Algoritmos e Programação Estruturada

Recursividade

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

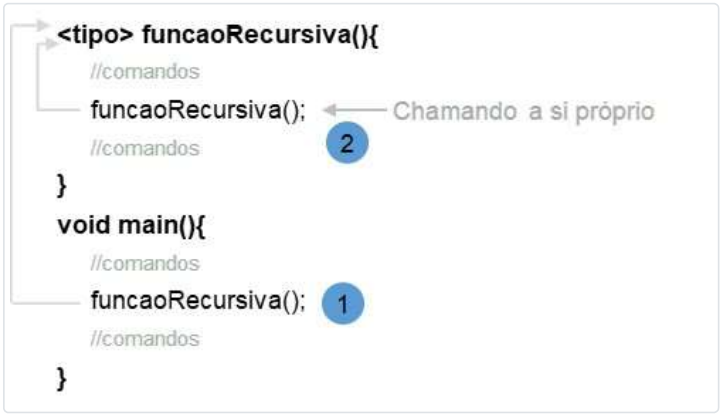
Vimos como criar uma função, qual sua importância dentro de uma implementação, estudamos a saída de dados de uma função, bem como a entrada, feita por meio dos parâmetros. E nesta webaula, vamos ver uma categoria especial de função, chamada de funções recursivas.

Recursividade

A palavra recursividade está associada a ideia de recorrência de uma determinada situação. Em programação, as funções recursivas, quando uma função chama a si própria.

A sintaxe para implementação de uma função recursiva, nada difere das funções gerais, a diferença está no corpo da função, pois a função será invocada dentro dela mesma.

Algoritmo para função recursiva



Fonte: elaborada pela autora.

Veja a seguir alguns pontos de atenção da função recursiva.

Ponto de parada

A função recursiva chama a si própria até que um ponto de parada seja estabelecido. O ponto de parada poderá ser alcançado através de uma estrutura condicional ou através de um valor informado pelo usuário.

Instâncias

Uma função possui em seu corpo variáveis e comandos, os quais são alocados na memória de trabalho. No uso de uma função recursiva, os recursos (variáveis e comandos) são alocados (instâncias) em outro local da memória, ou seja, para cada chamada da função novos espaços são destinados a execução do programa. E é justamente por esse ponto que o ponto de parada é crucial.

Variáveis

As variáveis criadas em cada instância da função na memória são independentes, ou seja, mesmo as variáveis tendo nomes iguais, cada uma tem seu próprio endereço de memória e a alteração do valor em uma não afetará na outra.

Caso base

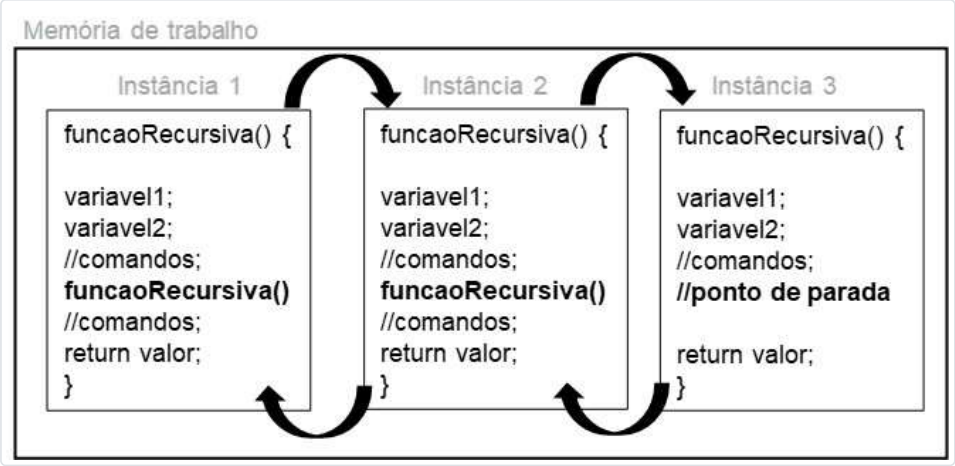
Toda função recursiva, obrigatoriamente, tem que ter uma instância com um caso que interromperá a chamada a novas instâncias. Essa instância é chamada de caso base, pois representa o caso mais simples, que resultará na interrupção.

Mecanismo da função recursiva

A instância 1 representa a primeira chamada à função `funcaoRecursiva()`, que por sua vez, possui em seu corpo um comando que chama a si mesma, nesse momento é criada a segunda instância dessa função na memória de trabalho. Um novo espaço é alocado, com variáveis e comandos, como a função é recursiva, novamente ela chama a si mesma, criando então a terceira instância da função.

Dentro da terceira instância, uma determinada condição de parada é satisfeita, nesse caso, a função deixa de ser instanciada e passa a retornar valores. A partir do momento que a função recursiva alcança o ponto de parada, cada instância da função passa a retornar seus resultados para a instância anterior (a que fez a chamada). No caso, a instância três retornará para a dois e, a dois, retornará para a um.

Mecanismo da função recursiva



Fonte: elaborada pela autora.

Veja no exemplo uma função recursiva que faz a somatória dos antecessores de um número, inteiro positivo, informado pelo usuário, ou seja, se o usuário digitar 5, o programa deverá retornar o resultado da soma 5 + 4 + 3 + 2 + 1 + 0. A função deverá somar até o valor zero, portanto esse será o critério de parada.

Explicação

Técnica de recursividade ou estruturas de repetição?

A técnica de recursividade pode substituir o uso de estruturas de repetição, tornando o código mais elegante, do ponto de vista das boas práticas de programação. Entretanto, as funções recursivas podem consumir mais memória que as estruturas iterativas.

Portanto, a recursividade é indicar quando um problema maior, pode ser dividido em instâncias menores do mesmo problema, porém considerando a utilização dos recursos computacionais que cada método empregará.

Veja um exemplo clássico de cálculo do fatorial. O fatorial de um número qualquer N, consiste em multiplicações sucessivas até que N seja igual ao valor unitário, ou seja, 5! = 5 . 4 . 3 . 2 .1, que resulta em 120.

Explicação

Recursividade em cauda

O mecanismo da função recursiva é custoso para o computador, pois tem que alocar recursos para as variáveis e comandos da função, procedimento chamado de **empilhamento**, e tem também que armazenar o local onde foi feita a chamada da função (OLIVEIRA, 2018). Para usar de forma mais otimizada a memória, existe uma alternativa chamada recursividade em cauda.

> Nesse tipo de técnica a recursividade funcionará como uma função iterativa (OLIVEIRA, 2018). Uma função é caracterizada como recursiva em cauda quando a chamada a si mesmo é a última operação a ser feita no corpo da função. Nesse tipo de função, o caso base costuma ser passado como parâmetro, o que resultará em um comportamento diferente.

Veja no exemplo como implementar o cálculo do fatorial usando essa técnica. Na linha 3, a função recursiva em cauda retorna o fatorial, sem nenhuma outra operação matemática e passa o número a ser calculado e o critério de parada junto. Foi preciso criar uma função auxiliar para efetuar o cálculo.

Explicação

Nesta webaula vimos sobre o assunto de funções recursivas. É importante que o desenvolvedor tenha um bom repertório de técnicas para solucionar os mais diversos problemas, sendo a recursividade uma delas.