712. Minimum ASCII Delete Sum for Two Strings

Given two strings `s1` and `s2`, return *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.
```

Constraints:

- `1 <= s1.length, s2.length <= 1000`
- `s1` and `s2` consist of lowercase English letters.

```
def minimumDeleteSum(self, s1: str, s2: str) -> int:
        dp = [[0]*(len(s2)+1) for _ in range(len(s1)+1)]
        n = len(s1)
        m = len(s2)
        for i in range(1,n+1):
            dp[i][0] = dp[i-1][0]+ord(s1[i-1])
        for j in range(1,m+1):
            dp[0][j] = dp[0][j-1]+ord(s2[j-1])
        for i in range(1,n+1):
            for j in range(1,m+1):
                if s1[i-1]==s2[j-1]:
                    dp[i][j] = dp[i-1][j-1]
                else:
                    dp[i][j] = min(ord(s1[i-1])+dp[i-1][j], ord(s2[j-
1])+dp[i][j-1])
        return dp[-1][-1]
```