Pontifícia Universidade Católica de Minas Gerais

ICEI – Instituto de Ciências Exatas e Informática
DCC – Departamento de Ciência da Computação
Campus Belo Horizonte – Unidade Praça da Liberdade
Bacharelado em Ciência da Computação

Algoritmos e Estruturas de Dados I
Professor: Lúcio Mauro Pereira
Lista de Exercícios nº 16
22 de abril de 2024

Algoritmos recursivos

Estudar:

Obra: C: como programar. 8ed. Autor: Deitel.
Disponível na biblioteca da PUC Minas de forma física e também em *e-book*.
Estudar o Capítulo 5, seções 5.13, 5.14, 5.15: **Recursão**

Para cada item desta lista, crie uma (ou mais) função(ões) que implemente(m) a solução proposta por você para cada problema. Considere o grau de reusabilidade provido por sua solução. Em seguida, construa uma função principal (main) que permita testar a solução apresentada de diferentes formas.

Busque alternativas para simplificar a interface da função utilizando recursão indireta.

1. Construa algoritmos para calcular o k-ésimo termo Fibonacci utilizando as abordagens abaixo. Antes de iniciar a codificação, busque e planeja, cuidadosamente, as estratégias que o algoritmo deverá adotar.
   a) Algoritmo recursivo
   b) Algoritmo iterativo

2. Abaixo é apresentada a função de Ackerman, válida para valores inteiros e não negativos de *m* e *n*. Para ela, construa uma versão recursiva do algoritmo. Antes da implementação em C, verifique manualmente o seu comportamento para a chamada **A(1, 2)**.

$$A(m, n) = \begin{cases} n + 1, & \text{para } m=0 \\ A(m-1, 1), & \text{para } m>0 \text{ e } n=0 \\ A(m-1, A(m, n-1)), & \text{para } m>0 \text{ e } n>0 \end{cases}$$

3. Da obra,
C: como programar, Deitel, resolva o exercício 5.39, **Torres de Hanói**.
Obs.: Pode ocorrer de o número do exercício mudar de acordo com a edição.

(Torres de Hanói) Todo cientista computacional principiante deve lidar com determinados problemas clássicos, e o problema das Torres de Hanói (veja a Fig. 5.18) é um dos mais famosos. Diz a lenda

que em um templo do Extremo Oriente os sacerdotes estão tentando mover uma pilha de discos de um pino para outro. A pilha inicial tinha 64 discos colocados em um pino e organizados na ordem decrescente, da base para o topo. Os sacerdotes estão tentando mover a pilha desse pino para um segundo pino com a restrição de que exatamente um disco deve ser movido de cada vez e em nenhum momento um disco maior pode ser colocado sobre um disco menor.

Há um terceiro pino disponível para colocação temporária de discos. Teoricamente o mundo terminará quando os sacerdotes completarem sua tarefa, portanto há pouco estímulo para facilitarmos seus esforços.

Vamos assumir que os sacerdotes estão tentando mover os discos do pino 1 para o pino 3. Desejamos desenvolver um algoritmo que imprimirá a sequência exata de transferências dc discos de um pino para outro. Se fossemos solucionar este problema com os métodos convencionais, rapidamente nos encontraríamos perdidos lidando com os discos. Em vez disso, se solucionarmos o problema com a recursão em mente, ele se torna imediatamente viável. Mover *n* discos pode ser considerado em termos de mover apenas *n - I dis*cos (daí a recursão) como se segue:

1 Mova *n* - 1 discos do pino I para o pino 2. usando o pino 3 como área de armazenamento temporário.
2. Mova o último disco (o maior) do pino 1 para o pino 3.
3. Mova os *n* - 1 discos do pino 2 para o pino 3. usando o pino 1 como área de armazenamento temporário. O processo termina quando a última tarefa envolver mover *n* = 1 disco, i.e., o caso básico. Isso é realizado movendo simplesmente o disco para o seu destino final, sem a necessidade de uma área de armazenamento temporário.

Escreva um programa para resolver o problema das Torres de Hanói. Use uma função recursiva com quatro parâmetros:
1. O número de discos a serem movidos
2. O pino no qual esses discos estão colocados inicialmente
3. O pino para o qual essa pilha de discos deve ser movida
4. O pino a ser usado como área de armazenamento temporário Seu programa deve imprimir instruções precisas necessárias para mover os discos do pino inicial para o pino de destino. Por exemplo, para mover uma pilha de três discos do pino 1 para o pino 3, seu programa deve imprimir a seguinte sequência de movimentos:

1 —> 3 (Isso significa mover o disco do pino 1 para o pino 3.)
1 => 2
3=>2
1 => 3
2 => 1
2=>3
1 -> 3