Respostas de Recursividade - Versão Informal

1. **O que é recursividade?**
Recursividade é quando uma função chama ela mesma. Pra funcionar, precisa de duas coisas:
- Um caso base (pra parar a função)
- Chamada recursiva (a função se chamando)

2. **Função exemplo (`misterio`)**
- O `if n == 0` é o caso base, que encerra tudo.
- Cada vez que a função é chamada, ela soma `n` com o resultado da próxima chamada.
- Saída:
  - n=1 -> 1
  - n=2 -> 3
  - n=3 -> 6
- Resultado final: 6

3. **Função `contar`**
Ela imprime de `n` até 1 e depois mostra “Fim!”.
- Exemplo: `contar(4)` ->
4
3
2
1
Fim!

4. **Quando usar recursão ou laço**
- Recursão: melhor pra problemas que se dividem em partes menores, tipo fatorial ou Fibonacci.
- Laços: melhor pra repetições simples, tipo somar números ou percorrer listas, e é mais rápido.

5. **Pilha de execução**
- Ela guarda as funções que estão rodando.
- Cada chamada recursiva entra na pilha.
- Se não tiver caso base, dá erro `StackOverflow`.

6. **Soma de números pares até n**
```
def soma_pares(n):
    if n == 0:
        return 0
    if n % 2 == 0:
        return n + soma_pares(n - 1)
    return soma_pares(n - 1)

soma_pares(6) # resultado: 12
```

7. **Maior elemento da lista**
```
def maior(lista):
    if len(lista) == 1:
        return lista[0]
    m = maior(lista[1:])
    return lista[0] if lista[0] > m else m

maior([3,7,2,9,5]) # resultado: 9
```

8. **Contar ocorrências de x em lista**
```
def contar_ocorrencias(lista, x):
    if lista == []:
        return 0
    cont = 1 if lista[0] == x else 0
    return cont + contar_ocorrencias(lista[1:], x)

contar_ocorrencias([2,4,2,6,2],2) # resultado: 3
```

9. **Inverter texto**
```
def inverter(texto):
    if texto == "":
        return ""
    return inverter(texto[1:]) + texto[0]

inverter("recursao") # resultado: "oasrucer"
```

10. **Contar elementos da lista**
```
def contar_elementos(lista):
```

```
    if lista == []:
        return 0
    return 1 + contar_elementos(lista[1:])

contar_elementos([2,4,6,8]) # resultado: 4
```