

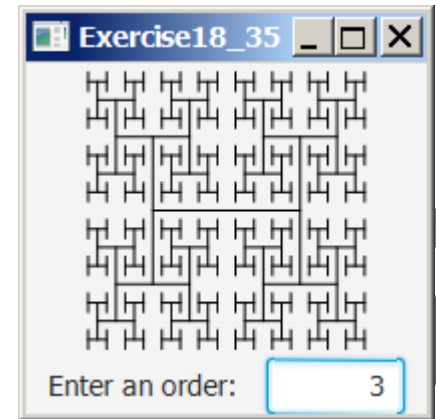
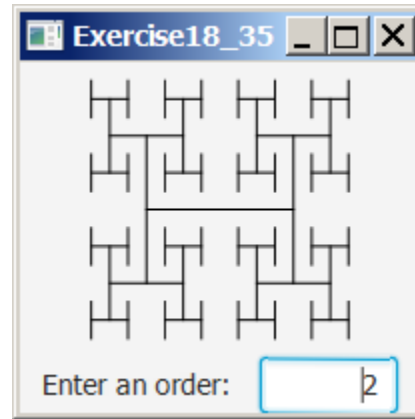
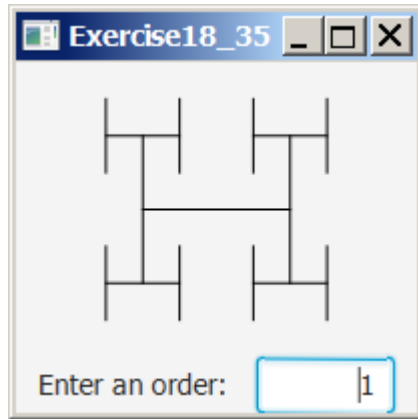
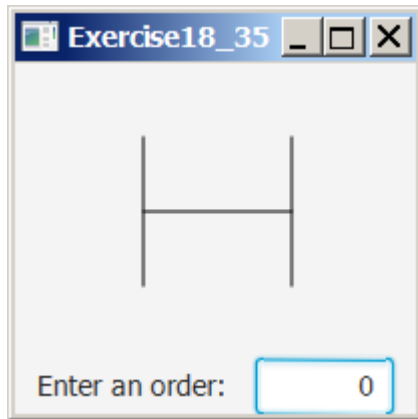
Recursion

Suppose you want to find all the files under a directory that contains a particular word. How do you solve this problem? There are several ways to solve this problem. An intuitive solution is to use recursion by searching the files in the subdirectories recursively.



Motivations

H-trees, depicted in Figure 18.1, are used in a very large-scale integration (VLSI) design as a clock distribution network for routing timing signals to all parts of a chip with equal propagation delays. How do you write a program to display H-trees? A good approach is to use recursion.



Computing Factorial

$\text{factorial}(0) = 1;$

$\text{factorial}(n) = n * \text{factorial}(n-1);$

$n! = n * (n-1)!$

$0! = 1$

ComputeFactorial

Run



Computing Factorial

factorial(4)

factorial(0) = 1;

factorial(n) = n*factorial(n-1);



Computing Factorial

$$\text{factorial}(4) = 4 * \text{factorial}(3)$$

$$\text{factorial}(0) = 1;$$

$$\text{factorial}(n) = n * \text{factorial}(n-1);$$



Computing Factorial

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2)\end{aligned}$$

$$\text{factorial}(0) = 1;$$

$$\text{factorial}(n) = n * \text{factorial}(n-1);$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2) \\ &= 4 * 3 * (2 * \text{factorial}(1))\end{aligned}$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2) \\ &= 4 * 3 * (2 * \text{factorial}(1)) \\ &= 4 * 3 * (2 * (1 * \text{factorial}(0)))\end{aligned}$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2) \\ &= 4 * 3 * (2 * \text{factorial}(1)) \\ &= 4 * 3 * (2 * (1 * \text{factorial}(0))) \\ &= 4 * 3 * (2 * (1 * 1))\end{aligned}$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2) \\ &= 4 * 3 * (2 * \text{factorial}(1)) \\ &= 4 * 3 * (2 * (1 * \text{factorial}(0))) \\ &= 4 * 3 * (2 * (1 * 1)) \\ &= 4 * 3 * (2 * 1)\end{aligned}$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * 3 * \text{factorial}(2) \\ &= 4 * 3 * (2 * \text{factorial}(1)) \\ &= 4 * 3 * (2 * (1 * \text{factorial}(0))) \\ &= 4 * 3 * (2 * (1 * 1)) \\ &= 4 * 3 * (2 * 1) \\ &= 4 * 3 * 2\end{aligned}$$



Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * (3 * \text{factorial}(2)) \\ &= 4 * (3 * (2 * \text{factorial}(1))) \\ &= 4 * (3 * (2 * (1 * \text{factorial}(0)))) \\ &= 4 * (3 * (2 * (1 * 1))) \\ &= 4 * (3 * (2 * 1)) \\ &= 4 * (3 * 2) \\ &= 4 * (6)\end{aligned}$$



Computing Factorial

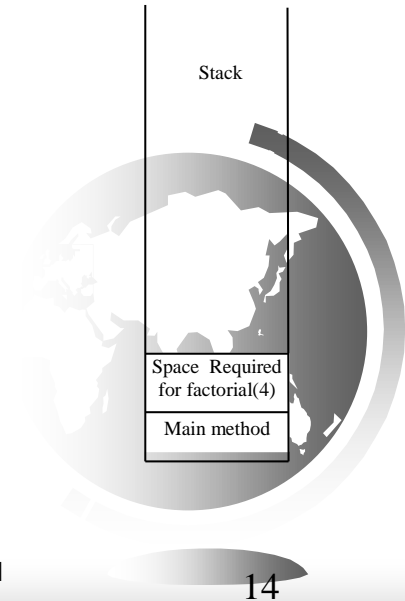
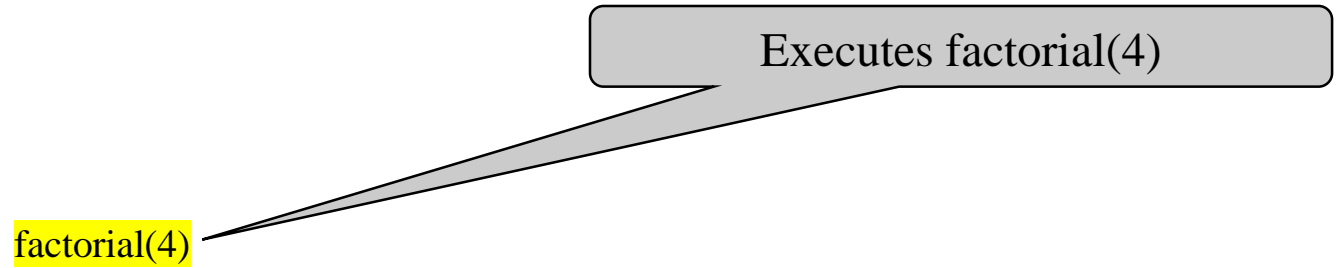
factorial(0) = 1;

factorial(n) = n*factorial(n-1);

