Lista de exercícios

Recursão

Q1 Através do algoritmo abaixo, calcule o fibonacci de 8. Neste procedimento, verifique quantas vezes o fibonacci(4) foi calculado.

```
1  int fibonacci(int n) {
2      if (n <= 1) {
3          return n;
4      }
5      else {
6          return fibonacci(n - 1) + fibonacci(n - 2);
7      }
8  }
```

Q2 Generalize a operação da questão **Q1** anterior considerando que estou calculando o fibonacci de qualquer valor maior do que 4.

Q3 Generalize a questão **Q2** considerando que estou calculando quantas vezes o fibonacci(n) é calculado para encontrar o fibonacci(m), sendo $m > n$.

Q4 No algoritmo abaixo, considere os seguintes tempos:

- Chamada recursiva demora $2ns$
- Retorno da chamada recursiva demora $1ns$;
- Atribuição e soma demora $0.5ns$;
- Divisão e multiplicação demora $1.5ns$

```
1  int funcRecursiva(int n) {
2      if (n == 0) {
3          return 1;
4      }
5      return funcRecursiva(n-1) + 1/funcRecursiva(n-1);
6  }
```

Calcule o tempo total para funcRecursiva(5).

Q5 Refaça o procedimento da questão **Q4** considerando uma modificação no algoritmo:

```
1  int funcRecursiva(int n) {
2      if (n == 0) {
3          return 1;
4      }
5      k = funcRecursiva(n-1);
6      return k + 1/k;
7  }
```

Q6 Escreva, passo a passo, a **execução** do algoritmo fatorial em seu formato recursivo. Evidencie as chamadas recursiva e retornos da recursão.

Q7 No calendário gregoriano, **geralmente** um ano X é bissexto se o ano (x-4) também foi. Este pode ser uma etapa para calcular o algoritmo recursivo para um ano bissexto, mas não está correto por completo. Explique o porquê e apresente uma solução.

Q8 Um fractal é uma estrutura geométrica complexa que exibe repetição infinita de padrões semelhantes em diferentes escalas. Um exemplo de fractal simples é o triângulo de Sierpinski. Ele começa com um triângulo equilátero grande. Em seguida, cada lado desse triângulo é dividido em três partes iguais e o triângulo central é removido. Esse processo é repetido para os triângulos restantes, criando uma sequência infinita de triângulos menores que se assemelham ao triângulo original. Desenhe um triângulo de Sierpinski tal qual descrito no texto.

Q9 Desenhe um triângulo Sierpinski (descrito na questão **Q8**) computacionalmente através do seguinte algoritmo:

(a) Marque os pontos: $A = (0, 1); B = (-1, -1); C = (1, -1)$.

(b) Escolha um ponto aleatório P que esteja no interior do triângulo formado por pelos pontos A, B e C.

(c) Marque o ponto médio entre P e um ponto escolhido aleatoriamente entre A, B e C.

Q10 Apresente um algoritmo de recursão com e sem cauda executa a seguinte expressão:

$$p(x, n) = \prod_{k=0}^{n} (x - k)$$

Q11 Apresente versões recursivas de cauda para cada uma das expressões abaixo:

(a) $f(n) = n!$

(b) $f(n) = 2f(n-1) + 3f(n-2), f(0) = 1, f(1) = 2$

(c) $\sum_{k=1}^{M} k$

Q12 Calcule o sin(80) considerando como caso base o resultado que $\sin(x) = x - \frac{x^3}{6}$ e que:

$$\begin{cases} \sin(x) & = \sin\left(\frac{x}{3}\right)\left(\frac{3-\tan^2\left(\frac{x}{3}\right)}{1+\tan^2\left(\frac{x}{3}\right)}\right) \\ \tan(x) & = \frac{\sin(x)}{\cos(x)} \\ \cos(x) & = 1 - \sin\left(\frac{x}{2}\right) \end{cases}$$

Q13 Implemente uma versão recursiva dos algoritmos abaixo:

(a) Somas sucessivas para calcular o produto de dois números.

(b) Divisão inteira entre dois números através de substrações sucessivas.

(c) Verificação se uma palavra é um palíndromo.

(d) Inversão de uma string.

(e) Geração de todos os números da megasena (6 números entre 1 e 60).

Complexidade

Q14 Explique as seguintes afirmações e questionamentos:

(a) A função f(n) tem complexidade O(n^2).

(b) O tempo necessário para execução do algoritmo tem complexidade $\Theta(n \log n)$

(c) Qual o algoritmo mais veloz, o que tem complexidade O(n) ou um outro com $\Theta(n^2)$?

Q15 Quais as complexidades de tempo dos algoritmos abaixo (*big O*):

(a) A_1: Ordenação de uma lista sequencial não ordenada;

(b) A_2: Busca de um elemento em uma pilha formado por lista encadeada.

