Arithmetic Slices in C++

```
#include <iostream>
#include <vector>
using namespace std;

int solution(const vector<int>& arr) {
    vector<int> dp(arr.size(), 0);
    //vector<int> dp;
    int ans = 0;
    for (size_t i = 2; i < arr.size(); i++) {
        if (arr[i] - arr[i - 1] == arr[i - 1] - arr[i - 2]) {
            dp[i] = dp[i - 1] + 1;
            ans += dp[i];
        }
    }
    return ans;
}

int main() {
    vector<int> arr = {2, 5, 9, 12, 15, 18, 22, 26, 30, 34,
36, 38, 40, 41};
    cout << solution(arr) << endl;
    return 0;
}
```

Given Input

vector<int> arr = {2, 5, 9, 12, 15, 18, 22, 26, 30, 34, 36, 38, 40, 41};

- **Size of array**: n = 14

Step-by-Step Dry Run

We'll track how dp[i] and ans evolve.

Initialization

Index (i)	arr[i]	dp[i]	ans (Sum of dp[i])
0	2	-	-
1	5	-	-

Loop Execution (i = 2 to i = 13)

i	arr[i]	Check Condition arr[i] - arr[i-1] == arr[i-1] - arr[i-2]	dp[i] Calculation	ans Update
2	9	(9 - 5) == (5 - 2) → 4 == 3 ✗	dp[2] = 0	ans = 0
3	12	(12 - 9) == (9 - 5) → 3 == 4 ✗	dp[3] = 0	ans = 0
4	15	(15 - 12) == (12 - 9) → 3 == 3 ✓	dp[4] = dp[3] + 1 = 1	ans = 1
5	18	(18 - 15) == (15 - 12) → 3 == 3 ✓	dp[5] = dp[4] + 1 = 2	ans = 3
6	22	(22 - 18) == (18 - 15) → 4 == 3 ✗	dp[6] = 0	ans = 3
7	26	(26 - 22) == (22 - 18) → 4 == 4 ✓	dp[7] = dp[6] + 1 = 1	ans = 4
8	30	(30 - 26) == (26 - 22) → 4 == 4 ✓	dp[8] = dp[7] + 1 = 2	ans = 6
9	34	(34 - 30) == (30 - 26) → 4 == 4 ✓	dp[9] = dp[8] + 1 = 3	ans = 9
10	36	(36 - 34) == (34 - 30) → 2 == 4 ✗	dp[10] = 0	ans = 9
11	38	(38 - 36) == (36 - 34) → 2 == 2 ✓	dp[11] = dp[10] + 1 = 1	ans = 10
12	40	(40 - 38) == (38 - 36) → 2 == 2 ✓	dp[12] = dp[11] + 1 = 2	ans = 12
13	41	(41 - 40) == (40 - 38) → 1 == 2 ✗	dp[13] = 0	ans = 12

Final dp Table

Index (i)	arr[i]	dp[i]	ans (Sum of dp[i])
0	2	-	-
1	5	-	-
2	9	0	0
3	12	0	0
4	15	1	1
5	18	2	3
6	22	0	3
7	26	1	4
8	30	2	6
9	34	3	9
10	36	0	9
11	38	1	10
12	40	2	12
13	41	0	12

Final Output

12

Output:-
12