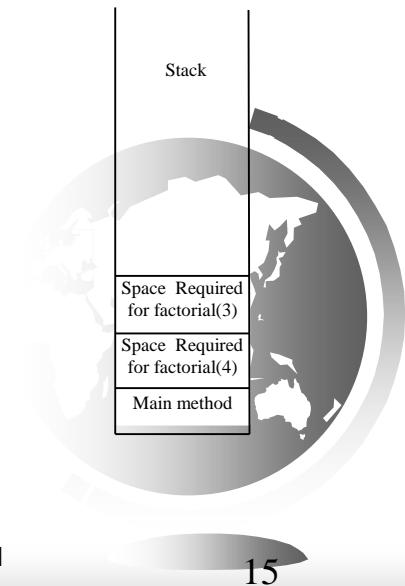
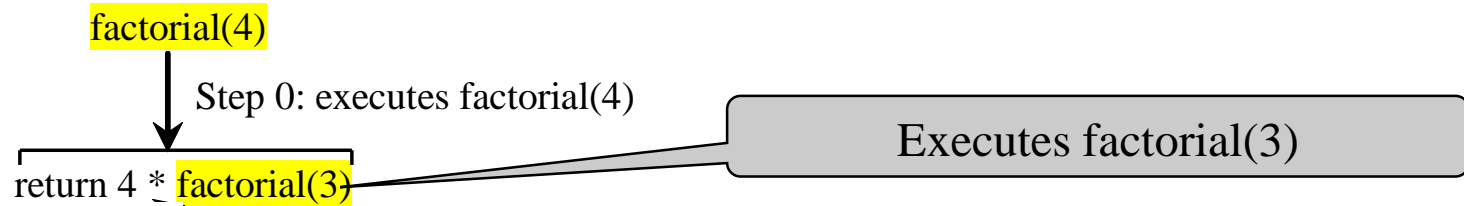
$$\begin{aligned}\text{factorial}(4) &= 4 * \text{factorial}(3) \\ &= 4 * (3 * \text{factorial}(2)) \\ &= 4 * (3 * (2 * \text{factorial}(1))) \\ &= 4 * (3 * (2 * (1 * \text{factorial}(0)))) \\ &= 4 * (3 * (2 * (1 * 1))) \\ &= 4 * (3 * (2 * 1)) \\ &= 4 * (3 * 2) \\ &= 4 * (6) \\ &= 24\end{aligned}$$



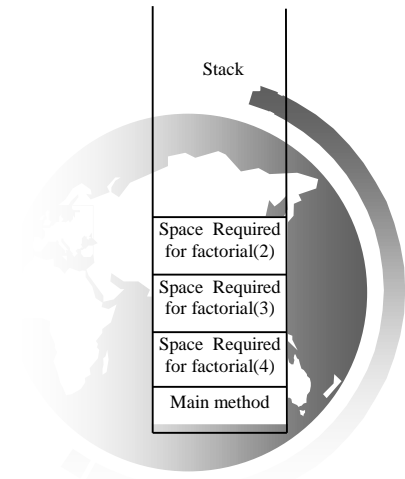
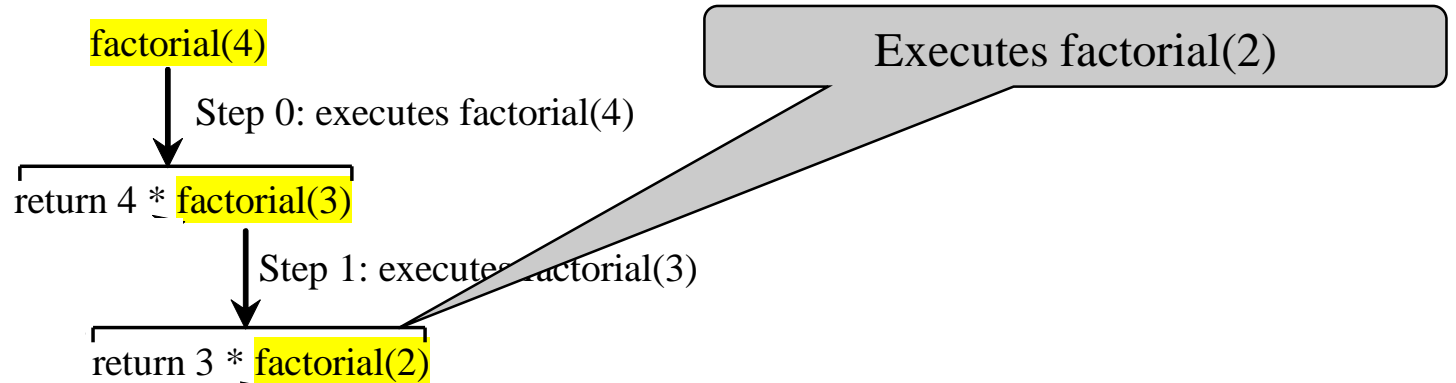
Trace Recursive factorial



