Escola de Artes, Ciências e Humanidades (USP-Leste)
Algoritmos e Estruturas de Dados I - 2º Semestre de 2015
Prof. Alexandre da Silva Freire (afreire@ime.usp.br – sala 322K-I1)

Aula 7 - 01/09 - Recursão

Torres de Hanói

"Para entender o que é recursão, precisamos primeiro entender o que é recursão." (Autor desconhecido)

A frase citada acima pode ser o prelúdio de uma confusão que possivelmente se formará na sua mente quando iniciarmos nossos estudos sobre recursão. Nos esforçaremos para evitar tal estado de espírito e aconselhamos que, antes de iniciar esse estudo, você relaxe sua mente por alguns instantes, visualizando uma bolha azul e dourada envolvendo sua cabeça.

Para dissolver as más impressões que a frase citada acima possa nos ter causado, conscientizemo-nos de que **os sábios sempre preferem simplicar as coisas ao invés de complicá-las, como fazem os tolos**, e pautando-nos por esse princípio é que utilizamos a recursão na resolução de muitos problemas. Ao invés de apresentarmos o tema de forma genérica, trataremos de um problema particular que ilustra o que é recursão.

Torres de Hanói é o nome de um jogo composto por uma base contendo três pinos (as torres), sendo que em um deles inicialmente estão empilhados alguns discos em ordem decrescente de diâmetro, conforme mostra a figura abaixo[1]

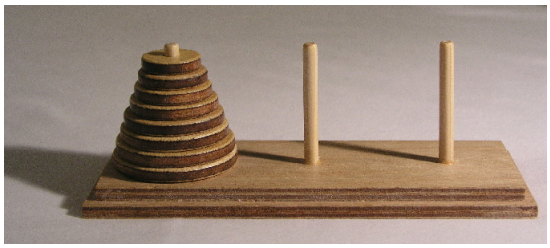

Figura 1: Torres de Hanói com 8 discos.

O objetivo do jogo é transferir todos os discos do pino "inicial" para o pino "final", utilizando um terceiro pino como "auxiliar", fazendo o menor número possível de movimentos e respeitando as seguintes regras: (1) só é permitido transferir um único disco por vez de um pino para outro; (2) só é permitido remover o disco que está no topo de um pino; (3) um disco não pode ser empilhado sobre discos menores.

[1]Imagem retirada do sítio https://en.wikipedia.org/wiki/Tower_of_Hanoi.

Há um simulador deste jogo disponível na página da Universidade Federal do Rio Grande do Sul: `http://www.ufrgs.br/psicoeduc/hanoi/`.

Se quisermos resolver esse problema para k discos, podemos proceder da seguinte maneira: (1) movemos os $k-1$ discos menores do pino "inicial" para o pino "auxiliar"; (2) movemos o disco que ficou no pino "inicial" para o pino "final"; (3) movemos os $k-1$ discos do pino "auxiliar" para o pino "final". **A solução é tão simples que custamos a acreditar que ela está correta!**

Agora a frase citada no início está começando a fazer sentido, pois para resolver o problema Torres de Hanói, primeiro precisamos resolver o problema Torres de Hanói. Mais precisamente, para resolvermos uma instância com k discos do Torres de Hanói, primeiro precisamos resolver uma instância com $k-1$ discos do Torres de Hanói. Mas como resolveremos uma instância com $k-1$ discos? Da mesma forma! Ou seja, resolvendo primeiro uma instância com $k-2$ discos. E assim sucessivamente...

É aí que nossa cabeça começa a entrar em parafuso, pois começamos a ficar com a impressão de que estamos sendo "enrolados" e o problema nunca será efetivamente resolvido, pois esse processo parece não ter fim. Porém, uma análise mais cuidadosa nos mostrará que isso não é verdade! Repare que a cada *chamada recursiva* tratamos uma instância com um disco a menos $(k, k-1, k-2, \ldots)$, de modo que em algum momento trataremos um caso trivial que saberemos resolver sem "enrolar", ou seja, sem a necessidade de fazer nenhuma chamada recursiva. Por exemplo, se quisermos resolver o Torres de Hanói para apenas 1 disco, basta mover tal disco do pino "inicial" para o pino "final". Essa pode ser a "base da recursão" ou, em outras palavras, o caso em que resolvemos o problema sem fazer chamadas recursivas. Uma outra base ainda mais simples, que adotaremos aqui, é quando queremos resolver o problema para 0 discos. Nesse caso simplesmente não fazemos nada. Abaixo, apresentamos um pseudocódigo para resolver o Torres de Hanói de forma recursiva:

```
Hanói(inicial, final, auxiliar, nDiscos)
   Se nDiscos > 0 então
      Hanói(inicial, auxiliar, final, nDiscos-1)
      Mova do pino inicial para o final
      Hanói(auxiliar, final, inicial, nDiscos-1)
```

Quem diria que esse problema poderia ser resolvido com apenas 4 linhas de código? Os sábios sempre preferem simplicar as coisas ao invés de complicá-las, como fazem os tolos...

Exercícios

Exercício 1: Escreva um algoritmo recursivo para encontrar um elemento mínimo de um vetor de inteiros.

Exercício 2: Escreva um algoritmo recursivo para calcular a soma dos elementos de um vetor de inteiros.

Exercício 3: Escreva um algoritmo recursivo para contar quantos números pares existem em um vetor de inteiros.

Exercício 4 (Desafio): Escreva uma versão iterativa (ou seja, que não faz chamadas recursivas) do algoritmo `Hanói` apresentado acima. **Dica: use uma pilha.**