Trabalho 4: Problema BeeCrowd #2842 via Programação Dinâmica

Nomes: Henrique Fugie de Macedo e Luiz Otávio Silva Cunha
Matrículas: 0056151 e 0056153

Problema BeeCrowd #2842

O problema 2842 no BeeCrowd se trata de um problema para encontrar a menor string que contém duas strings dadas como subsequência.
"Dabriel está brincando com suas duas maravilhosas strings e, ao fazer algumas operações com elas, percebeu uma coisa: sempre existe uma terceira string que contém como subsequência as suas outras duas strings. Uma subsequência é formada através da remoção de alguns caracteres, e os restantes se mantém na mesma posição relativa. Por exemplo: A string 'casa' contém como subsequência a string 'cs', mas não contém a string 'ac'. Após um tempo analisando essas propriedades, Dabriel percebeu que para gerar a terceira string bastava concatenar as outras duas, uma coisa muito trivial. Portanto, ele solicitou sua ajuda para determinar qual o tamanho da menor string que possui as duas como subsequência.

Exemplo 1:

 # Entrada:
casa
casaco

 # Saída:
6

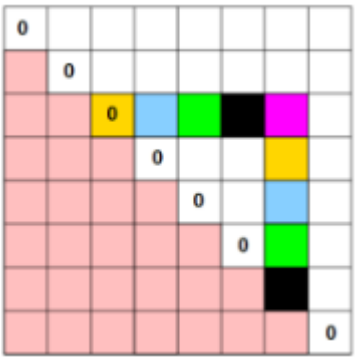
Exemplo 2:

 # Entrada:
bola
bota

 # Saída:
5

Programação Dinâmica

A Programação Dinâmica (PD) é uma técnica de otimização que visa resolver problemas complexos dividindo-os em subproblemas menores e resolvendo cada subproblema apenas uma vez, armazenando suas soluções em uma tabela para evitar cálculos redundantes.



As informações coletadas anteriormente pela programação dinâmica são guardadas em uma dp table, como mostrada na imagem anterior, e foi utilizada para solucionar este problema no código.

Explicação do código

O código começa recebendo duas strings, que se tiver espaços em branco depois, são removidos, para garantir que a entrada seja correta, em seguida ele chama um print que dentro dele, tem a função principal do programa, menor_tamanho_subsequencia, que recebe como entrada as duas strings digitadas pelo usuário. A abordagem escolhida para solucionar o problema é através da tabela dp, que armazena o resultado de soluções anteriores e através deles, continua solucionando o problema, de modo que ele monta uma matriz, colocando mais um de tamanho tanto nas linhas quanto nas colunas para garantir que os casos base, como por exemplo, na analise da primeira letra tiver uma igual ou se alguma das outras letras achar uma igual logo no primeiro caso, não aconteça nenhum erro, aqui está uma imagem de exemplo da matriz gerada através das palavras bola e bota.

```
bota
bola
Matriz inicial gerada:

0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0
```

Em seguida ele possui dois loops, aninhados, onde o primeiro percorre a primeira palavra inteira e o segundo a segunda palavra, fazendo com que a combinação entre todas as letras sejam verificadas, seguido por uma condição que verifica se as duas letras que estão sendo analisadas são iguais ou não, se sim, ele recebe o valor encontrado anteriormente resolvido pelo subproblema, considerando que os casos base, da linha e coluna zero, vai receber zero e somar mais 1, e na segunda opção, é a parte onde ele escolhe entre o resultado da célula a sua esquerda ou a célula acima dela, o programa vai escolher o maior entre eles e armazenar o valor na célula atual. Ele vai executando estes passos até chegar

no final, onde ele fez todas as combinações possíveis, o que resulta na última posição da matriz, a quantidade total de letras que se repetem, aí para encontrar o resultado final, basta somar a quantidade de letras das duas strings e subtrair as repetidas de uma das strings, resultando na menor string que contém as duas strings dadas como subsequência.

```
Encontrou que b e igual a b
Paga o valor na posicao 0x0 = 0 e soma 1, ficando 1 na posicao 1x1

0 0 0 0 0
0 1 0 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0


Comparando b da primeira palavra com o da segunda palavra

0 0 0 0 0
0 1 1 0 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0


Comparando b da primeira palavra com t da segunda palavra

0 0 0 0 0
0 1 1 1 0
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0


Comparando b da primeira palavra com a da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 0 0 0 0
0 0 0 0 0
0 0 0 0 0


Comparando o da primeira palavra com b da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 1 0 0 0
0 0 0 0 0
0 0 0 0 0


Comparando o da primeira palavra com o da segunda palavra

Encontrou que o e igual a o
Paga o valor na posicao 1x1 = 1 e soma 1, ficando 2 na posicao 2x2

0 0 0 0 0
0 1 1 1 1
0 1 2 0 0
0 0 0 0 0
0 0 0 0 0


Comparando o da primeira palavra com t da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 1 2 2 0
0 0 0 0 0
0 0 0 0 0


Comparando o da primeira palavra com a da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 1 2 2 2
0 0 0 0 0
0 0 0 0 0


Comparando l da primeira palavra com b da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 1 2 2 2
0 1 0 0 0
0 0 0 0 0


Comparando l da primeira palavra com o da segunda palavra

0 0 0 0 0
0 1 1 1 1
0 1 2 2 2
0 1 2 0 0
0 0 0 0 0
```

Estas imagens mostram o que é feito na matriz em ambos os casos, quando a letra analisada é igual e quando a letra analisada não é igual, que ele escolhe qual o maior resultado de subproblema encontrado anteriormente e armazena.