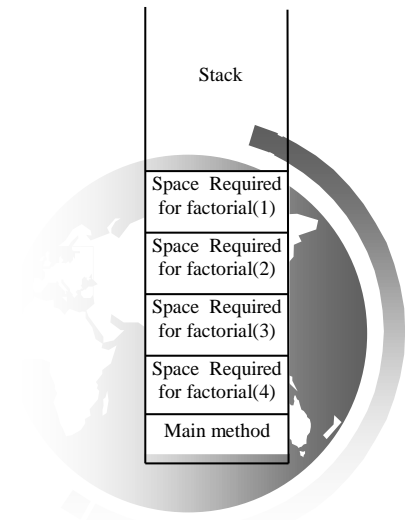
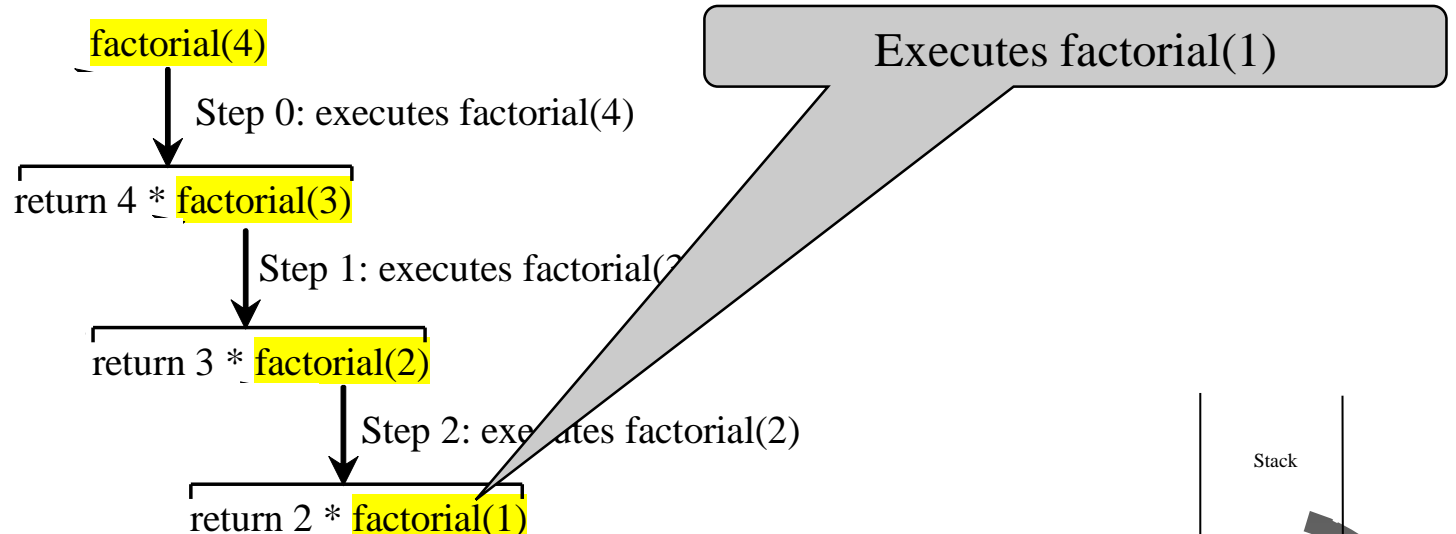
Trace Recursive factorial



