

INF319 — Projeto e Implementação Orientados a Objetos

Progressões: Recursão, Iteração

Luiz E. Busato

Instituto de Computação – UNICAMP
buzato@ic.unicamp.br

Especialização em Engenharia de Software

Especificação

Progressão Matemática

- Uma progressão (seqüência) é uma lista ordenada de inteiros não negativos, denominados **termos** da progressão.
- A determinação de um termo de uma progressão é feita por uma função inteira que mapeia o inteiro que representa a posição do termo na seqüência ao inteiro que corresponde ao termo.
- Freqüentemente, a função inteira que define a progressão é **recursiva** e pode usar um, dois ou um número arbitrário de termos anteriores em sua definição.
- Entretanto, uma implementação pode optar alternativamente por uma formulação **iterativa**.

Aritmética

Recursiva

$$a(i) = \begin{cases} 0 & \text{se } i = 0 \\ a(i - 1) + \text{incremento} & \text{se } i \geq 1 \end{cases}$$

Iterativa

$$a(i) = \begin{cases} 0 & \text{se } i = 0 \\ a(0) + i \times \text{incremento} & \text{se } i \geq 1 \end{cases}$$

Geométrica

Recursiva

$$g(i) = \begin{cases} 1 & \text{se } i = 0 \\ g(i - 1) \times \text{base} & \text{se } i \geq 1 \end{cases}$$

Iterativa

$$g(i) = \begin{cases} 1 & \text{se } i = 0 \\ g(0) \times \text{base}^i & \text{se } i \geq 1 \end{cases}$$

Fibonacci

Recursiva

$$f(i) = \begin{cases} 0 & \text{se } i = 0 \\ 1 & \text{se } i = 1 \\ f(i - 1) + f(i - 2) & \text{se } i > 1 \end{cases}$$

Iterativa

$$\begin{aligned}\Phi &= (1 + \sqrt{5})/2 \approx 1.61803 \\ \hat{\Phi} &= (1 - \sqrt{5})/2 \approx -0.61803 \\ f(i) &= \frac{1}{\sqrt{5}}(\Phi^i - \hat{\Phi}^i)\end{aligned}$$

Fibonacci

Exemplos

$$f(0) = \frac{1}{\sqrt{5}}(\Phi^0 - \hat{\Phi}^0) = 0$$

$$f(1) = \frac{1}{\sqrt{5}}(\Phi^1 - \hat{\Phi}^1) = 1$$

$$f(10) = \frac{1}{\sqrt{5}}(\Phi^{10} - \hat{\Phi}^{10})$$

$$= \frac{1}{\sqrt{5}}(122,98883 - 0,00813) = 54,998 \approx 55$$

Josephus

Fórmula iterativa? Para os curiosos.

Concrete Mathematics. Graham, Knuth, Patashnik.

Projeto Orientado a Objetos

Recursão versus Iteração

O uso de fórmulas iterativas induziu o projeto orientado a objetos realizado até aqui. Hierarquia, reuso de atributos, uso de polimorfismo.

Se o projetista optar por utilizar fórmulas recursivas para o cálculo dos termos das progressões matemáticas, podem ocorrer mudanças significativas no POO de Progressoes?

Observem que a implementação de `iesimoTermo(...)` localiza-se em `Progressao` e se vale da implementação polimórfica de `proxTermo` e `inicia` para manter-se imutada durante a evolução do projeto, mesmo com a adição de novas progressões.

Projeto Orientado a Objetos

Recursão versus Iteração

O uso de fórmulas iterativas permite implementações de `iesimoTermo` e `proxTermo` que são independentes entre si. Por um lado, isso torna o cálculo dos termos computacionalmente mais eficiente, iteração é, em geral, menos custosa computacionalmente que recursão. Por outro lado, como cada progressão utiliza uma fórmula iterativa **diferente** para o seu cálculo, `iesimoTermo` deixa de ser imutável, geral, e deve agora ser implementada de forma diferente para cada uma das progressões e nas classes específicas de cada progressão. O que acontece com `inicia`? Avaliem.

Projeto Orientado a Objetos

Recursão versus Iteração

Esta discussão sobre formas recursivas versus formas iterativas para progressões ilustra um ponto importante de projeto orientado a objetos: **a escolha do algoritmo e das estruturas de dados tem influência sobre o projeto e pode alterá-lo significativamente.**

Em contraste, o modelo resultante da análise está em um nível de abstração mais alto, não se importa com detalhes algorítmicos. A fase de projeto orientado a objetos é a fase que terá a maior influência concreta sobre os modelos orientados a objetos vindos da análise porque é nela que a abstração *se materializa*, vira *código*.