ACH-2002 - Introdução à Análise de Algoritmos - Turma 94, Prova 1 – 25.10.2024 – Prof. Fábio Nakano

Nome: **nusp**

Orientações

- Duração: 1h30min. Entregar a avaliação ao professor.
- Mostre que você sabe, dê respostas detalhadas
- Escrever a lápis ou tinta, como preferir
- Escrever seu nome e número USP em todas as folhas. Quando houver local indicado, usá-lo.
- Entregar esta folha junto com as folhas de resposta.
- Indicar claramente a que questão e item refere-se a resolução
- Apresentar a resolução na ordem que preferir. (o enunciado deve ser lido sequencialmenete ;-)
- Na entrega, colocar as folhas de resposta e esta uma dentro da outra de forma que formem um único bloco.
- Caso o tempo tenha se esgotado e o professor precise ir ao aluno recolher a avaliação será atribuída nota ZERO. Avaliações que não forem entregues receberão nota ZERO.
- É proibida qualquer consulta, por exemplo (não limitado a) colegas, livros, anotações feitas antes da avaliação e anotações de colegas.
- Mostrar o encadeamento lógico das idéias e conceitos é essencial nas respostas.

---

1. Demonstre que $f(n)$ {é/não é} $O(g(n))$ (use a aproximação de Stirling, nota: independente do contexto em que foi apresentada, é uma função como muitas outras) **2,5pt**

   $f(n) = log(n!)$, $g(n) = n * log(n)$

2. Considere, no quicksort, casos em que o particionamento é proporcional mas assimétrico em que, sistematicamente, 1/5 do array fica em uma partição e os outros 4/5 do array ficam em outra partição. Obtenha a função de complexidade de tempo para esse caso e justifique **1,5pt**; aplique o Teorema Mestre (mostre que você testou os casos, diga qual a função de complexidade obtida pela aplicação do Teorema Mestre) **1,0pt**

```
QUICKSORT(A, p, r)
if p < r
    q = PARTITION(A, p, r)
    QUICKSORT(A, p, q − 1)
    QUICKSORT(A, q + 1, r)
```

3. Considere $f(n) = n^2$ e $g(n) = n^3$. É possivel afirmar que $f(n) = O(g(n))$? Demonstre **0,5pt**. É possível afirmar que $f(n) = o(g(n))$? Demonstre **0,5pt**. É possível afirmar que $f(n) = \Theta(g(n))$? Demonstre **0,5pt**. Associe esses resultados com o conceito de *"asymptotically tight bounds"* (limitantes assintoticamente ajustados) **0,5pt**, definido abaixo:

   > Figure 3.1(a) gives an intuitive picture of functions $f(n)$ and $g(n)$, where $f(n) = \Theta(g(n))$. For all values of n at and to the right of n_0, the value of $f(n)$ lies at or above $c_1 g(n)$ and at or below $c_2 g(n)$. In other words, for all $n \geq n_0$, the function $f(n)$ is equal to $g(n)$ to within a constant factor. We say that $g(n)$ is an ***asymptotically tight bound*** for $f(n)$.

   g é um limitante assintoticamente ajustado para f? **0,5pt**

4. Considere a execução de Max-Heapify. Qual a função de complexidade que corresponde à altura da árvore de execução (árvore de chamadas) de Max-Heapify? Justifique. **2,5pt**

```
MAX-HEAPIFY(A, i)
l = LEFT(i)
r = RIGHT(i)
if l ≤ A.heap-size and A[l] > A[i]
    largest = l
else largest = i
if r ≤ A.heap-size and A[r] > A[largest]
    largest = r
if largest ≠ i
    exchange A[i] with A[largest]
    MAX-HEAPIFY(A, largest)
```

Teorema Mestre:
Dada a recorrência $T(n) = aT(\frac{n}{b}) + f(n)$
caso 2: Se $f(n) \in \Theta(n^{log_b(a)}) \Rightarrow T(n) \in \Theta(n^{log_b(a)} * lg(n))$
caso 1: Se $f(n) \in O(n^{log_b(a)-\epsilon}) \Rightarrow T(n) \in \Theta(n^{log_b(a)})$
caso 3: Se $f(n) \in \Omega(n^{log_b(a)+\epsilon})$ ...
e $a * f(\frac{n}{b}) \leq c * f(n); 0 < c < 1 \Rightarrow T(n) \in \Theta(f(n))$
Aproximação de Stirling:

$$n! = \sqrt{2\pi n}(\frac{n}{e})^n(1 + O(\frac{1}{n}))$$

Alguns operadores assintóticos:
$O(g(n)) = \{f(n) : \exists\, c, n_0$, positivas, tais que $0 \leq f(n) \leq c \cdot g(n)\, \forall n \geq n_0\}$
$o(g(n)) = \{f(n) : \exists\, c, n_0$, positivas, tais que $0 < f(n) < c \cdot g(n)\, \forall n \geq n_0\}$
$\Theta(g(n)) = \{f(n) : \exists\, c_1, c_2, n_0$, positivas, tais que $0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)\, \forall n \geq n_0\}$
Escrito usando Overleaf.