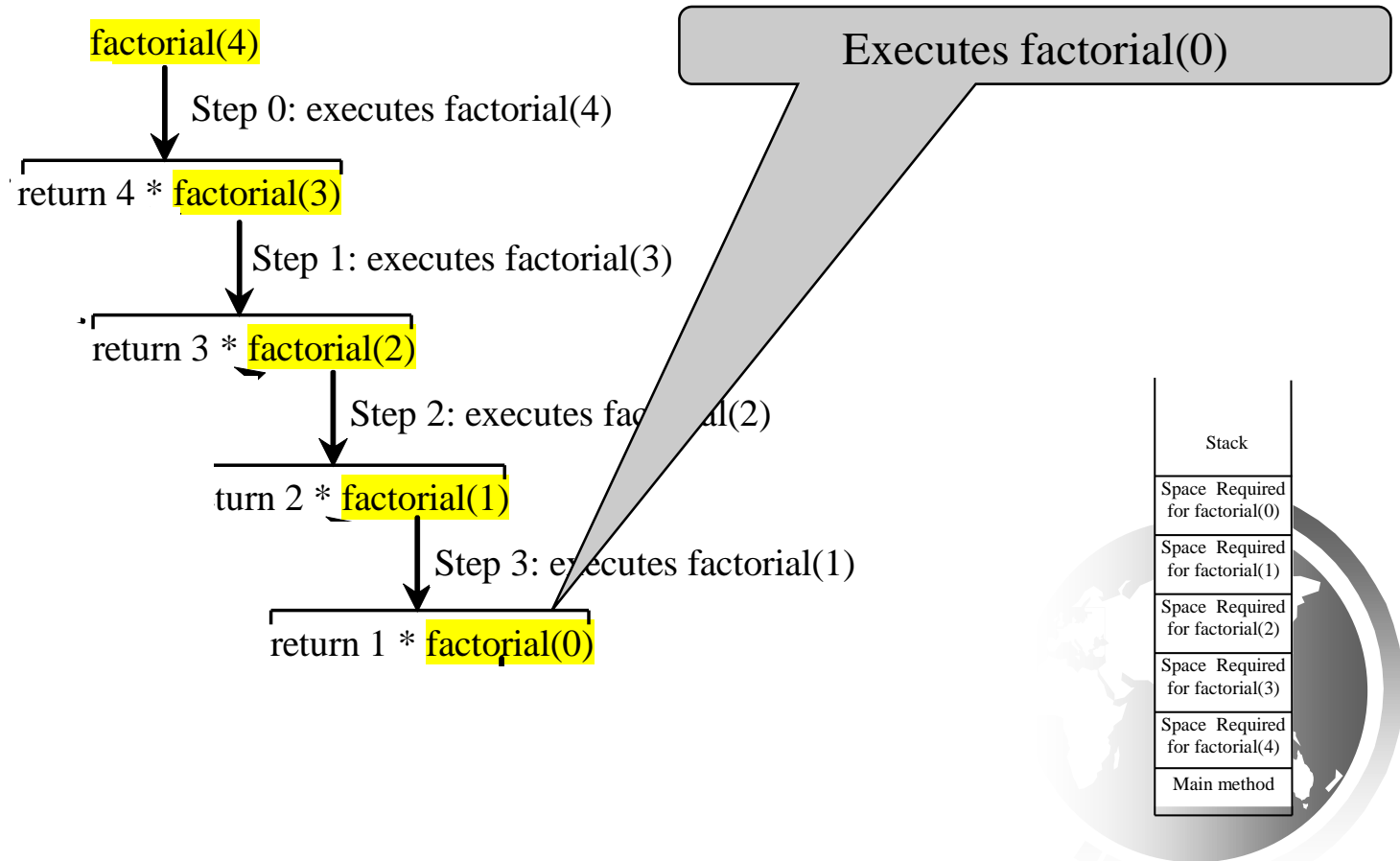
Trace Recursive factorial



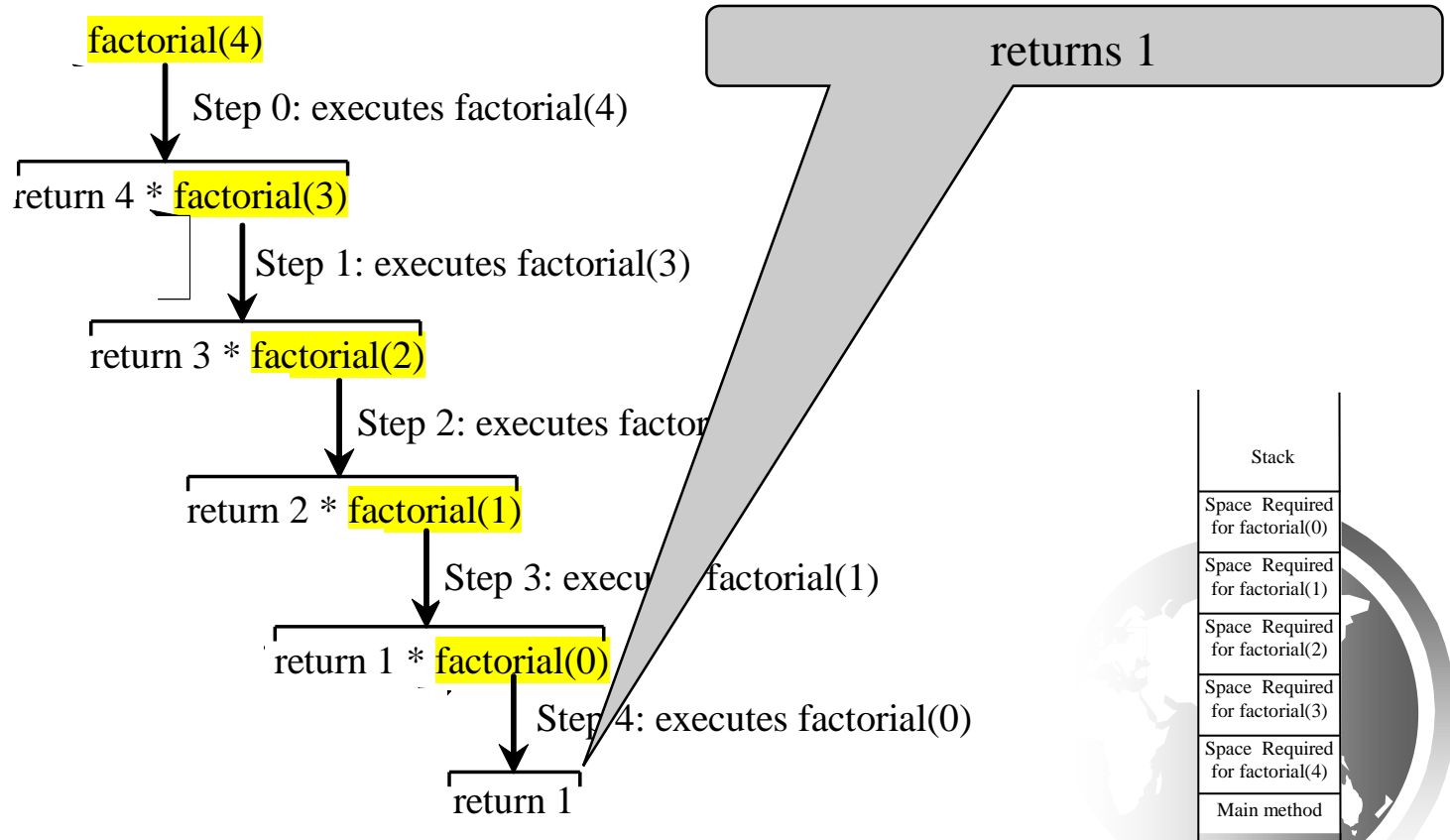
Trace Recursive factorial



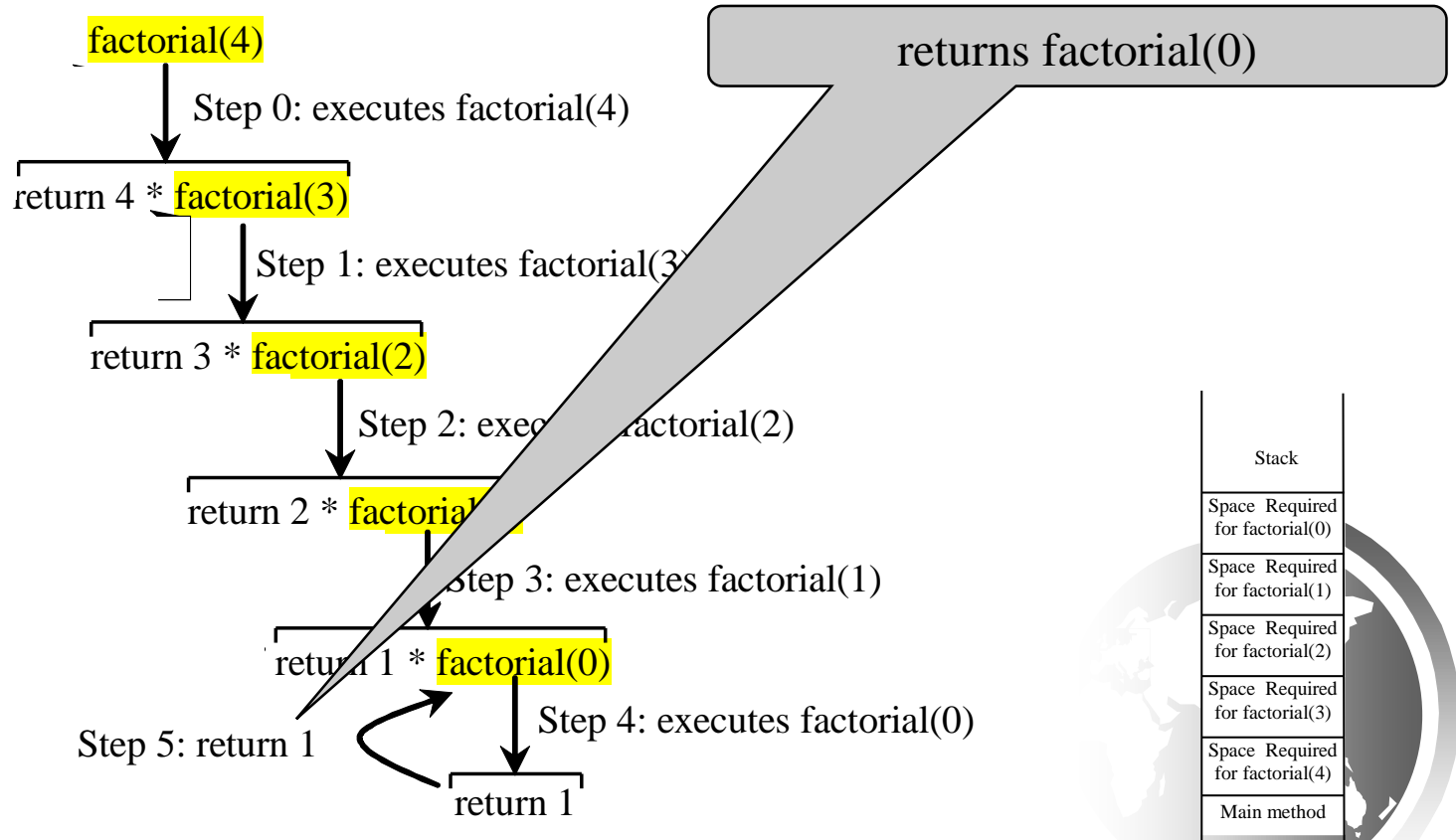
Trace Recursive factorial



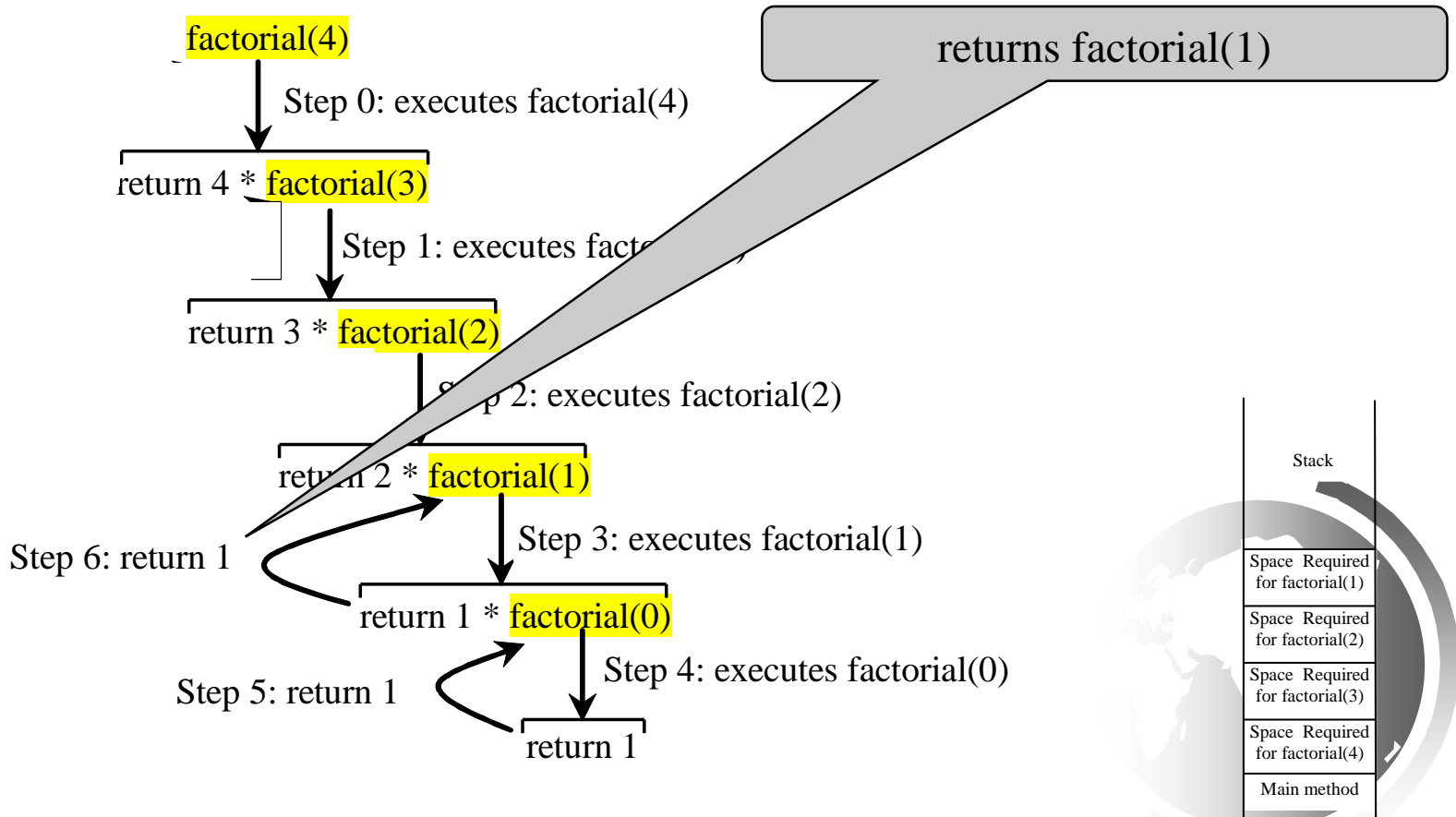
Trace Recursive factorial



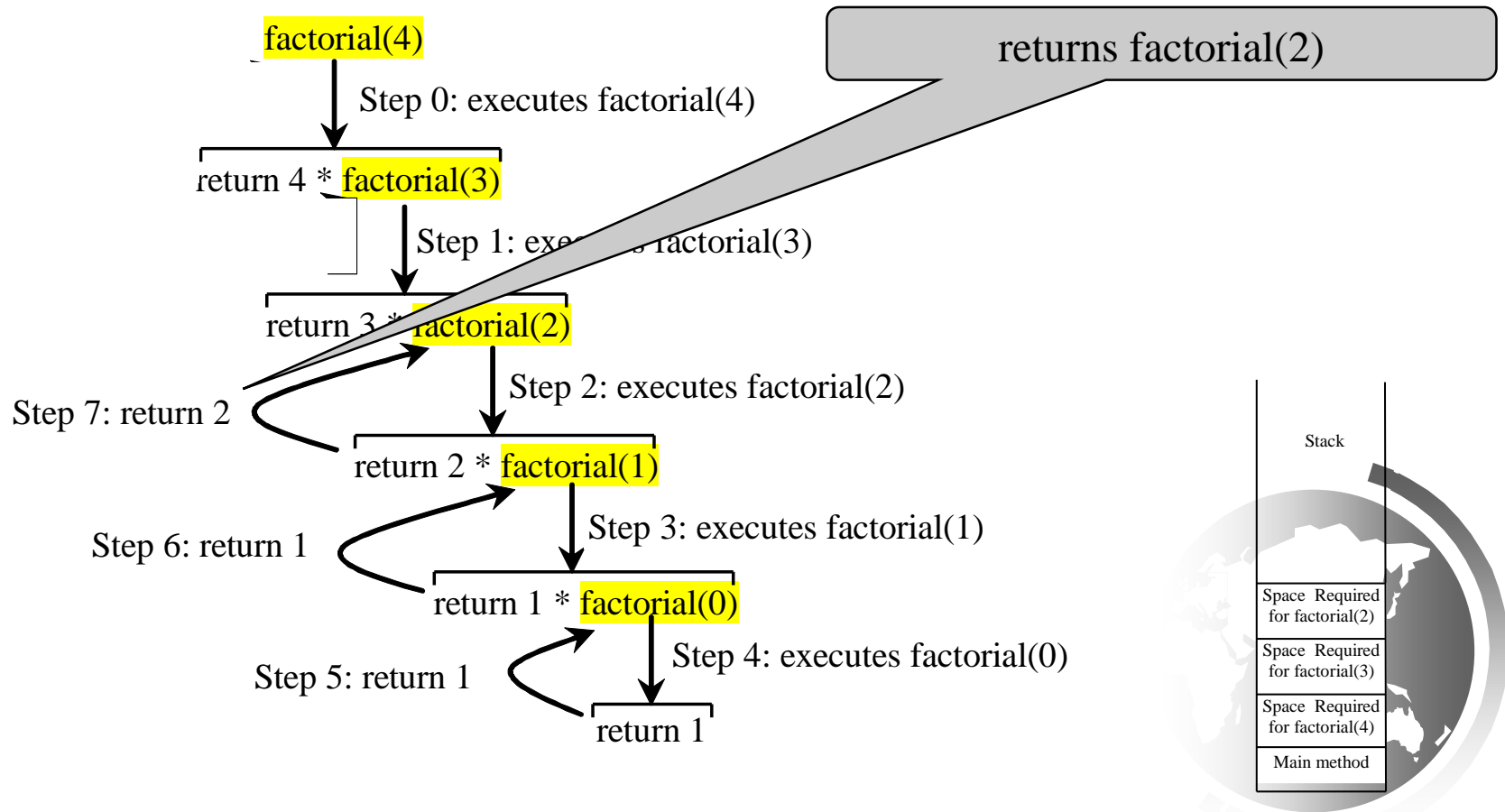
Trace Recursive factorial



