Problem: Carrying Stones

In ancient Egypt, slaves were given the difficult task of carrying stones used to build the pyramids. In particular, stones would be laid in piles, from left to right and one slave would have to pick up the first pile, carry it to the second pile, add the stones from that pile to the current pile and carry all of it to the third pile and so forth, until he arrived at the last pile. Each pile was separated by one unit of distance. We define a stone unit of work as being the amount of work necessary to carry one stone one unit of distance. For example, if the piles from left to right were:

```
3          6          2          5
```

then the total number of stone units of work would be 3 x 3 + 6 x 2 + 2 x 1 = 23, since we carry the leftmost stones 3 units of distance, the second set of stones two units of distance and the third set of stones one unit of distance.

As the slave owner, you have some sort of compassion for your slave. Thus, given the sizes of each pile of stones, you would like to lay them out in such a way that minimizes the amount of work (in stone units) your slave has to carry out.

The Problem

Given the number of stones in each pile, determine the minimum amount of work our slave must do in stone units, to move all the stones into one pile in the manner described.

The Input

The first line starts with a single integer, *n*, representing the number of input cases. Each case follows. On the first line of each input case will be a single positive integer, *m (m ≤ 1000000)*, representing the number of piles of stones. This is followed by *m* integers, separated by white space (either spaces or newlines), each in between 1 and 32768, representing the number of stones in the given pile.

The Output

For each input case, output a single integer, representing the minimum amount of work the slave must do in stone units to move all the stones into one pile at the right end as described above, assuming the slave owner arranges the piles in the optimal manner.

Sample Input

```
2
4
3 6 2 5
10
5 2 3 4 2
1 1 1 1 1
```

Sample Output

```
17
59
```

Implementation Requirements

You must implement either Merge Sort of Quick Sort in your solution. Note: Other methods to solve this problem exist, but the purpose of this assignment is to practice one of these two sorting methods.

Hints

The final answer to some of the queries will not fit in an int. Thus, it's best to carry out all calculations using the type "long long".