ANÁLISE E PROJETO DE ALGORITMOS – LISTA 1

Prof. Thiago Cavalcante

1. Prove por indução que $n^3 + 2n$ é divisível por 3 para todo $n \geq 0$ e que $13^n - 6^n$ é divisível por 7 para todo $n \geq 1$. *Lembre-se que para aplicar o método da indução, você: 1.* ***mostra*** *que a afirmação é verdadeira para o menor número considerado no intervalo (nesse caso, $n = 0$ e $n = 1$, respectivamente); 2.* ***assume*** *que a afirmação é verdadeira para todos os valores menores ou iguais a um número qualquer k; 3. mostra, usando a suposição do passo anterior, que a afirmação* ***continua valendo*** *para $k + 1$.*

2. Prove por indução que a soma dos cubos dos primeiros n inteiros positivos é igual ao quadrado da soma desses inteiros, ou seja,

$$\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$$

   Lembre-se que

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

3. Um algoritmo de ordenação leva 1 segundo para ordenar $n_1 = 1.000$ itens no seu computador. Quanto tempo ele vai levar para ordenar $n_2 = 10.000$ itens se o algoritmo tem um tempo de execução proporcional a:

   (a) n^2?

   (b) $n \log n$?

4. Para cada par de funções $f(n)$ e $g(n)$ a seguir, determine se $f(n) = O(g(n))$, $g(n) = O(f(n))$, ou ambos.

   (a) $f(n) = (n^2 - n)/2$, $g(n) = 6n$

   (b) $f(n) = n + 2\sqrt{n}$, $g(n) = n^2$

   (c) $f(n) = n \log n$, $g(n) = n\sqrt{n}/2$

   (d) $f(n) = n + \log n$, $g(n) = \sqrt{n}$

   (e) $f(n) = 2(\log n)^2$, $g(n) = \log n + 1$

   (f) $f(n) = 4n \log n + n$, $g(n) = (n^2 - n)/2$

   Lembre-se que os valores constantes podem ser ignorados e que existe uma ***relação de dominância*** *entre as funções:*

$$n! \gg c^n \gg n^3 \gg n^2 \gg n^{1+\epsilon} \gg n \log n \gg$$
$$n \gg \sqrt{n} \gg (\log n)^2 \gg \log n \gg 1$$

   onde $f(n) \gg g(n)$ é o mesmo que $g(n) = O(f(n))$ (f domina g).

5. Prove que se $f_1(n) = \Omega(g_1(n))$ e $f_2(n) = \Omega(g_2(n))$, então $f_1(n) + f_2(n) = \Omega(g_1(n) + g_2(n))$.

6. Considere o código abaixo:

```
for i = 1 to n do
  for j = i to 2 * i do
    print "ufal"
```

   Expresse a quantidade de vezes que a palavra **ufal** é impressa na tela como um somatório e simplifique até chegar a uma fórmula. *Use a fórmula da questão 2 para conseguir finalizar essa questão.*

7. Você tem acesso a um dicionário balanceado que realiza as operações de busca, inserção, remoção, mínimo, máximo, sucessor e antecessor em um tempo $O(\log n)$. Explique como você pode modificar as operações de **inserção** e **remoção**, de forma que elas continuem com o tempo logarítmico, mas que o **máximo** e **mínimo** agora sejam $O(1)$. *Dica: você pode armazenar os valores máximo e mínimo em variáveis auxiliares que estão sempre à disposição sem a necessidade de uma busca (portanto, $O(1)$). No entanto, o que deve ser feito nas funções de inserção e remoção para garantir que esses valores estarão sempre atualizados? Pense em termos das operações do dicionário e não em termos de programação.*

8. Dados dois conjuntos S_1 e S_2, com n elementos cada, e um número x, descreva um algoritmo $O(n \log n)$ para encontrar um par de elementos, um de S_1 e um de S_2, cuja soma é igual a x. *Lembre-se dos algoritmos de busca e ordenação e de seus respectivos tempos de execução.*

9. Dado um conjunto S com n números reais e um número real x, descreva um algoritmo que seja capaz de determinar se existem dois números em S cuja soma é x.

   (a) Assuma que S não está ordenado e dê um algoritmo de tempo $O(n \log n)$.

   (b) Assuma que S está ordenado e dê um algoritmo $O(n)$.

10. Descreva um algoritmo eficiente para reorganizar um *array* de valores de forma que todos os números negativos precedam os números positivos (não necessariamente em ordem). Você não pode usar um *array* auxiliar para guardar elementos. Qual é o tempo de execução do algoritmo? *Dica: qual é a operação realizada na execução do algoritmo quicksort?*