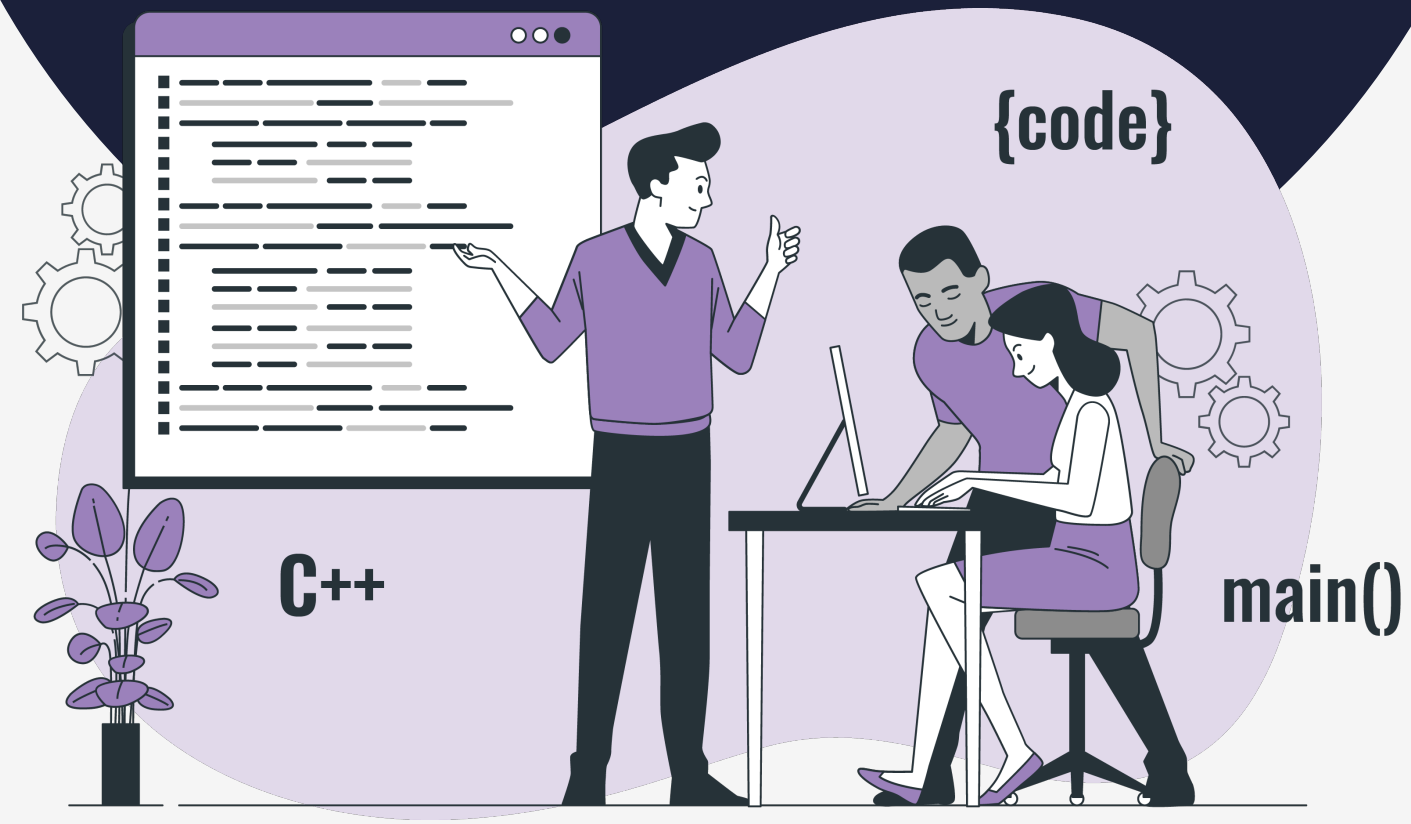
Lesson:

Problems on Array - 3



PW SKILLS

Pre-Requisites

- Arrays
- Vectors

List of Concepts Involved

- Array Problems based on prefix sums concept.

Pattern: Prefix Sums

In this approach, we create an array by taking the sum of values from the beginning of the array and keep adding them. Prefix[i] represents the values from the beginning of the array till index 'i'. Using this prefix sums technique, we can calculate range sums easily which will be seen in the problems discussed below.

Problem 1: Given an integer array 'a', return the prefix sum/ running sum in the same array without creating a new array.

Input :
5 4 1 2 3

Output :
5 9 10 12 15

Code:

```
vector<int> runningSum(vector<int> a) {
      for (int i = 1; i < a.size(); ++i){
          a[i] += a[i - 1];
  }
      return a;
  }
```

Problem 2: Check if we can partition an input array into two subarrays with equal sum. More formally, check that the prefix sum of a part of the array is equal to the suffix sum of the rest of the array.

Input :
5 2 3 4

Output :
True

Explanation of approach:
We can calculate the total sum of the whole array in the first traversal. In the second traversal, check at every point that the sum of the prefix part of the array is equal to the suffix part of the array. Calculate the suffix using total sum - current prefix.

Code:

```
bool check(vector<int> &a) {
  int n = a.size();
  int pref = 0, total_sum = 0;
  for (int i = 0; i < n; i++) {
    total_sum += a[i];
  }

  for (int i = 0; i < n; i++) {
    pref += a[i];
    int suff = total_sum - pref;
    if (pref == suff) return true;
  }

  return false;
}
```

Problem 3: Given an array of integers 'a' of size n. For q number of inputs, print the sum of values in a given range of indices from l(starting index for the range) to r(ending index for the range),both l and r included in the sum.
More formally, print a[l] + a[l+1] + a[l+2] + ... + a[r] for each q.

Note: Array 'a' follows **1-based indexing** i.e. element 1 is at index 1 and not 0, as it usually is.

Input :

```
5     //Number of elements in the array a
5 1 2 3 4   //All the elements (space separated)
4    //value of q
1 1    //l and r for the 1st q, q=1
1 3    //l and r for the 2nd q, q=2
4 5   //l and r for the 3rd q, q=3
1 5   // l and r for the 4th q, q=4
```

Output :

```
5   //sum from a[1] to a[1], i.e. 5 itself
8   //sum from a[1] to a[3], i.e. 5+1+2 = 8
7   //sum from a[4] to a[5], i.e. 4+3=7
15   //sum from a[1] to a[5], i.e. 5+1+2+3+4=15
```

Code:

```
#include<iostream>
using namespace std;
int main(){
int n;
cin >> n;
vector<int> a(n + 1);
for (int i = 1; i ⩽ n; i++) { // taking one based indexing for easier calculations
  cin >> a[i];
}
for (int i = 1; i ⩽ n; i++) {
  a[i] += a[i - 1]; // making a prefix sum array out of given array
}
int q; // no of queries
cin >> q;
while (q--) {
  int l, r;
  cin >> l >> r;
  // we need to find sum of values of indices from l to r (both included)
  // so that is equal to (total sum till r - total sum till l-1 )
  // note we also need to include the value at index l so subtracting only till (l-1)
  cout << a[r] - a[l - 1] << endl;
 }
}
```

That is all for this lesson! Do not miss the next one !!

Upcoming class teaser :

- 2-D array problems