# Funções Recursisas

## Algoritmos e Estrutura de Dados I

**Aluno: Reinan Gabriel Dos Santos Souza**
Bacharelado em Sistemas de Informação
Instituto Federal de Sergipe
Campus Lagarto

23 de setempo de 2023

**Reinan Gabriel Dos Santos Souza**

Sou um entusiasta de tecnologia apaixonado por aprender e explorar novas áreas.

GitHub.com/reinanhs
Linkedin.com/in/reinanhs

# Sumário

# Recursividade

A recursividade é um conceito fundamental em programação, onde uma **função chama a si mesma** para resolver um problema. Ela é amplamente utilizada na solução de problemas complexos.

- Consiste em utilizar a própria função que estamos a definir na sua definição;
- Em todas as função recursivas existem:
  - Um passo básico (ou mais) cujo resultado é imediatamente conhecido.
  - Um passo recursivo em que se tenta resolver um sub-problema do problema inicial.

# Recursividade em Algoritmos

- Algoritmos recursivos são aqueles que se dividem em subproblemas menores.
- Utilizam resultados desses subproblemas para resolver o problema original.
- Exemplos incluem o **cálculo do fatorial** e a **sequência de Fibonacci**.

# Vantagens da Recursividade

- A recursão pode tornar o código mais claro e fácil de entender.
- Recomendado para resolver problemas que possuem uma estrutura recursiva natural.
- Uma tarefa complexa pode ser dividida em sub-tarefas mais simples usando recursão.
- Proporciona soluções elegantes.

# Desvantagens da Recursividade

- Pode consumir muita memória em casos de profundidade excessiva.
- Pode ser menos eficiente do que abordagens iterativas para alguns problemas.
- Funções recursivas são difíceis de depurar.

# Exemplo de Código Recursivo

Implementação da sequência de Fibonacci, ilustrando como a recursão funciona na prática:

```java
public class FibonacciRecursivo {
    public static int fibonacci(int n) {
        if (n <= 1) {
            return n; // Caso base: Fibonacci(0) = 0 e Fibonacci(1) = 1
        } else {
            // Chamada recursiva para Fibonacci(n-1) e Fibonacci(n-2)
            return fibonacci(n - 1) + fibonacci(n - 2);
        }
    }

    public static void main(String[] args) {
        int n = 10; // Altere o valor de 'n' para obter diferentes números na sequência de Fibonacci
        System.out.println("O " + n + "º número na sequência de Fibonacci é: " + fibonacci(n));
    }
}
```

# Exemplo de expressões matemáticas

Veja como utilizar uma expressão matemática usando esse template:

$$
\begin{aligned}
S(\omega) &= \frac{\alpha g^2}{\omega^5} e^{[-0.74\left\{ \frac{\omega U_\omega 19.5}{g} \right\}^{-4}]} \\
&= \frac{\alpha g^2}{\omega^5} \exp\left[-0.74\left\{ \frac{\omega U_\omega 19.5}{g} \right\}^{-4}\right]
\end{aligned}
$$

# Conclusão

A recursão é uma ferramenta poderosa em programação que oferece soluções elegantes para muitos problemas. Compreender quando e como usá-la é fundamental para se tornar um programador mais eficiente e capaz de resolver problemas complexos de forma mais simples.

1: Recursividade: https://pt.wikipedia.org/wiki/Recursividade

# Obrigado

Gostaria de expressar meu sincero agradecimento a todos vocês pela participação nesta apresentação sobre "**Recursividade em Algoritmos e Estruturas de Dados**".

Se você tiver mais dúvidas ou quiser continuar a discussão sobre qualquer tópico relacionado à programação, algoritmos ou estruturas de dados, estou à disposição para ajudar.