PROGRAMAÇÃO ORIENTADA A OBJETOS

Recursão

Prof. Emanoel Barreiros

UNIVERSIDADE DE PERNAMBUCO

Recursão

- Até agora, a maioria dos nossos programas foram escritos com métodos que chamam uns aos outros em uma estrutura bem simples
- Para alguns problemas, podemos pensar em uma maneira diferente para resolve-los
- Em algumas situações, pode ser muito interessante um método que invoque a si mesmo repetidas vezes. Chamamos esses métodos de recursivos.

Conceito

- Quando um método recursivo é chamado para resolver um problema, ele é capaz apenas de resolver o caso mais simples, chamado de caso base
- Se o método for chamado para resolver o caso base, ele retorna um resultado.
- Se o método for chamado para resolver um caso mais complexo, ele irá dividir o problema em pedaços menores e mais triviais
  - Um pedaço que ele sabe como resolver, e um pedaço que ele não sabe como resolver, mas que é menor que o pedaço inicial
  - Essa divisão em um problema menor e a nova chamada do método a ele mesmo é chamado de **passo recursivo**

Conceito

- O passo recursivo executa enquanto a chamada anterior ainda está ativa, i.e., ela ainda não foi finalizada
- Para que a recursão possa terminar, a execução tem que convergir para o caso base
- Quando o método reconhece o caso base, ele retorna o resultado para a chamada anterior, e assim sucessivamente até a primeira chamada

Exemplo usando recursão: fatorial

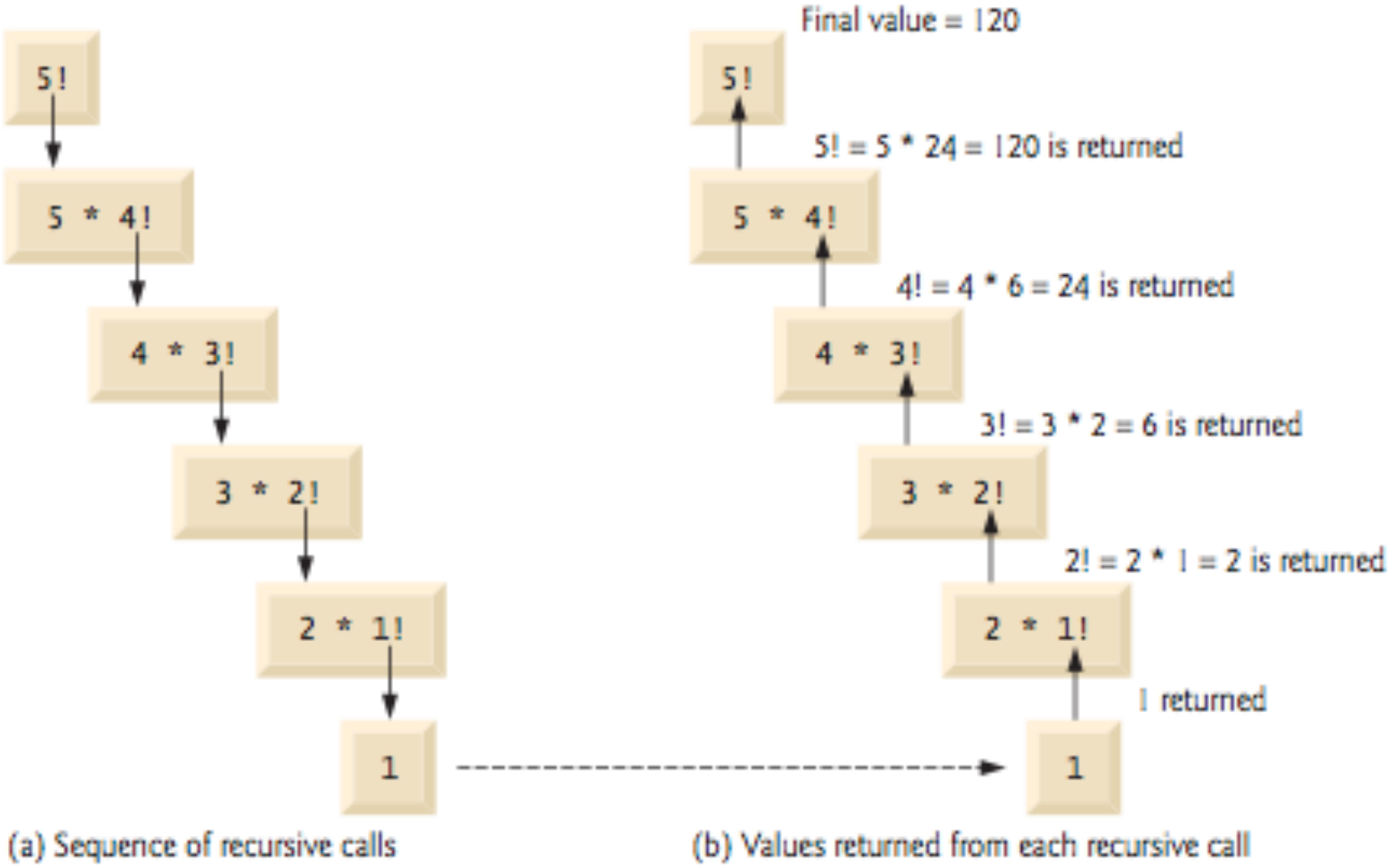
- Vamos implementar um programa que calcula uma operação matemática chamada fatorial
- O fatorial de um número inteiro não negativo *n* é definido da seguinte maneira:
  - n! = n * (n-1) * (n-2) * … * 1