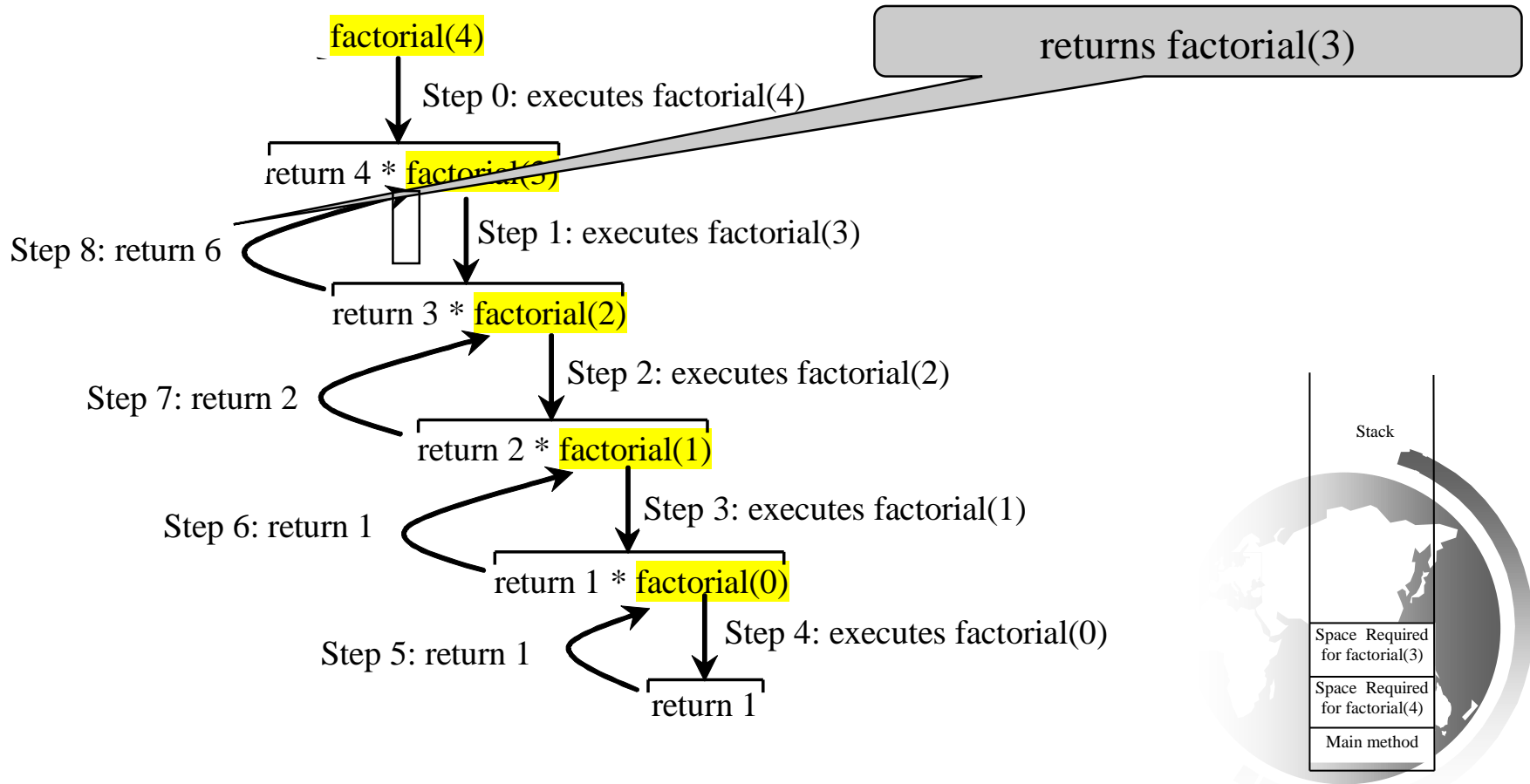
Trace Recursive factorial



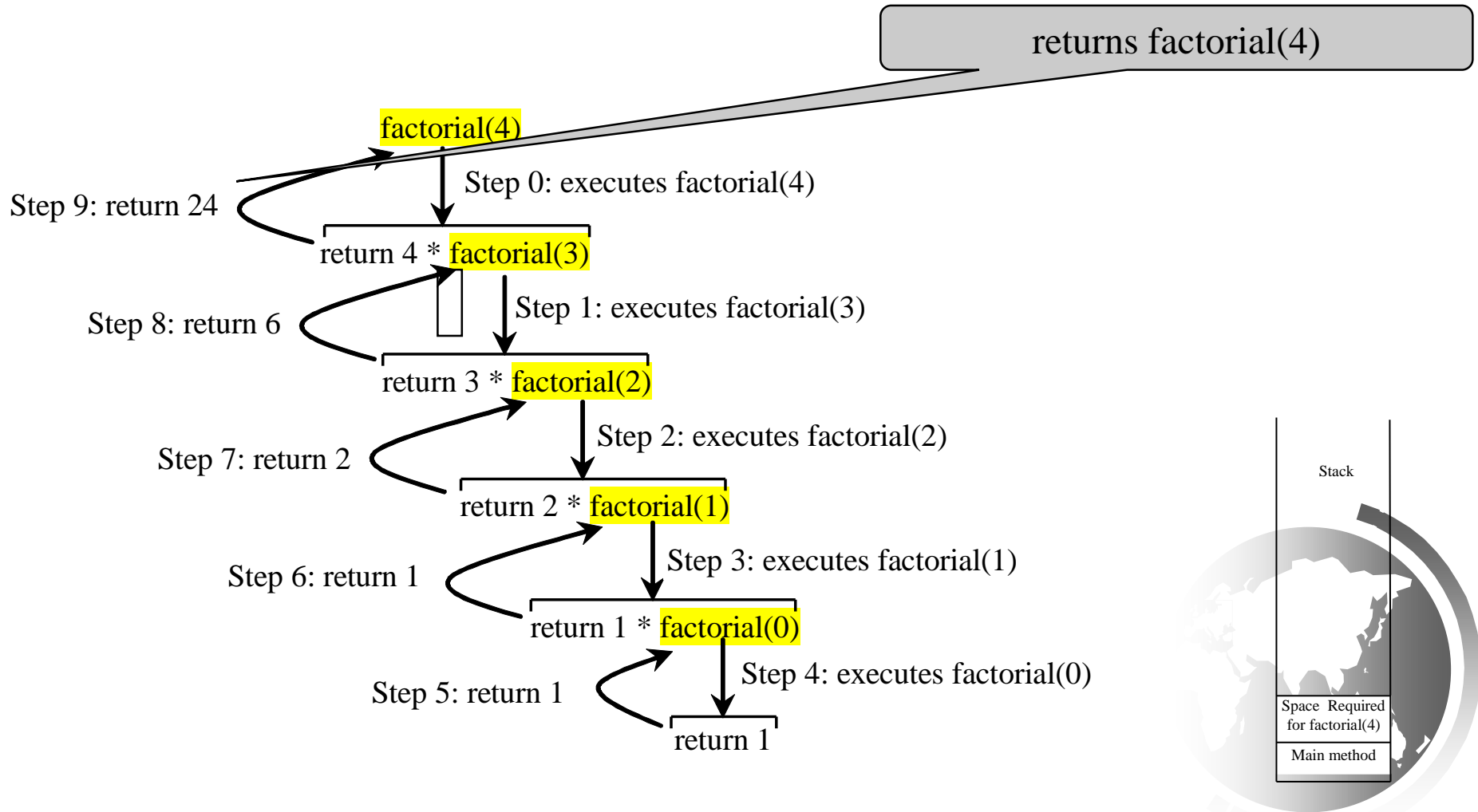
Trace Recursive factorial



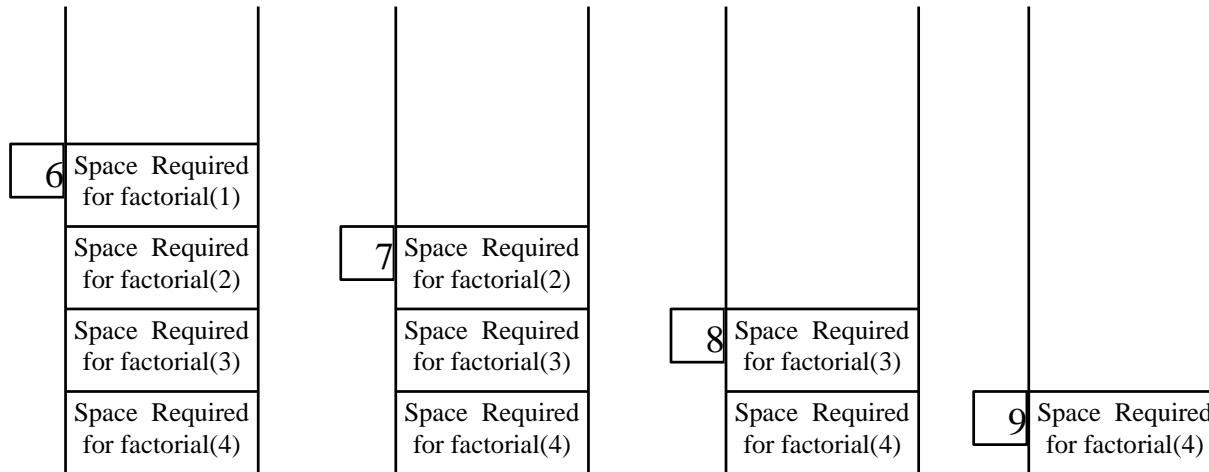
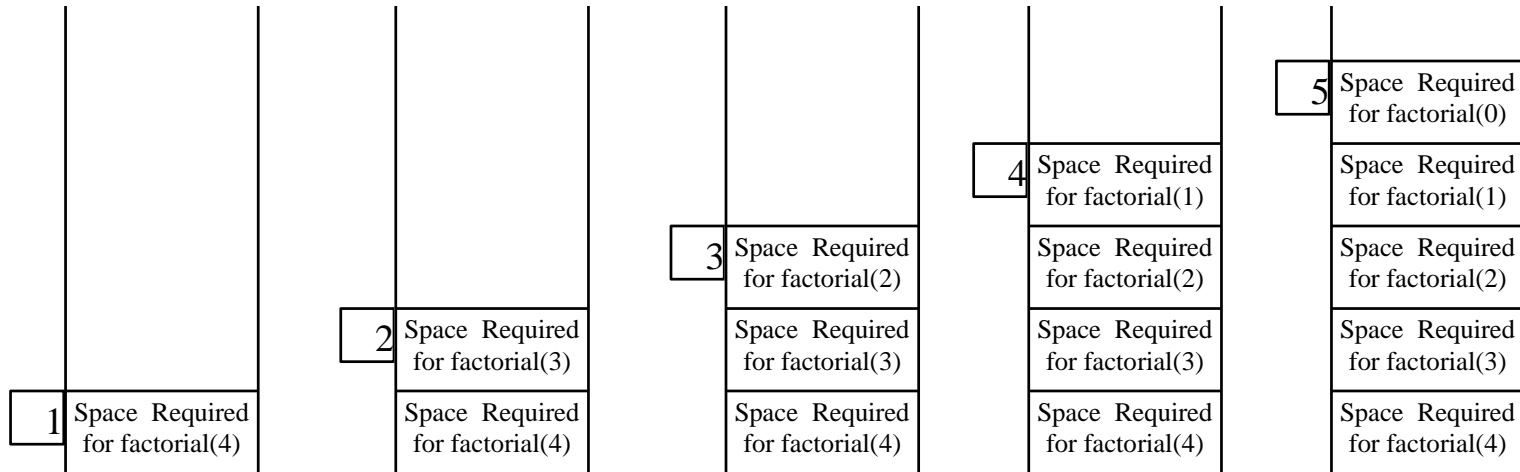
Trace Recursive factorial



Trace Recursive factorial



factorial(4) Stack Trace



Other Examples

$$f(0) = 0;$$

$$f(n) = n + f(n-1);$$



