Exercício Proposto 4

Disciplina: Projeto e Análise de Algoritmos
Aluno: Diego Santos Seabra
Matrícula: 0040251

Baseado no que foi discutido do problema de Subsequência comum mais longa, implemente em uma linguagem de programação de sua preferência a solução para o respectivo problema utilizando programação dinâmica. Teste a solução implementada com programação Dinâmica com a sua respectiva solução recursiva.

Abaixo, as duas sequências de entrada para os testes.

bbhbigkcgbdehgejggaajibgkbekkjjcblghehfeeclehcaghbbblfebkggfbifcjdbfcfgjdllkfejcddljfecj
hldccchebkjlkgbkiigjcejiafalgijaigkhaifjabeedcgjbbkeagbjfejlfehgfijjggkifddikccjbkkhglkecfhbe

Tamanho: 399

In [82]:
```
# Importações
import numpy as np

# Definição do Algoritmo
def lcs_dinamico(A, B, imprimeMatriz):
    # Armazena o tamanho das sequências A e B nas variáveis m e n, respect
    m = len(A)
    n = len(B)

    # Memoização - Declarando a matriz
    # (tabela cujas entradas são calculadas em ordem de linha principal)
    mat = [[0] * (n+1) for i in range(m+1)]

    # Percorre a matriz
    for i in range(m+1):
        for j in range(n+1):
            # Aplica as regras da programação dinâmica
            if i == 0 or j == 0: # "Correção" dos índices
                mat[i][j] = 0
            elif A[i-1] == B[j-1]:
                mat[i][j] = 1 + mat[i-1][j-1]
            else:
                mat[i][j] = max(mat[i-1][j], mat[i][j-1])

    # Imprime a matriz
    if imprimeMatriz:
        print(np.array(mat))

    # Retorna a posição da matriz que contém o comprimento máximo
    # Diferentemente do pseudocódigo, aqui é m e n já que m e n são, respe
    #  o tamanho de A e B
    return mat[m][n]
```

In [90]:
```
# Teste 1 - Slides(bd/abcd)
seq1 = "bd"
seq2 = "abcd"

lcs_dinamico(seq1, seq2, True)
```

```
[[0 0 0 0 0]
 [0 0 1 1 1]
 [0 0 1 1 2]]
```

Out[90]: 2

In [91]:
```
# Teste 2 - Slides(stone/longest)
seq1 = "stone"
seq2 = "longest"

lcs_dinamico(seq1, seq2, True)
```

```
[[0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 1]
 [0 0 0 0 0 0 1 2]
 [0 0 1 1 1 1 1 2]
 [0 0 1 2 2 2 2 2]
 [0 0 1 2 2 3 3 3]]
```

Out[91]: 3

In [93]:
```
# Teste 3 - Bioinformática(AGGTAB/GXTXAYB)
seq1 = "AGGTAB"
seq2 = "GXTXAYB"

lcs_dinamico(seq1, seq2, True)
```

```
[[0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 1 1]
 [0 1 1 1 1 1 1 1]
 [0 1 1 1 1 1 1 1]
 [0 1 1 2 2 2 2 2]
 [0 1 1 2 2 3 3 3]
 [0 1 1 2 2 3 3 4]]
```

Out[93]: 4

In [94]:
```
# Teste 4 - Bioinformática(TAGTCACG/AGACTGTC)
seq1 = "TAGTCACG"
seq2 = "AGACTGTC"

lcs_dinamico(seq1, seq2, True)
```

```
[[0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 1 1 1]
 [0 1 1 1 1 1 1 1 1]
 [0 1 2 2 2 2 2 2 2]
 [0 1 2 2 2 3 3 3 3]
 [0 1 2 2 3 3 3 3 4]
 [0 1 2 3 3 3 3 3 4]
 [0 1 2 3 4 4 4 4 4]
 [0 1 2 3 4 4 5 5 5]]
```

Out[94]: 5

In [95]:
```
# Resolução do Exercício Proposto 4
seq1 = "hldccchebkjlkgbkiigjcejiafalgijaigkhaifjabeedcgjbbkeagbjfejlfehgfi
seq2 = "bbhbigkcgbdehgejggaajibgkbekkjjcblghehfeeclehcaghbbblfebkggfbifcjdl

lcs_dinamico(seq1, seq2, True)
```

```
[[  0   0   0 ...   0   0   0]
 [  0   0   0 ...   1   1   1]
 [  0   0   0 ...   2   2   2]
 ...
 [  0   1   2 ... 397 397 397]
 [  0   1   2 ... 398 398 398]
 [  0   1   2 ... 398 398 399]]
```

Out[95]: 399