A2. Анализ MERGE+INSERTION SORT. MERGE+INSERTION SORT — реализация

посылка: 293156676

ссылка на репозиторий: https://github.com/matveevaolga/set3-A2

Код реализации:

```
#include <iostream>
#include <vector>
#include <algorithm>

void insertionSort(std::vector<int>& arr_to_sort, int left, int right);

void merge(std::vector<int>& arr_to_sort, int ptr_l, int mid, int ptr_r) {
    int size_l = mid - ptr_l + 1;
    int size_r = ptr_r - mid;
    std::vector<int> left_half(size_l);
    std::vector<int> right_half(size_r);
    int ind = 0, jnd = 0, curr = ptr_l;
    while (ind < size_l) {
        left_half[ind] = arr_to_sort[ptr_l + ind];
        ind++;
    }
    while (jnd < size_r) {
        right_half[jnd] = arr_to_sort[mid + 1 + jnd];
        jnd++;
    }
    ind = jnd = 0;
    while (ind < size_l && jnd < size_r) {
        if (left_half[ind] <= right_half[jnd]) {
            arr_to_sort[curr] = left_half[ind];
            ind++;
        } else {
            arr_to_sort[curr] = right_half[jnd];
            jnd++;
        }
        curr++;
    }
    while (ind < size_l) {
        arr_to_sort[curr] = left_half[ind];
        curr++;
        ind++;
    }
    while (jnd < size_r) {
        arr_to_sort[curr] = right_half[jnd];
        curr++;
        jnd++;
    }
}

void mergeSort(std::vector<int>& arr_to_sort, int left, int right) {
    int len = right - left + 1;
    if (len <= 15) {
```

```
        insertionSort(arr_to_sort, left, right);
        return;
    }
    if (left >= right) return;
    int mid = left + (right - left) / 2;
    mergeSort(arr_to_sort, left, mid);
    mergeSort(arr_to_sort, mid + 1, right);
    merge(arr_to_sort, left, mid, right);
}

void insertionSort(std::vector<int>& arr_to_sort, int left, int right) {
    int ind = left + 1;
    while (ind <= right) {
        int curVal = arr_to_sort[ind];
        int jnd = ind - 1;
        while (jnd >= left && arr_to_sort[jnd] > curVal) {
            std::swap(arr_to_sort[jnd + 1], arr_to_sort[jnd]);
            jnd--;
        }
        arr_to_sort[jnd + 1] = curVal;
        ind++;
    }
}

int main() {
    std::ios::sync_with_stdio(false);
    std::cin.tie(nullptr);
    int n;
    std::cin >> n;
    std::vector<int> arr_to_sort(n);
    int ind = 0;
    while (ind < n) {
        std::cin >> arr_to_sort[ind];
        ind++;
    }
    mergeSort(arr_to_sort, 0, n - 1);
    ind = 0;
    while (ind < n) {
        std::cout << arr_to_sort[ind] << " ";
        ind++;
    }
    std::cout << std::endl;
    return 0;
}
```