(c) A_3: Busca de elementos em uma lista linear encadeada ordenada;

(d) A_4: Inserção de elemento numa fila formado por lista encadeada;

(e) Remoção de elemento em uma lista sequencial;

Q16 Quais as complexidades de memória dos algoritmos abaixo (*big O*):

(a) Inserção de elementos em uma pilha;

(b) Inserção de elementos em uma fila;

(c) Remoção de elementos em uma pilha;

(d) Remoção de elementos em uma fila;

Q17 Prove se é verdadeiro ou falso:

(a) n^2 é $O(n^3)$;

(b) n^3 é $O(n^2)$;

(c) $log_{10}(n^2)$ é $O(\lg(n))$

(d) $n^2 \sin^2(n)$ é $O(n^2)$

Q18 O algoritmo A possui complexidade $O(n^5)$ e o algoritmo B possui complexidade $O(1.5^n)$. Ambos realizam a mesma operação. Qual dos dois você utilizaria?

Q19 A quantidade de operações de um algoritmo A é de $T_A(n) = 2n^2+5$, do algoritmo B é $T_B(n) = 100n$. Até qual tamanho de problema o algoritmo A é mais eficiente do que o B?

Q20 Calcule a complexidade do algoritmo abaixo:

```
1 int f(int n) {
2     int s = 0;
3     for(int i=0; i<n; i++)
4         for(int k=n; k<i; k++)
5             s = s + i;
6 }
```

Q21 Calcule a complexidade do da função main:

```
1  int f(int n) {
2      int s = 0;
3      for(int i=0; i<n*n; i++)
4          s = s + i;
5  }
6  int g(int n){
7      f(n/2) * f(n/2);
8  }
9  int main(){
10     int d;
11     scanf("%d\n", d);
12     g(d);
13 }
```

Q22 Prove se é verdadeiro ou falso:

(a) n^2 é $\Theta(n^3)$;

(b) n^3 é $\Theta(n^2)$;

(c) $log_{10}(n^2)$ é $\Theta(\lg(n))$

(d) $n^2 \cos^2(n)$ é $\Theta(n^2)$

Q23 Prove se é verdadeiro ou falso:

(a) n^2 é $\Omega(n^3)$;

(b) n^3 é $\Omega(n^2)$;

(c) $log_{10}(n^2)$ é $\Omega(\lg(n))$

(d) $n^2 \cos^2(n)$ é $\Omega(n^2)$

Q24 Julgue as afirmações em (V)erdadeiro ou (F)also:

(a) $c_1 O(f(n)) = O(c_1 * f(n))$

(b) $O(f(n) + g(n)) = O(f(n) * g(n))$

(c) $O(f(n)) + O(g(n)) = O(max(f(n), g(n)))$

(d) $f(n)O(g(n)) = O(g(n))$

Conceitos iniciais de Árvores

Q25 Construa uma árvore binária qualquer com os elementos:

49 50 44 54 12 61 68 87 59 30 42 51 33 41 27

Q26 Construa uma árvore estritamente binária com os elementos:

28 99 78 96 64 63 51 86 43 76 86 76 12 18 89

Q27 Construa uma árvore binária cheia com os elementos:

61 66 83 39 78 18 95 19 63 45 44 55 47 45 86

Q28 Construa uma árvore binária completa com os elementos:

84 96 36 30 78 49 77 55 79 76 74 54 67 98 10

Q29 Construa uma árvore binária de busca com os elementos:

91 32 56 90 63 49 20 62 47 87 16 56 35 32 90

Q30 Construa uma árvore binária de busca Zigue-Zague com os elementos:

19 22 48 46 24 39 80 41 62 73 75 32 46 79 24

Utilizem das questões **Q25** a **Q30** para responder as questões **Q31** a **Q36**:

Q31 Calcule os sucessores e antecessores dos nós raízes.

Q32 Localize o nível de todos os nós.

Q33 Localize a altura de todos os nós.

Q34 Realize o percurso em pré-ordem.

Q35 Realize o percurso em pos-ordem.

Q36 Realize o percurso em in-ordem.

Q37 Julgue (V)erdadeiro ou (F)also:

(a) Toda árvore binária cheia é completa.

(b) Toda árvore binária de busca Zigue-Zague possui como raiz o menor elemento.

(c) É possível uma árvore ser simultaneamente: Estitamente binária, Cheia, Completa e Zigue-Zague.

(d) É possível uma árvore ser simultaneamente: Estitamente binária, Cheia, Completa. Mas ser também Zigue-Zague é impossível.

(e) Se o antecessor e sucessor de um nó são irmãos, então o nó é pai dos dois.

(f) O tio de um nó é ancestral a ele.

(g) O neto de um nó é um descendente dele.

Q38 Qual a altura de uma árvore cheia que possui N=324 nós?

Q39 Quantos nós faltam para uma árvore cheia com N=348 nós se torne uma árvore completa?

Q40 Quantos nós precisariam ser removidos para que uma árvore cheia com N=538 nós se torne uma árvore completa?

Q41 Qual a maior altura do nó raiz em uma árvore estritamente binária com N=251 nós.