- Onde 1! e 0! são iguais a 1
- Por exemplo, 5! é igual a 5*4*3*2*1 = 120

Exemplo usando recursão: fatorial

- Olhando para o cálculo do fatorial como um problema recursivo, temos que:
  - n! = n * (n-1)!, ou seja
  - 5! = 5 * 4!
  - = 5 * 4 * 3!
  - = 5 * 4 * 3 * 2!
  - = 5 * 4 * 3 * 2 * 1!
  - = 5 * 4 * 3 * 2 * 1
  - = 5 * 4 * 3 * (2 * 1)
  - = 5 * 4 * (3 * 2)
  - = 5 * (4 * 6)
  - = 5 * 24
  - = 120

Pilha de chamadas recursivas



(a) Sequence of recursive calls

(b) Values returned from each recursive call

Exercício rápido

- Implementem um método que calcule o fatorial

A série de Fibonacci

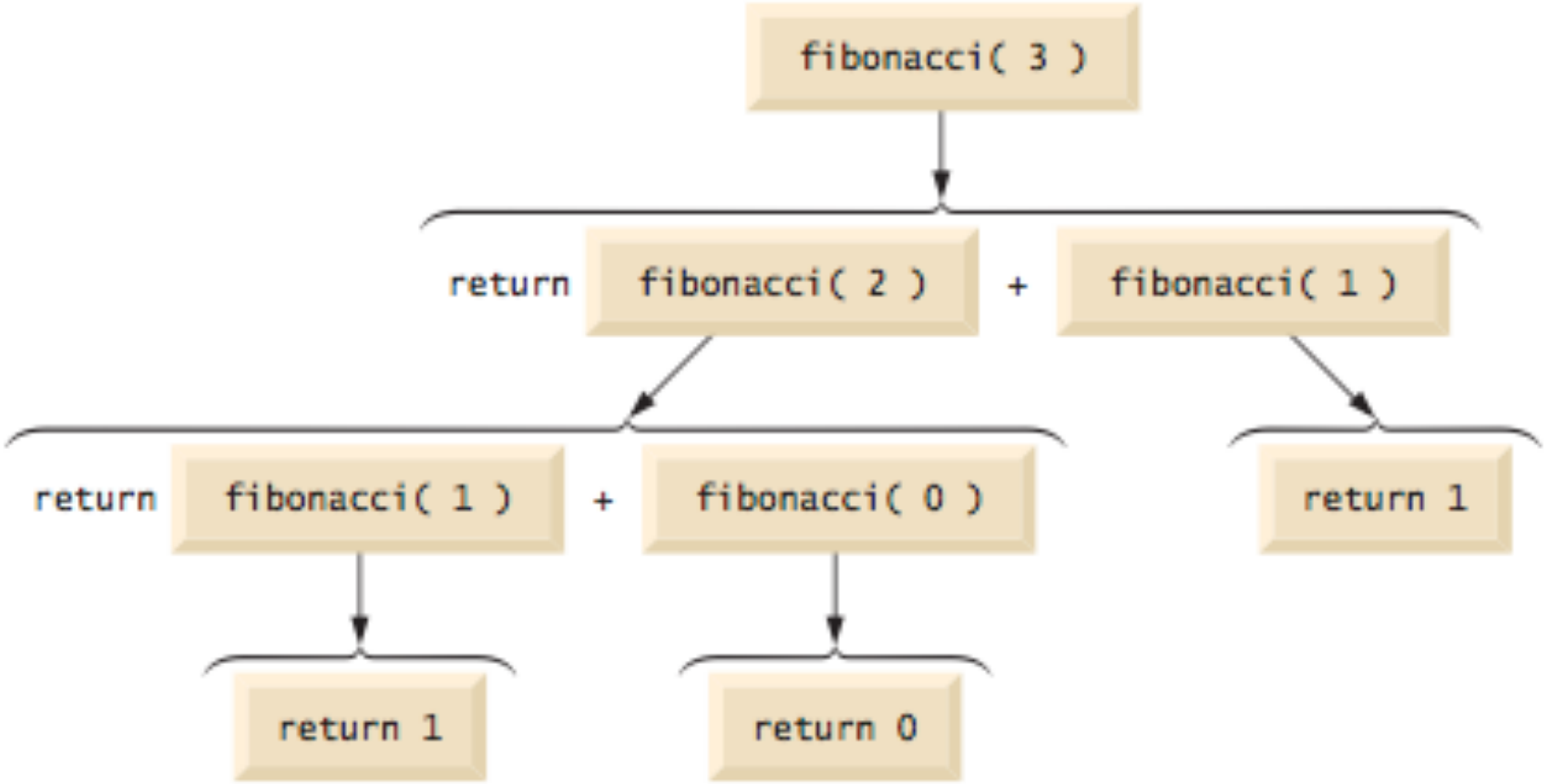
- A série de Fibonacci é outro exemplo clássico de recursão
- A série de Fibonacci inicia-se com os números 0 e 1, e determina que todos os próximos números na sequencia devem ser a soma dos dois números anteriores, logo:
  - 0 1 1 2 3 5 8 13 21 34 …