Fibonacci Numbers

Fibonacci series: 0 1 1 2 3 5 8 13 21 34 55 89...

indices: 0 1 2 3 4 5 6 7 8 9 10 11

$\text{fib}(0) = 0;$

$\text{fib}(1) = 1;$

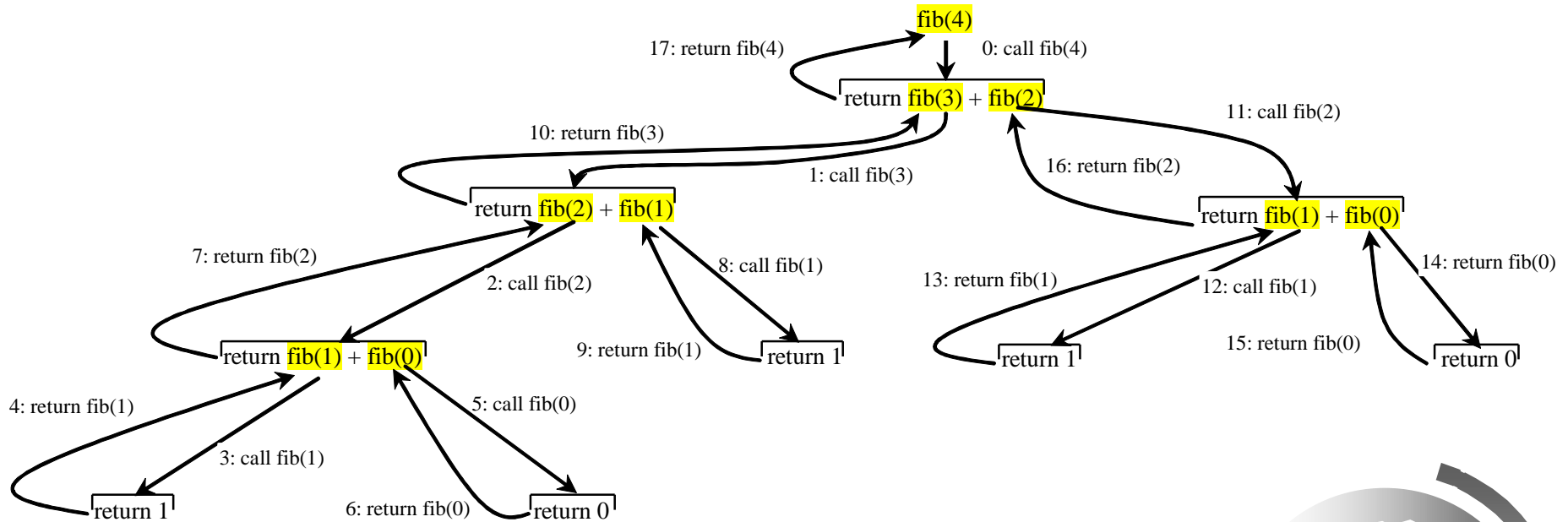
$\text{fib}(\text{index}) = \text{fib}(\text{index} - 1) + \text{fib}(\text{index} - 2); \text{index} \geq 2$

$\text{fib}(3) = \text{fib}(2) + \text{fib}(1) = (\text{fib}(1) + \text{fib}(0)) + \text{fib}(1) = (1 + 0) + \text{fib}(1) = 1 + \text{fib}(1) = 1 + 1 = 2$

ComputeFibonacci

Run

Fibonacci Numbers, cont.



Characteristics of Recursion

All recursive methods have the following characteristics:

- One or more base cases (the simplest case) are used to stop recursion.
- Every recursive call reduces the original problem, bringing it increasingly closer to a base case until it becomes that case.

In general, to solve a problem using recursion, you break it into subproblems. If a subproblem resembles the original problem, you can apply the same approach to solve the subproblem recursively. This subproblem is almost the same as the original problem in nature with a smaller size.