[](https://leetcode.com/KnockCat/)

✅ C++ || Easy || 3 Approaches || Brute Force || Recursive || Memoization

[KnockCat](https://leetcode.com/KnockCat/)



9970

31016

Mar 02, 2022

C

Dynamic Programming

**392. Is Subsequence**

**KNOCKCAT**

1. Easy C++

2. Line by Line Explanation with Comments.

3. Detailed Explanation ✅

4. Brute force & dynamic Programming Apprach.

5. Please Upvote if it helps⬆️

6. Link to my Github Profile contains a repository of Leetcode with all my Solutions. ⬇️

// 😉If you Like the repository don't foget to star & fork the repository😉

[LeetCode](https://github.com/knockcat/Leetcode) **LINK TO LEETCODE REPOSITORY**

Please upvote my comment so that i get to win the 2022 giveaway and motivate to make such discussion post.  
**Happy new Year 2023 to all of you**  
**keep solving keep improving**  
Link To comment  
[Leetcode Give away comment](https://leetcode.com/discuss/general-discussion/2946993/2022-Annual-Badge-and-the-Giveaway/1734919)

**BruteForce Approach**

* Just we need to compare both string by traversing
* if t[i] == s[i] , we will increase the count.
* if cnt == s.length() this means t is the subsequence of s
* As, it contains alll the characters.

**CODE**

// 😉😉😉😉Please upvote if it helps 😉😉😉😉

class Solution {

public:

bool isSubsequence(string s, string t) {

int j = 0; // For index of str1 (or subsequence

// Traverse str2 and str1, and

// compare current character

// of str2 with first unmatched char

// of str1, if matched

// then move ahead in str1

for (int i = 0; i < t.length() && j < s.length(); i++)

if (s[j] == t[i])

j++;

// If all characters of str1 were found in str2

return (j == s.length());

}

};

**RECURSIVE IMPLEMENTATION**

* The idea is simple, we traverse both strings from one side to another side
* (say from rightmost character to leftmost).
* If we find a matching character, we move ahead in both strings.
* Otherwise, we move ahead only in str2.

**CODE**

// 😉😉😉😉Please upvote if it helps 😉😉😉😉

class Solution {

public:

bool isSubs(string& s, string& t, int m, int n)

{

if(m == 0)

return true;

if(n == 0)

return false;

// If last characters of two

// strings are matching

if (s[m - 1] == t[n - 1])

return isSubs(s, t, m - 1, n - 1);

// If last characters are

// not matching

return isSubs(s, t, m, n - 1);

}

bool isSubsequence(string s, string t) {

if( isSubs(s,t,s.length(),t.length()))

return true;

return false;

}

};

**MEMOIZATION TECHNIQUE**

* Here the idea is to check whether the size of the longest common subsequence is equal to the size of str1.
* If it’s equal it means there is a subsequence that exists in str2.

**CODE**

// 😉😉😉😉Please upvote if it helps 😉😉😉😉

class Solution {

public:

// returns the length of longest common subsequence

int isSubs(string& s1, string& s2, int i , int j,vector<vector<int>> &t)

{

if(i == 0 || j == 0)

return 0;

if(t[i][j] != -1)

return t[i][j];

if(s1[i-1] == s2[j-1])

return t[i][j] = 1 + isSubs(s1,s2,i-1,j-1,t);

else

return t[i][j] = isSubs(s1,s2,i,j-1,t);

}

bool isSubsequence(string s1, string s2) {

int m = s1.length();

int n = s2.length();

// intialising dp matrix with -1

if(m > n)

return false;

vector<vector<int>> t(m+1,vector<int> (n+1,-1));

if(isSubs(s1,s2,m,n,t) == m)

return true;

return false;

}

};

``