A série de Fibonacci

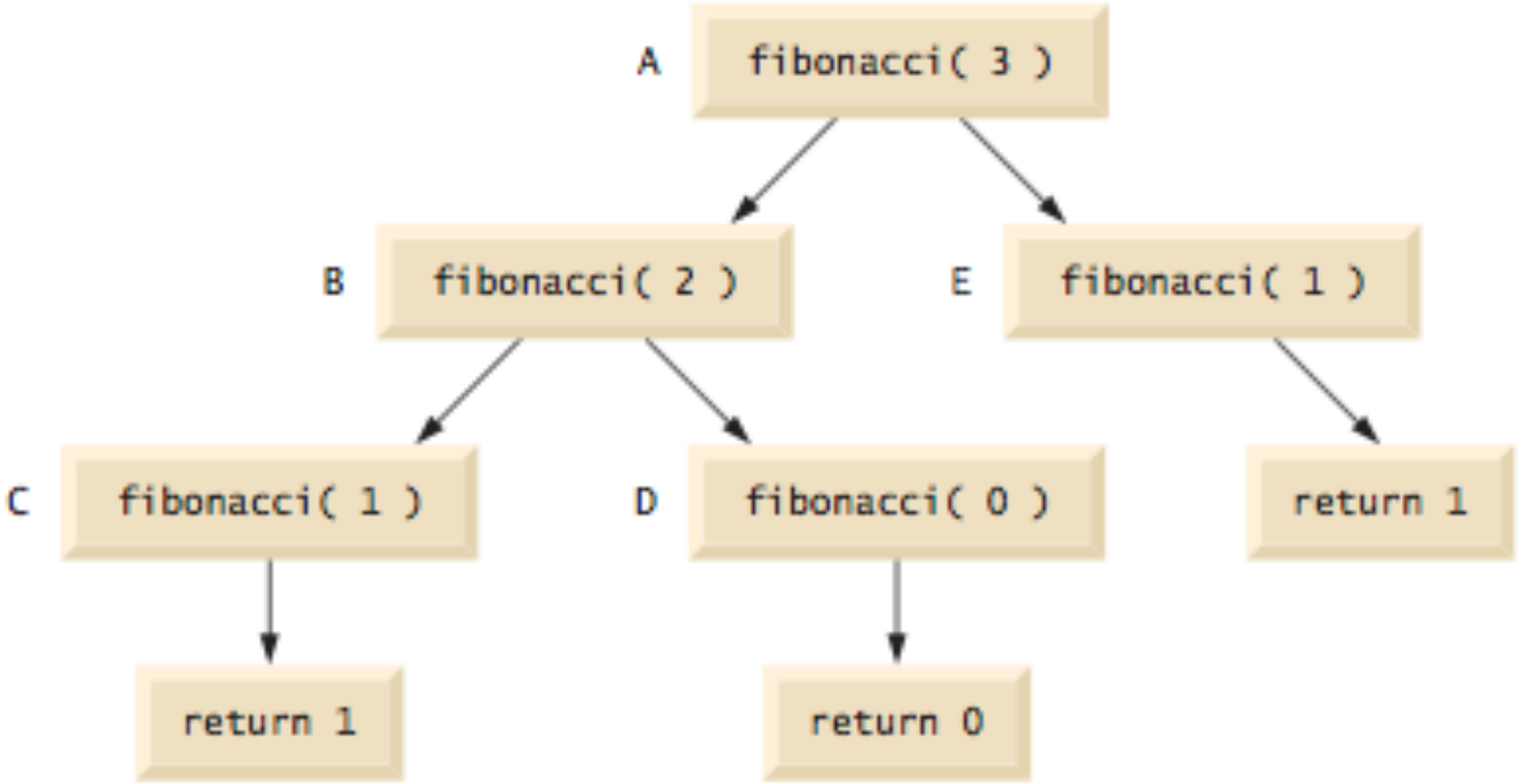
- A série de Fibonacci pode ser definida recursivamente da seguinte forma:
  - fibonacci(0) = 0
  - fibonacci(1) = 1
  - fibonacci(n) = fibonacci(n – 1) + fibonacci(n – 2)

- Como pode-se observar, existem dois casos base para a sequencia de Fibonacci, para o 0 a resposta é 0 e para o 1 a resposta é 1
- Implemente um algoritmo que resolva a sequencia de Fibonacci agora!

Analisando as chamadas ao método do Fibonacci



Ordem em que as chamadas são realizadas



Pilha de chamadas aos métodos