|  |  |
| --- | --- |
| **SortKSortedArray in C++** | |
| #include <iostream>  #include <vector>  #include <queue>  using namespace std;  class KthLargest {  public:  static int kthLargest(int n, vector<int>& input, int k) {  // Use a priority queue (max heap) to find the kth largest element  priority\_queue<int> pq;  // Insert all elements into the max heap  for (int i = 0; i < n; i++) {  pq.push(input[i]);  }  // Remove the top k-1 elements to get the kth largest element  for (int j = 0; j < k - 1; j++) {  pq.pop();  }  // Return the kth largest element  return pq.top();  }  };  int main() {  // Example input  vector<int> arr = {2, 4, 1, 9, 6, 8};  int k = 3;  // Call the static method kthLargest from KthLargest class  int result = KthLargest::kthLargest(arr.size(), arr, k);  // Print the result  cout << "Kth largest element: " << result << endl;  return 0;  } | Input: arr = {2, 4, 1, 9, 6, 8}  k = 3 🧾 Dry Run Table:  | **Step** | **Action** | **Heap (Max-Heap structure)** | **Top Element** | | --- | --- | --- | --- | | Init | Empty |  |  | | Insert 2 | pq.push(2) | [2] | 2 | | Insert 4 | pq.push(4) | [4, 2] | 4 | | Insert 1 | pq.push(1) | [4, 2, 1] | 4 | | Insert 9 | pq.push(9) | [9, 4, 1, 2] | 9 | | Insert 6 | pq.push(6) | [9, 6, 1, 2, 4] | 9 | | Insert 8 | pq.push(8) | [9, 6, 8, 2, 4, 1] | 9 | | Pop #1 | pq.pop() | [8, 6, 1, 2, 4] | 8 | | Pop #2 | pq.pop() | [6, 4, 1, 2] | 6 |   ➡️ Final result = 6 (3rd largest) 🖨 Output: Kth largest element: 6 |
| 6 | |