Lecture 38 Heap

1. Priority queue STL
2. #include <iostream>
3. #include <queue>
4. using namespace std;
5. int main() {
6. //priority\_queue<int> pq; // max heap
7. priority\_queue<int, vector<int>, greater<int>> pq; // min heap
8. int n;
9. cin>>n;
10. for(int i=0; i<n; i++)
11. {
12. int no;
13. cin>>no;
14. pq.push(no);
15. }
16. while(!pq.empty())
17. {
18. cout<<pq.top()<<", ";
19. pq.pop();
20. }
21. return 0;
22. }

2. [https://practice.geeksforgeeks.org/problems/minimum-cost-of-ropes-1587115620/1/?problemStatus=solved&page=1&category[]=Heap&category[]=priority-queue&query=problemStatussolvedpage1category[]Heapcategory[]priority-queue#](https://practice.geeksforgeeks.org/problems/minimum-cost-of-ropes-1587115620/1/?problemStatus=solved&page=1&category%5b%5d=Heap&category%5b%5d=priority-queue&query=problemStatussolvedpage1category%5b%5dHeapcategory%5b%5dpriority-queue#)

long long minCost(long long arr[], long long n) {

// Your code here

long long cost = 0;

priority\_queue<long, vector<long>, greater<long>> pq(arr, arr+n);

while(pq.size()>1)

{

long A = pq.top();

pq.pop();

long B = pq.top();

pq.pop();

cost += A+B;

pq.push(A+B);

}

return cost;

}

3. <https://leetcode.com/problems/kth-largest-element-in-an-array/>

class Solution {

public:

int findKthLargest(vector<int>& nums, int k) {

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

int n = nums.size();

for(int i=0; i<k; i++)

{

pq.push(nums[i]);

}

for(int i=k; i<n; i++)

{

if(nums[i] > pq.top())

{

pq.pop();

pq.push(nums[i]);

}

}

return pq.top();

}

};