Solved Problem - Subarray Sum

In this lesson, we'll see an effective way to compute sum of an subarray.

We'll cover the following

Problem statement

Given an array, A, of N integers, answer Q queries of the type (l, r) - sum of the elements in the subarray $A[l...r]$. Print the sum for each query.

Input format

The first line contains two space-separated integers N and Q $(1 \leq N, Q \leq 10^6)$.

The second line contains N integers representing the array $A[]$ $(1 \leq A[i] \leq 10^6)$.

The next Q lines each contain a pair of integers l and r.

Output format

Print Q integers and answer to the queries.

Sample

Input

```
5 3
1 2 4 8 16
1 5
2 3
3 5
```

Output

```
31 6 28
```

Brute force

The brute force method would be to loop over the subarray in the query and sum all the elements.

I am skipping the code for this solution because it is trivial.

The time for each query would be $O(N)$, the total complexity of the solution would be $O(Q * N)$. This means it is not good enough for the given constraints.

Optimization

First, let’s discuss what the prefix sum array is.

The prefix sum array $sum[]$ of an array $A[]$ is defined as

$sum[i] = \sum_{k=1}^{i} A[k]$

or, ith element of $sum[]$ is sum of first i elements of $A[]$

We can use the prefix sum array to find the sum of a subarray in $O(1)$ time.

$sum[i...j] = A[i] + A[i+1] + ... + A[j]$

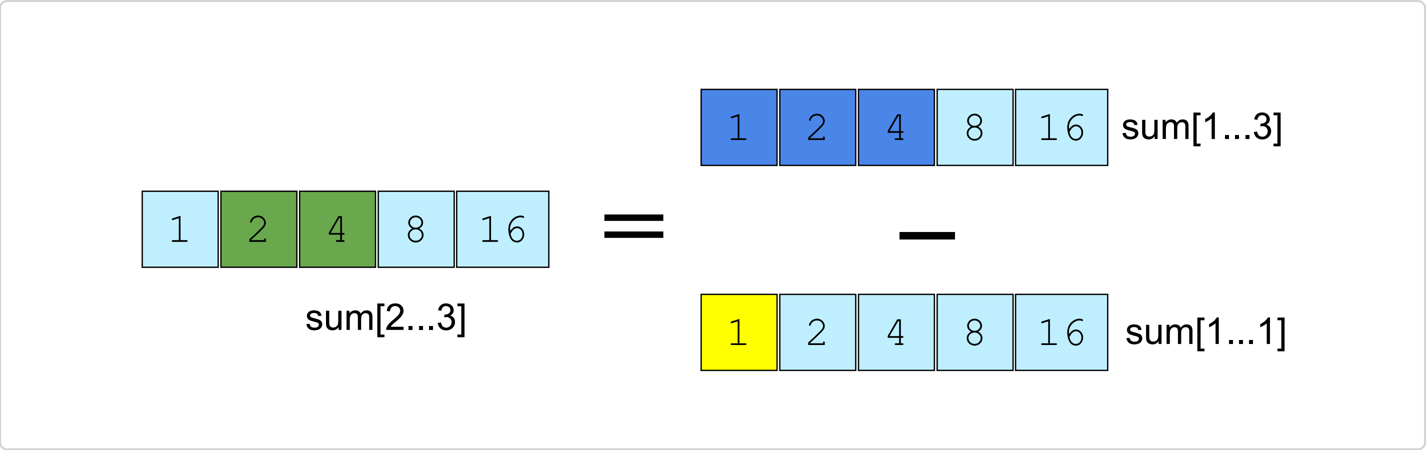
$= (A[1] + A[2] + ... + A[j]) - (A[1] + A[2] + ... + A[i-1])$

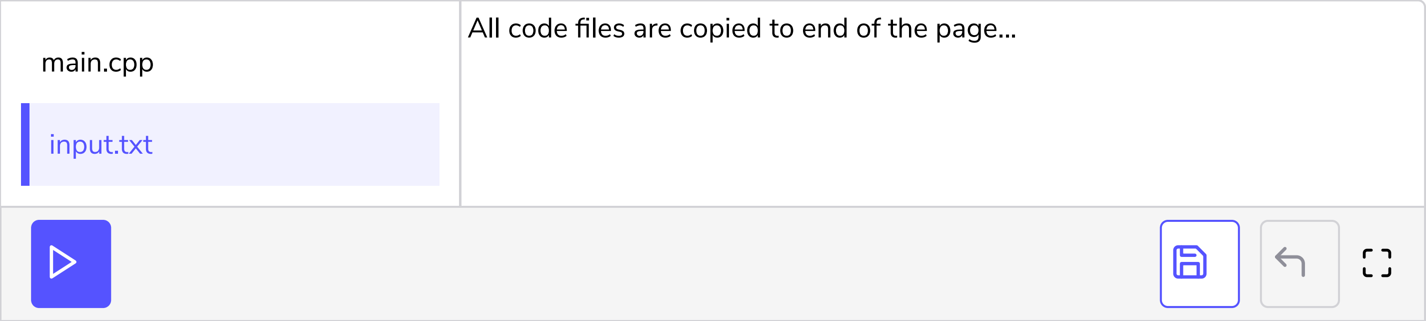
$= sum[j] - sum[i-1]$

From preprocessing to computing the $sum[]$ array takes $O(N)$ time. Each query is just $O(1)$.

So the total time complexity is $O(N + Q)$.

See the below illustration for a better understanding.





All code files are copied to end of the page...

In the next chapter, we'll start with Sieve of Eratosthenes, an efficient algorithm to generate primes.

Code Files Content !!!

⟱ ⟱
⟱

```
-------------------------------------------------------------------------
|  main.cpp [1]
-------------------------------------------------------------------------



#include
#include
#include
using namespace std;

int main() {
  ifstream cin("input.txt");

  int N, Q;
  cin >> N >> Q;
  vector A(N);
  for (int i = 0; i < N; i++) cin>>A[i];
  for (int i = 0; i < N; i++) cin>>A[i];

  vector sum(N);
  sum[0] = A[0];
  for(int i = 1; i < N; i++)
    sum[i] = sum[i - 1] + A[i];

  for(int i = 0; i < Q; i++) {
    int l, r;
    cin >> l >> r;
    l--; r--; // Convert to 0-based indexing
    int ans = sum[r];
    if (l > 0)
      ans -= sum[l - 1];

    cout << ans << " ";
  }

  return 0;
}


-------------------------------------------------------------------------
|  input.txt [1]
-------------------------------------------------------------------------



5 3
1 2 4 8 16
1 5
2 3
3 5





**********************************************************************************
```