712. Minimum ASCII Delete Sum for Two Strings

DescriptionHintsSubmissionsDiscussSolution

Given two strings `s1, s2`, find the lowest ASCII sum of deleted characters to make two strings equal.

Example 1:

```
Input: s1 = "sea", s2 = "eat"
Output: 231
Explanation: Deleting "s" from "sea" adds the ASCII value of "s" (115) to the
sum.
Deleting "t" from "eat" adds 116 to the sum.
At the end, both strings are equal, and 115 + 116 = 231 is the minimum sum
possible to achieve this.
```

Example 2:

```
Input: s1 = "delete", s2 = "leet"
Output: 403
Explanation: Deleting "dee" from "delete" to turn the string into "let",
adds 100[d]+101[e]+101[e] to the sum.  Deleting "e" from "leet" adds 101[e] to
the sum.
At the end, both strings are equal to "let", and the answer is 100+101+101+101 =
403.
If instead we turned both strings into "lee" or "eet", we would get answers of
433 or 417, which are higher.
```

Note:

- `0 < s1.length, s2.length <= 1000`.
- All elements of each string will have an ASCII value in `[97, 122]`.

Seen this question in a real interview before?

- Difficulty:Medium
- Total Accepted:778
- Total Submissions:1.9K
- Contributor: m_deepakraja
-
- Subscribe to see which companies asked this question.

  Related Topics Dynamic Programming

  Similar Questions

  Edit Distance Delete Operation for Two Strings

This is clearly a DP problem.

```
dp[i][j] is the cost for s1.substr(0,i) and s2.substr(0, j). Note s1[i], s2[j]
not included in the substring.

Base case: dp[0][0] = 0
target: dp[m][n]

if s1[i-1] = s2[j-1]   // no deletion
    dp[i][j] = dp[i-1][j-1];
else   // delete either s1[i-1] or s2[j-1]
    dp[i][j] = min(dp[i-1][j]+s1[i-1], dp[i][j-1]+s2[j-1]);
```

```cpp
#include<iostream>
#include<stdio.h>
#include<vector>
#include<math.h>
#include<unordered_map>
#include<limits.h>
using namespace std;
int minimumDeleteSum(string s1, string s2) {
    int m=s1.size();
    int n=s2.size();
    vector<vector<int>>dp(m+1,vector<int>(n+1,0));
    for(int j=1;j<=n;j++)
          dp[0][j]=dp[0][j-1]+s2[j-1];
     for(int i=1;i<=m;i++)
     {
          dp[i][0]=dp[i-1][0]+s1[i-1];
          for(int j=1;j<=n;j++)
          {
               if(s1[i-1]==s2[j-1])
                    dp[i][j]=dp[i-1][j-1];
               else{
                    dp[i][j]=min(dp[i-1][j]+s1[i-1],dp[i][j-
1]+s2[j-1]);
               }
          }
     }
     return dp[m][n];
}
```