  |  |  |

Программа сортировок на с++

//Shell sort

vector<int> Sorts::shell() {

this->start\_clock = clock();

vector<int> \_temp = this->input;

for (size\_t gap = \_temp.size() / 2; gap > 0; gap /= 2)

{

for (int i = gap; i < \_temp.size(); i += 1)

{

int temp = \_temp[i];

size\_t j;

for (j = i; j >= gap && \_temp[j - gap] > temp; j -= gap)

\_temp[j] = \_temp[j - gap];

\_temp[j] = temp;

}

}

this->estimated\_time = (clock() - start\_clock) / 1000.0;

return \_temp;

}

//Shaker sort

vector<int> Sorts::shaker() {

this->start\_clock = clock();

vector<int> \_temp = this->input;

int left = 0, right = \_temp.size() - 1;

bool flag = 1;

while ((left < right) && flag != 0)

{

flag = 0;

for (size\_t i = left; i < right; ++i)

{

if (\_temp[i] > \_temp[i + 1])

{

swap(\_temp[i], \_temp[i + 1]);

flag = 1;

}

}

right--;

for (size\_t i = right; i > left; i--)

{

if (\_temp[i - 1] > \_temp[i])

{

swap(\_temp[i], \_temp[i - 1]);

flag = 1;

}

}

left++;

}

this->estimated\_time = (clock() - start\_clock) / 1000.0;

return \_temp;

}

//Merge sort

vector<int> Sorts::merge() {

this->start\_clock = clock();

vector<int> \_temp = this->input;

this->merge\_sort(\_temp, 0, \_temp.size() - 1);

this->estimated\_time = (clock() - start\_clock) / 1000.0;

return \_temp;

}

void Sorts::merge\_sort(vector<int>& mas, int l, int r) {

if (l < r)

{

int m = l + (r - l) / 2;

merge\_sort(mas, l, m);

merge\_sort(mas, m + 1, r);

merge\_support(mas, l, m, r);

}

}

void Sorts::merge\_support(vector<int>& mas, int l, int m, int r)

{

int i, j, k;

int n1 = m - l + 1;

int n2 = r - m;

vector<int> L(n1);

vector<int> R(n2);

for (i = 0; i < n1; i++)

L[i] = mas[l + i];

for (j = 0; j < n2; j++)

R[j] = mas[m + 1 + j];

i = 0;

j = 0;

k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

mas[k] = L[i];

i++;

}

else

{

mas[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

mas[k] = L[i];

i++;

k++;

}

while (j < n2)

{

mas[k] = R[j];

j++;

k++;}}

Результаты выполнения программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sort | 1000 | 10000 | 100000 | 1000000 |
| Shell | 0.0004s | 0.001s | 0.014s | 0.15s |
| Shaker | 0.001s | 0.132s | 13.118s | 1420.72s |
| Merge | 0.001s | 0.003s | 0.03s | 0.262s |