Count Distinct Subsequence C++

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

int countDistinctSubsequences(const string& str) {
    int n = str.length();
    int dp[n + 1];
    dp[0] = 1; // Empty subsequence

    unordered_map<char, int> lastOccurrence;

    for (int i = 1; i <= n; i++) {
        dp[i] = 2 * dp[i - 1];
        char ch = str[i - 1];
        if (lastOccurrence.find(ch) !=
lastOccurrence.end()) {
            int j = lastOccurrence[ch];
            dp[i] -= dp[j - 1];
        }
        lastOccurrence[ch] = i;
    }
    return dp[n] - 1;
}

int main() {
    string str = "abc";
    cout << countDistinctSubsequences(str) << endl;
    return 0;
}
```

Dry Run with Input "abc"

Initialization:

str = "abc";
n = 3;
dp[0] = 1; // Empty subsequence
lastOccurrence = {} // Initially empty

Iteration Table

i	str[i-1]	dp[i] Calculation	dp[i] Value	lastOccurrence Update
1	'a'	dp[1]=2×dp[0]=2×1	**2**	{'a': 1}
2	'b'	dp[2]=2×dp[1]=2×2	**4**	{'a': 1, 'b': 2}
3	'c'	dp[3]=2×dp[2]=2×4	**8**	{'a': 1, 'b': 2, 'c': 3}

Final Calculation

Result=dp[n]−1=8−1=7

(The -1 removes the empty subsequence.)

Final Output

7

The distinct non-empty subsequences of "abc":

a, b, c, ab, ac, bc, abc

Output:-
7