ALGORITMOS E PROGRAMAÇÃO DE COMPUTADORES II

Recursão I

Recursão

Uma função é dita recursiva quando é definida em seus próprios termos, direta ou indiretamente.

Por exemplo, podemos modificar a função abaixo, que imprime os elementos de uma lista, para que seja recursiva:

```
def imprime(l):
    for i in range(len(l)):
        print(l[i])
```

```
def imprime_rec(l, i=0):
    if i < len(l):
        print(l[i])
        imprime_rec(l, i+1)
```

Recursão

```
def imprime_rec(l, i=0):
  if i < len(l):
    print(l[i])
    imprime_rec(l, i+1)
```

```
>>> l = [1, 2, 3]
>>> imprime_rec(l)
```

Efeitos da recursão

A cada chamada:

- Empilham-se na memória os dados locais (variáveis e parâmetros) e o endereço de retorno.
  - A funcão corrente só termina quando a função chamada terminar.
- Executa-se a nova chamada (que também pode ser recursiva)
- Ao retornar, desempilham-se os dados da memória, restaurando o estado antes da chamada recursiva

Efeitos da recursão

Mesmo resultado, mas com duplicação de código:

```
def imprime0(l, ...):
    print(l[0])
    imprime1(l, ...)
```

```
def imprime2(l, ...):
    print(l[2])
    imprime3(l, ...)
```

```
def imprime1(l, ...):
    print(l[1])
    imprime2(l, ...)
```

```
def imprime3(l, ...):
    ...
```

Quando usar?

Dependendo do problema, a recursão pode ser bem ou mal empregada.

Geralmente, se existe uma versão simples e não recursiva da função, ela deve ser usada.

Usamos recursão quando o problema pode ser definido recursivamente de forma natural.

Como usar?

Necessário definir 3 pontos:

1) **Definir o problema de forma recursiva, ou seja, em termos dele mesmo**
2) **Definir a condição de término (ou condição básica)**
3) **A cada chamada recursiva, deve-se tentar garantir que se está mais próximo de satisfazer a condição de término**

Como usar?

Exemplo: problema do fatorial

1) **Definir o problema de forma recursiva, ou seja, em termos dele mesmo**
   - **n! = n * (n – 1)!**
2) **Definir a condição de término (ou condição básica)**
   - **n = 0**
3) **A cada chamada recursiva, deve-se tentar garantir que se está mais próximo de satisfazer a condição de término**
   - **A cada chamada, n é decrementado**

Exemplo

Implementar uma função recursiva em Python para calcular o fatorial de um número inteiro positivo passado como parâmetro.

Exercício

Implementar uma função recursiva para calcular o n-ésimo termo da sequência de Fibonacci.

Considere os três pontos definidos para o problema:

1) **f(0) = 0, f(1) = 1, f(n) = f(n-1) + f(n-2) p/ n>=2**
2) **n=0 ou n=1**
3) **n deve ser decrementado a cada chamada**

ALGORITMOS E PROGRAMAÇÃO DE COMPUTADORES II

Recursão I