Casamento perfeito

Pilhas e filas são geralmente consideradas estruturas de dados básicas e podem ser empregadas em sistemas operacionais, simulações discretas de eventos, dentre outros. As pilhas são muito importantes também na teoria da computação / linguagens formais. Neste exercício, você deverá implementar um algoritmo automatizado para identificar se os parênteses, colchetes e chaves estão sendo utilizados corretamente em um texto.

Dado um texto de entrada, você deverá escrever um programa em python 3 que verificará se há um casamento perfeito entre os parênteses, colchetes e chaves em uma string. Um casamento perfeito ocorre quando os delimitadores da string (parênteses, colchetes e chaves) à esquerda possuem os correspondentes à direita na ordem de ocorrência inversa, tal como utilizado nas equações matemáticas.

Exemplos de strings **com casamento perfeito dos delimitadores**:

- bsi [ufrpe { recife (dois) mil }aaaa]kkk
- huuuumm{}asa[ssd]fox{}(asas{ss})
- sddssdds
- (( ) [( )])
- [ ] ( ) [ ]
- (([ ]) [( )]) [( )]

Exemplos de strings **sem casamento perfeito dos delimitadores**:

- essa(nao[ta muito) legal] ne
- tem{coisa{a{mais}aqui}ne}kkkk}
- ([ )]
- ([ ])]
- (( ) [ ]

Entrada
A única linha da entrada terá a string a ser verificada. Você deverá usar a função input() do python 3.

Saída
A saída deverá imprimir uma das duas opções a seguir: “casamento perfeito” ou “casamento imperfeito”. Não use aspas. Não imprima espaços extras antes ou depois do texto.

REGRA GERAL: você deverá usar uma pilha para armazenar os delimitadores que forem abertos e ela deve ser a única estrutura de armazenamento de dados utilizada no algoritmo. Não é permitido usar bibliotecas. É permitido o uso de uma lista de python e somente as funções da lista: len, append e pop.

Exemplos:

Exemplo 1 – entrada:
bsi [ufrpe { recife (dois) mil }aaaa]kkk
Exemplo 1 - saída:
casamento perfeito

Exemplo 2 – entrada:
essa(nao[ta muito) legal] ne
Exemplo 2 - saída:
casamento imperfeito