FILE ORGANIZATION

MUHAMMET TALHA ODABAŞI
1306220012

KÜTÜPHANELER

```
#include <iostream>
#include <fstream>
#include <chrono>

using namespace std::chrono;
```

Yandaki görülen kütüphaneleri ekledim.

- fstream dosya işlemleri için
- iostream printlemek için
- chrono ise zaman hesaplaması yapmak için

SIRALAMA ALGORITMALARI

Sıralama algoritmalarının uygulamalarını C++ ile yaptım ve ekran görüntülerini aldım. Fonksiyon isimleri ilgili sıralama algoritmasını belirtmektedir:

```
void selectionSort(int arr[], int n)
{
    int minindex;
    for (int i = 0; i < n; i++)
    {
        minindex = i;
        for (int j = i + 1; j < n; j++)
        {
            if (arr[j] < arr[minindex])
                minindex = j;
        }
        if (minindex != i)
        {
            int temp = arr[i];
            arr[i] = arr[minindex];
            arr[minindex] = temp;
        }
    }
}
```

```
void insertionSort(int arr[], int n)
{
    for (int i = 1; i < n; i++)
    {
        int j = i;
        int key = arr[i];
        while (key < arr[--j] && j >= 0)
            arr[j + 1] = arr[j];
        arr[j + 1] = key;
    }
}
```

```
void merge(int left[], int right[], int mai[], int size)
{
    int l = 0;
    int r = 0;
    int i = 0;
    int lmax = size / 2;
    int rmax = size - lmax;
    while (l < lmax && r < rmax)
    {
        if (left[l] < right[r])
            mai[i++] = left[l++];
        else
            mai[i++] = right[r++];
    }
    while (l < lmax)
        mai[i++] = left[l++];
    while (r < rmax)
        mai[i++] = right[r++];
}

void merge_sort(int arr[], int n)
{
    int mid = n / 2;
    int left[mid];
    int right[n - mid];

    if (n <= 1) return ;
        //fill arrays
    for (int i = 0; i < n; i++)
    {
        if (i < mid)
            left[i] = arr[i];
        else
            right[i - mid] = arr[i];
    }
    merge_sort(left, mid);
    merge_sort(right, n - mid);
    merge(left, right, arr, n);
}
```

```cpp
int main()
{
    std::ifstream file("numbers.txt");
    std::string line;
    microseconds duration;
    _V2::system_clock::time_point start_time, end_time;
    int arr[1000];
    int i = 0;
    int n = 1000;

    if (file.is_open()){
        while (std::getline(file, line)){
            arr[i++] = std::stoi(line);
        }
        file.close();
        std::cout << "Original Array:\n";
        print_array(arr);

        int a;
        std::cout << "[1] Selection Sort\n[2] Insertion Sort\n[3] Merge Sort\nSelect an option: ";
        std::cin >> a;


        switch(a){
            case 1:
                start_time = high_resolution_clock::now();
                selectionSort(arr, n);
                end_time = high_resolution_clock::now();
                duration = duration_cast<microseconds>(end_time - start_time);
                std::cout << "Selection Sort took " << (double)(duration.count() / 1000.0) << " microseconds\n\n";
                std::cout << "Selection Sort:\n";
                print_array(arr);
                break;
            case 2:
                start_time = high_resolution_clock::now();
                insertionSort(arr, n);
                end_time = high_resolution_clock::now();
                duration = duration_cast<microseconds>(end_time - start_time);
                std::cout << "Insertion Sort took " << (double)(duration.count() / 1000.0) << " microseconds\n\n";
                std::cout << "Insertion Sort:\n";
                print_array(arr);
                break;
            case 3:
                start_time = high_resolution_clock::now();
                merge_sort(arr, n);
                end_time = high_resolution_clock::now();
                duration = duration_cast<microseconds>(end_time - start_time);
                std::cout << "Merge Sort took " << (double)(duration.count() / 1000.0) << " microseconds\n\n";
                std::cout << "Merge Sort:\n";
                print_array(arr);
                break;
            default:
                std::cout << "Invalid option" << std::endl;
                break;
        }
    }
    else
        std::cout << "File could not be opened" << std::endl;
}
```

int main() içerisindeki kod şekilde görüldüğü gibidir. İlk olarak numbers.txt dosyasından okunan 1000 adet integer array içerisinde tutulur ve ekrana yazdırılır. Sonrasında kullanıcıdan alınan girdiye göre ilgili sıralama algoritması uygulanır ve geçen süre mikrosaniye birimi ile gösterilir.

