MAC 122 – Princípios de Desenvolvimento de Algoritmos

Primeiro semestre de 2017

Lista de exercícios—Recursão

"Para fazer uma função recursiva
é preciso ter fé."
Siang Wun Song

1. (a) Faça uma função recursiva MaxMin que calcula o elemento máximo e o elemento mínimo de um vetor com n números inteiros.

   (b) Quantas comparações (em função de n) envolvendo elementos do vetor o seu algoritmo faz?

2. Considere a função abaixo que calcula o n-ésimo termo de Fibonacci:

```
int fibonacci(int n)
{
   printf("*");
   if ((n == 1) || (n == 2))
      return 1;
   return (fibonacci(n-1) + fibonacci(n-2));
}
```

   Quantas "*" são impressas no cálculo de `fibonacci(n)`?

3. Faça uma função recursiva Dígito que recebe um número inteiro n e calcula a soma dos digitos de n. Exemplo: se $n = 132$ então Dígito(n) $= 6$.

4. (a) Faça uma função recursiva que verifica se uma expressão de parênteses é bem formada.

   (b) Idem para uma expressão com parênteses, colchetes ('[','](')') e chaves ('{','}').

5. Considere a função abaixo:

```
double f(double x, double y)
{
  if (x >= y)
     return ((x+y)/2);
  return (f(f(x+2, y-1), f(x+1, y-2));
}
```

   Qual é o valor de $f(1, 10)$? Como se poderia calcular $f(a, b)$ de maneira mais simples?

6. A função de Akermann é definida da seguinte maneira:

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

Escreva uma função recursiva que recebe inteiros não negativos m e n e devolve $A(m,n)$.

7. Simule a execução do programa abaixo:

```
#include <stdio.h>
int fusc(int n)
{
  if (n <= 1) return (1);
  if (n % 2 == 0)
     return( fusc(n / 2) );
  return( fusc((n-1)/2) + fusc((n+1)/2) );
}

int main()
{
  int m = 7;
  printf("Fusc = %d\n", fusc(m));
}
```

8. Considere a seguinte função:

```
void misterio (int A[], int inic, int fim)
{
  int aux;
  while (A[fim] % 2 == 0 && inic < fim)
     fim --;
  while (A[inic] % 2 == 1 && inic < fim)
     inic++;
  if (inic < fim){
    aux = A[inic];
    A[inic] = A[fim];
    A[fim] = aux;
    misterio(A, inic, fim);
  }
}
```

(a) Simule a função Mistério para

	0	1	2	3	4	5	6	7	8
$A =$	8	10	3	6	5	2	9	1	4

Inicio= 0 e Fim = 8.

(b) O que faz a função Mistério? Quantas comparações envolvendo elementos do vetor A são feitas? Escreva um algoritmo que faz a mesma coisa com um número menor de comparações.

9. Simule a seguinte função recursiva para $N = 6$:

```
int zzz(int n)
{
  int aux;
  if (n <= 2)
    return(1);
  n--;
  aux = zzz(n);
  n--;
  return (aux + zzz(n));
}
```

O que faz a função `zzz`?

10. Escreva uma função recursiva `Tabela` que recebe como parâmetro um inteiro não negativo n e calcula um par de inteiros (x, y), onde x e y são as coordenadas de n na tabela abaixo:

	0	1	2	3	4	...	Y
0	0	2	5	9	14		
1	1	4	8	13			
2	3	7	12				
3	6	11					
4	10						
...							
X							N

11. Suponha que temos de calcular o produto de matrizes

$$M_1 \times M_2 \times \ldots \times M_n,$$

onde cada M_i é uma matriz com r_i linhas e r_{i+1} colunas. (Portanto, $r_1, \ldots, r_{n+1}$ descrevem as dimensões das matrizes.) Suponha ainda que o produto de qualquer par de matrizes será calculado pelo algoritmo usual; assim o produto de uma matriz $p \times q$ por uma matriz $q \times r$ requer $p \times q \times r$ operações de multiplicação entre números. A ordem em que a multiplicação de três ou mais matrizes é executada pode afetar sensivelmente o número total de multiplicações. Por exemplo, se $n = 4$ e $(r_1, \ldots, r_5) = (10, 20, 50, 1, 100)$ então o cálculo da expressão $M_1 \times (M_2 \times (M_3 \times M_4))$ requer 125000 multiplicações enquanto que o cálculo da expressão $M_1 \times (M_2 \times M_3)) \times M_4$ requer 2200 multiplicações. Seja $m_{i,j}$ o número mínimo de multiplicações necessárias para calcular $M_i \times \cdots \times M_j$. Temos que

$$m_{i,j} = \begin{cases} 0 & \text{se } i = j \\ \min_{i \le k \le j}\{m_{i,k} + m_{k+1,j} + r_i \times r_{k+1} \times r_{j+1}\} & \text{se } i < j. \end{cases}$$

Escreva uma função recursiva que recebe n e a seqüência $(r_1, \ldots, r_{n+1})$ e calcula $m_{1,n}$.

12. Escreva uma função que dado n imprime todas as permutações dos números inteiros $1, \ldots, n$. Escreva duas versões, uma iterativa e uma recursiva. (Sugestão: O conjunto das permutações dos inteiros $1, \ldots, n$ pode ser obtida através do conjunto das permutações dos inteiros de $1, \ldots, n-1$ inserindo-se n em cada possível posição de cada permutação.)

13. Escreva uma função que dados dois inteiros positivos n e k imprima todas as combinações de $1, \ldots, n$ em grupos de tamanho k. Escreva duas versões, uma iterativa e outra recursiva.

14. Escreva um programa para imprimir em ordem lexicográfica todos os subconjuntos do conjunto $\{1, \ldots, n\}$. Para $n = 4$ o resultado deve ser:

$$\emptyset\ \{1\}\ \{1,2\}\ \{1,2,3\}\ \{1,2,3,4\}\ \{1,2,4\}\ \{1,3\}\ \{1,3,4\}$$
$$\{1,4\}\ \{2\}\ \{2,3\}\ \{2,3,4\}\ \{2,4\}\ \{3\}\ \{3,4\}\ \{4\}.$$

15. A função abaixo calcula o maior divisor comum dos inteiros positivos m e n. Escreva uma função recursiva equivalente.

```
int Euclides (int m, int n)
{
  int r;
  do{
     r = m % n;
     m = n;
     n = r;
  }
  while (r != 0);
  return(m);
}
```