**Minimum Cost of ropes**

There are given **N** ropes of different lengths, we need to connect these ropes into one rope. The cost to connect two ropes is equal to sum of their lengths. The task is to connect the ropes with minimum cost. Given **N** size array **arr[]** contains the lengths of the ropes.

**Example 1:**

**Input:**

n = 4

arr[] = {4, 3, 2, 6}

**Output:**

29

**Explanation:**

We can connect the ropes in following ways.

1) First connect ropes of lengths 2 and 3.

Which makes the array {4, 5, 6}. Cost of

this operation 2+3 = 5.

2) Now connect ropes of lengths 4 and 5.

Which makes the array {9, 6}. Cost of

this operation 4+5 = 9.

3) Finally connect the two ropes and all

ropes have connected. Cost of this

operation 9+6 =15

Total cost for connecting all ropes is 5

+ 9 + 15 = 29. This is the optimized cost

for connecting ropes.

Other ways of connecting ropes would always

have same or more cost. For example, if we

connect 4 and 6 first (we get three rope of 3,

2 and 10), then connect 10 and 3 (we get

two rope of 13 and 2). Finally we

connect 13 and 2. Total cost in this way

is 10 + 13 + 15 = 38.

**Example 2:**

**Input:**

n = 5

arr[] = {4, 2, 7, 6, 9}

**Output:**

62

**Explanation:**

First, connect ropes 4 and 2, which makes

the array {6,7,6,9}. Cost of

this operation 4+2 = 6. Next, add ropes

6 and 6, which results in {12,7,9}.

Cost of this operation 6+6 = 12.

Then, add 7 and 9, which makes the array {12,16}.

Cost of this operation 7+9 = 16. And

finally, add these two which gives {28}.

Hence, the total cost is 6 + 12 + 16 +

28 = 62.

**Expected Time Complexity** : O(nlogn)  
**Expected Auxilliary Space** : O(n)

**Constraints:**  
1 ≤ N ≤ 200000  
1 ≤ arr[i] ≤ 106

**Company Tags**

[**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) [**OYO Rooms**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=OYO%20Rooms) [**Goldman Sachs**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Goldman%20Sachs)

**Topic Tags**

[**Queue**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Queue) [**Heap**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Heap) [**Data Structures**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Data%20Structures)

Java code

import java.math.\*;

import java.util.\*;

import java.io.\*;

class Driverclass

{

// Driver Code

static class FastReader{

BufferedReader br;

StringTokenizer st;

public FastReader(){

br = new BufferedReader(new InputStreamReader(System.in));

}

String next(){

while (st == null || !st.hasMoreElements()){

try{ st = new StringTokenizer(br.readLine()); } catch (IOException e){ e.printStackTrace(); }

}

return st.nextToken();

}

String nextLine(){

String str = "";

try{ str = br.readLine(); } catch (IOException e) { e.printStackTrace(); }

return str;

}

Integer nextInt(){

return Integer.parseInt(next());

}

Long nextLong(){

return Long.parseLong(next());

}

}

public static void main(String args[])

{

FastReader sc = new FastReader();

PrintWriter out = new PrintWriter(System.out);

int t = sc.nextInt();

while(t>0)

{

int n = sc.nextInt();

long arr[] = new long[n];

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

arr[i] = sc.nextLong();

out.println(new Solution().minCost(arr, n));

t--;

}

out.flush();

}

}

class Solution

{

long minCost(long arr[], int n)

{

PriorityQueue<Long> pq=new PriorityQueue<>();

for(long num : arr)pq.add(num);

long ans=0;

while(pq.size()!=1){

long num=pq.poll()+pq.poll();

ans+=num;

pq.add(num);

}

return ans;

}

}