Örnek:

```
3768, 2867, 4937, 241, 9537, 2863, 3258, 4528, 9300, 3053, 4703, 8824, 7933, 8250, 3263, 69, 8867, 6398, 7429, 5630, 7887, 4726, 4476, 7887, 4301, 1557, 3524, 5588, 8556, 4072, 7383, 5529, 5115, 4435, 3267, 8234
, 3477, 3008, 7917, 5730, 7061, 7740, 5465, 5303, 280, 7530, 8861, 9009, 7139, 6215, 4842, 796, 4348, 3839, 9487, 7101, 4331, 9753, 712, 5661, 9242, 8697, 949, 9914, 8757, 6756, 8752, 3046, 8087, 7488, 5093, 670
1, 5212, 1429, 5029, 6712, 9481, 2838, 5486, 3045, 6456, 7534, 319, 4558, 2947, 2077, 2074, 6593, 3710, 5344, 1809, 5509, 6550, 8783, 7614, 8143, 6964, 9467, 6793, 2985, 6067, 9791, 6540, 3149, 6104, 4963, 9869,
 7977, 6771, 5174, 5923, 6183, 3434, 5578, 6628, 1921, 131, 9953, 893, 719, 4577, 991, 3644, 4567, 5686, 5175, 4409, 3194, 4180, 590, 590, 2540, 4039, 3902, 1635, 3664, 150, 4746, 2806, 5630, 4686, 8521,
[1] Selection Sort
[2] Insertion Sort
[3] Merge Sort
Select an option: 1
Selection Sort took  15.624 microseconds

Selection Sort:
0, 27, 42, 69, 78, 78, 87, 109, 111, 123, 131, 150, 150, 184, 201, 241, 267, 270, 277, 277, 279, 280, 287, 304, 317, 319, 321, 344, 387, 421, 427, 434, 479, 484, 497, 510, 512, 551, 568, 573, 577, 581, 590, 590,
 609, 611, 613, 622, 622, 625, 629, 631, 649, 658, 666, 669, 675, 680, 689, 691, 692, 705, 705, 706, 712, 718, 719, 750, 787, 789, 796, 818, 836, 845, 849, 863, 867, 873, 876, 892, 893, 900, 908, 914, 919, 934,
949, 955, 959, 974, 991, 1035, 1044, 1045, 1061, 1082, 1093, 1101, 1105, 1124, 1128, 1151, 1156, 1159, 1161, 1162, 1163, 1168, 1170, 1201, 1202, 1213, 1222, 1242, 1250, 1269, 1284, 1289, 1297, 1318, 1321, 1325,
1336, 1343, 1355, 1375, 1378, 1392, 1406, 1426, 1429, 1447, 1456, 1463, 1483, 1485, 1492, 1508, 1513, 1526, 1547, 1549, 1557, 1589, 1608, 1614, 1617, 1617, 1635, 1639, 1651, 1653, 1711, 1718, 1719, 1732, 1737, 1
770, 1798, 1809, 1810, 1813, 1829, 1839, 1864, 1874, 1878, 1886, 1887, 1898, 1910, 1920, 1921, 1960, 1971, 2067, 2071, 2074, 2077, 2080, 2088, 2090, 2092, 2119, 2124, 2148, 2164, 2172, 2179, 2187, 2191, 2207, 22
22, 2234, 2249, 2251, 2281, 2292, 2315, 2342, 2344, 2357, 2358, 2367, 2387, 2389, 2392, 2394, 2426, 2427, 2427, 2441, 2471, 2477, 2481, 2503, 2511, 2516, 2540, 2547, 2578, 2622, 2627, 2628, 2629, 2634, 2658, 266
```