|  |
| --- |
| #include <iostream> |
|  |  |
|  | //Bubble sort |
|  |  |
|  | template <typename Т> |
|  | void bubbleSort(Т arr[], int n) |
|  | { |
|  | int i, j; |
|  | for (i = 0; i < n - 1; i++) |
|  |  |
|  | //i элементов уже отсортированы |
|  | for (j = 0; j < n - i - 1; j++) |
|  | if (arr[j] > arr[j + 1]) { |
|  | int temporaryValue = arr[j]; |
|  | arr[j] = arr[j + 1]; |
|  | arr[j + 1] = temporaryValue; |
|  | } |
|  | } |
|  |  |
|  | //Selection sort |
|  |  |
|  | template <typename Т> |
|  | void selectionSort(Т arr[], int n) |
|  | { |
|  | for (int i = 0; i < n - 1; i++) |
|  | { |
|  | // находим минимальный элемент |
|  | int indexOfMin = i; |
|  | for (int j = i + 1; j < n; j++) |
|  | if (arr[indexOfMin] > arr[j]) |
|  | indexOfMin = j; |
|  |  |
|  | // элемент в начало массива |
|  | int curElement = arr[indexOfMin]; |
|  | while (indexOfMin > i) |
|  | { |
|  | arr[indexOfMin] = arr[indexOfMin - 1]; |
|  | indexOfMin--; |
|  | } |
|  | arr[i] = curElement; |
|  | } |
|  | } |
|  |  |
|  | //Heap sort |
|  |  |
|  | template <typename Т> |
|  | void rootNode(Т arr[], int n, int i) //выделение корневого элемента |
|  | { |
|  | int largest = i; |
|  | int l = 2 \* i + 1; // left = 2\*i + 1 |
|  | int r = 2 \* i + 2; // right = 2\*i + 2 |
|  |  |
|  | // первый элемент больше чем корень |
|  | if (l < n && arr[l] > arr[largest]) |
|  | largest = l; |
|  |  |
|  | // правый элемент больше чем максимум |
|  | if (r < n && arr[r] > arr[largest]) |
|  | largest = r; |
|  |  |
|  | // большый элемент не является корнем |
|  | if (largest != i) |
|  | { |
|  | std::swap(arr[i], arr[largest]); |
|  |  |
|  | rootNode(arr, n, largest); //рекурсивное уменьшение размера кучи на 1 элемент |
|  | } |
|  | } |
|  |  |
|  | //основная функция для сортировки кучей |
|  | void heapSort(int arr[], int n) |
|  | { |
|  | for (int i = n / 2 - 1; i >= 0; i--) |
|  | rootNode(arr, n, i); |
|  |  |
|  | //удаляем элемент из кучи |
|  | for (int i = n - 1; i >= 0; i--) |
|  | { |
|  | std::swap(arr[0], arr[i]); |
|  |  |
|  | rootNode(arr, i, 0); //рекурсивно вызываем функцию |
|  | } |
|  | } |
|  |  |
|  |  |
|  | int main() |
|  | { |
|  | int arr[] = { 17, 766, 634, 49, 9, 91, 11, 12, 25, 561 }; |
|  | int lengthArray = sizeof(arr) / sizeof(arr[0]); |
|  |  |
|  | bubbleSort(arr,lengthArray); |
|  | //selectionSort(arr,lengthArray); |
|  | //heapSort(arr, lengthArray); |
|  |  |
|  | std::cout << "Sorted array:"<< std::endl; |
|  |  |
|  | for (int i = 0; i < lengthArray; ++i) //выводим массив |
|  | std::cout << arr[i] << " "; |
|  | std::cout << std::endl; |
|  | } |