Recursividade ou Iteratividade? - Parte I

Mário Leite

...

Na implementação do código de um programa é muito comum a criação de módulos responsáveis pelas tarefas funcionais com o objetivo de se obter fácil manutenção, maior legibilidade e modularidade do sistema; e isto vale para qualquer paradigma de programação; até para orientação a objetos, mesmo já sendo esse paradigma modularizado pelas classes de objetos. Para executar as diversas tarefas mais específicas do programa, esses módulos (sub-rotinas) podem ser utilizados de duas maneiras, mais formalmente: como *funções* ou como *procedimentos*. O objetivo principal de utilizar sub-rotinas é o de modularizar o programa, tornando-o mais manutenível e mais extensível. Por exemplo, num sistema com CRUD, se o módulo "Imprimir Relatórios" estiver com algum problema, a sub-rotina responsável por essa tarefa poderá ser desativada temporariamente, e o módulo "Cadastrar Clientes" continuar sendo utilizado normalmente enquanto a manutenção é feita naquele módulo isolado, pois não teria sentido encerrar o sistema e não poder cadastrar novos clientes, apenas pelo fato da rotina de relatórios estar com problemas. Seria o mesmo que ter de sair de casa, apenas para consertar um vazamento no banheiro! E dependendo da arquitetura do programa e da habilidade do programador, as sub-rotinas podem ser, teoricamente implementadas, de duas maneiras: *iterativamente* ou *recursivamente*. No primeiro caso é repetido um processo novo, e de novo, no início de cada nova iteração com o resultado da iteração anterior, permitindo acelerar a execução do programa. No segundo caso, de operação recursiva, o processo se repete até que uma instrução de término é executada dentro do processamento: o chamado "ponto de parada", que é crucial neste caso. De uma maneira mais objetiva, pode-se definir recursividade como "*um processo em que um módulo se auto executa vária vezes*"; isto é, ele chama a si mesmo como se estivesse num *loop*, porém finito.

Existem muitos exemplos de sub-rotinas que são empregadas dentro dos programas, principalmente nos casos que envolvam cálculos numéricos, como: *fatorial*, *classificação rápida*, *soma de números inteiros*, *validação de números primos*, *quadrado perfeito*, *cálculo de raiz quadrada*, *Função de Ackermann*, e até em jogos, como a "Torre de Hanói". Objetivamente, a questão que se coloca é a seguinte: qual técnica utilizar na criação de um módulo: ***Iteração*** ou ***Recursão***? A resposta desta questão pode envolver até a paixão por uma ou outra solução, ou mesmo dependendo do tipo do profissional como, por exemplo, um matemático e um programador. O matemático certamente deverá optar por uma técnica que lhe dê mais prazer na discussão teórica; e a recursividade será sempre sua preferência, na medida em que estaria interessado em explorar ao máximo sua capacidade de abstração com a matemática pura. Já o programador poderá optar por uma ou por outra, levando em conta suas habilidades na codificação para conseguir o melhor programa. De qualquer forma, no caso do programador, ele deve sempre se lembrar que o programa não é feito para ele, e sim para quem realmente importa: o USUARIO: o verdadeiro dono do programa e quem avalia o produto, através de sua aplicação na solução de seus problemas.

Continua na próxima Parte