**1.Non Repeating character**

class Solution {

public:

char nonRepeatingChar(string &s) {

unordered\_map<char,int> cs;

for(char ch : s)

cs[ch]++;

for (char ch: s){

if (cs[ch]==1)

return ch;

}

return '$';

}

};

**2. k largest element**

#include <iostream>

#include <vector>

#include <queue>

#include <algorithm>

using namespace std;

vector<int> kLargestElements(vector<int>& arr, int k) {

priority\_queue<int, vector<int>, greater<int>> minHeap;

for (int i = 0; i < arr.size(); i++) {

minHeap.push(arr[i]);

if (minHeap.size() > k) {

minHeap.pop();

}

}

vector<int> result;

while (!minHeap.empty()) {

result.push\_back(minHeap.top());

minHeap.pop();

}

sort(result.rbegin(), result.rend());

return result;

}

**3.Bubble sort:**

#include <iostream>

using namespace std;

void bubbleSort(int arr[], int n) {

for (int i = 0; i < n - 1; i++) {

for (int j = 0; j < n - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

int temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

void printArray(int arr[], int n) {

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

int main() {

int arr[] = {64, 34, 25, 12, 22, 11, 90};

int n = sizeof(arr) / sizeof(arr[0]);

cout << "Original array: ";

printArray(arr, n);

bubbleSort(arr, n);

cout << "Sorted array: ";

printArray(arr, n);

return 0;

}

**4.quick sort:**

#include <iostream>

using namespace std;

int partition(int arr[], int low, int high) {

int pivot = arr[high];

int i = low - 1;

for (int j = low; j < high; j++) {

if (arr[j] < pivot) {

i++;

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

int temp = arr[i + 1];

arr[i + 1] = arr[high];

arr[high] = temp;

return i + 1;

}

void quickSort(int arr[], int low, int high) {

if (low < high) {

int pi = partition(arr, low, high);

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

void printArray(int arr[], int n) {

for (int i = 0; i < n; i++) {

cout << arr[i] << " ";

}

cout << endl;

}

int main() {

int arr[] = {10, 7, 8, 9, 1, 5};

int n = sizeof(arr) / sizeof(arr[0]);

cout << "Original array: ";

printArray(arr, n);

quickSort(arr, 0, n - 1);

cout << "Sorted array: ";

printArray(arr, n);

return 0;

}