Projeto e Análise de Algoritmo
Lista de Exercício 1 -

1 - a) → Rode de $i=0$ a $i=n$:
Receba $A[i] = a_i$;

→ Receba v;

→ Rode de $i=0$ a $i=n$:
Se $A[i] == v$, retorna i;

→ Termino o laço;
Retorno -1;

b) O algoritmo mantém um invariante de laço que verifica os n índices do vetor, começando no primeiro índice e terminando no n-ésimo índice. Se colocarmos o i para rodar $n-1$ índices, o v pode estar na n-ésima posição, comprometendo o algoritmo. Se colocarmos o i para rodar $n+1$ índices, ele irá captar uma componente inexistente do vetor, comprometendo o funcionamento do algoritmo

c) Pelo algoritmo de busca linear, devemos buscar em todos os n elementos, ou seja, checar de 1 por 1 os elementos do determinado vetor, pois todos os elementos de A tem o mesmo probabilidade de serem encontrados pelo algoritmo.

d) Tomando o pseudocódigo acima, temos a função de custo dada por $f(n) = 2n + 2$. O caso médio é dado quando, no segundo laço, há um critério de parada reduzido. Tomando $g(n) = n$, temos, que, para $c_1 = 1$ e $c_2 = 3$, $1 \cdot n \leq 2n+2 \leq 3n$. Como o $n \leq 2n+2$ para qualquer n_0, ou seja $f(n) \in \Omega(n)$. Como $3n \geq 2n+2$, para qualquer $n_0 \geq 2$, ou seja $f(n) \in O(n)$. Como $f(n) \in \Omega(n)$ e $f(n) \in O(n)$, $f(n) \in \Theta(n)$

2- for $i=1$ to n do
  for $j=i$ to $2i$ do
    print "hello"

a) Como possuímos laços encadeados, teremos que:
$f(n) = \sum_{i=1}^{n} \sum_{j=i}^{2i} 1$, e esta é a forma da função complexidade em somatórios

b) Simplificando, teremos:
$f(n) = \sum_{i=1}^{n} \sum_{j=i}^{2i} 1$, como $i \to 2i$, temos uma quantidade i de termos a serem somados, ou seja, $f(n) = \sum_{i=1}^{n} \underbrace{(1+\dots+1)}_{i \text{ vezes}} = \sum_{i=1}^{n} i$. Aplicando a forma de Gauss da soma de PA, temos:

$f(n) = \sum_{i=1}^{n} i = \frac{n(n+1)}{2} = \frac{n^2+n}{2}$, que é a complexidade do algoritmo.

3- a) $n^3 - 3n^2 - n + 1 = \Theta(n^3)$ (Verdadeiro).
Analisando $c_1 n^3 \leq n^3 - 3n^2 - n + 1 \leq c_2 n^3$, temos:
$c_1 \leq 1 - \frac{3n^2}{n^3} - \frac{n}{n^3} + \frac{1}{n^3} \leq c_2 \Rightarrow \lim_{n \to \infty} c_1 \leq \lim_{n \to \infty} 1 - \frac{3}{n} - \frac{1}{n^2} + \frac{1}{n^3} \leq c_3$

$\Rightarrow c_1 \leq 1 \leq c_2$, tomando $c_1 = 1$ e $c_2 \geqslant 1$, temos que vale a afirmação.

b) $2^{2n} = O(2^n)$ (Falso).

Analisando $2^{2n} \leq c2^n$, teremos:

$$2^{2n} \leq c2^n \Rightarrow \frac{2^{2n}}{2^n} \leq \frac{c2^n}{2^n} \Rightarrow 2^n \leq c,$$ e não existe um c natural que satisfaça a condição.

c) $\log n^2 = \Omega(\log n)$ (Verdadeiro).

Analisando $\log(n^2) \geqslant c \log(n)$

$$\log(n^2) \geqslant c\log(n) \Rightarrow \frac{2\log(n)}{\log(n)} \geqslant \frac{c\log(n)}{\log(n)} \Rightarrow 2 \geqslant c,$$

portanto existe natural c que satisfaça.

d) $\sqrt{n} = O(\log(n))$ (Falso).

Analisando $\sqrt{n} \leq c\log(n)$, teremos:

$$\sqrt{n} \leq c\log(n) \Rightarrow \frac{\sqrt{n}}{\log(n)} \leq c\frac{\log(n)}{\log(n)} \Rightarrow \lim_{n\to\infty} \frac{\sqrt{n}}{\log(n)} \leq \lim_{n\to\infty} c,$$

aplicando L'Hospital, temos:

$$\Rightarrow \lim_{n\to\infty} \frac{\frac{1}{2}\cdot\frac{1}{\sqrt{n}}}{\frac{1}{n}} \leq c \Rightarrow \lim_{n\to\infty} \frac{\sqrt{n}}{2} \leq c,$$ e como $\frac{\sqrt{n}}{2} \to \infty$, então a hipótese é falsa.

e) $\log n^2 = \Theta(\log(n)+5)$ (Verdadeiro)

Analisando $c_1(\log(n)+5) \leq \log(n^2) \leq c_2(\log(n)+5)$, temos,

$c_1(\log(n)+5) \leq 2\log(n) \leq c_2(\log(n)+5) \Rightarrow$

$$\Rightarrow \lim_{n\to\infty} c_1\left(\frac{\log(n)}{\log(n)} + \frac{5}{\log(n)}\right) \leq \lim_{n\to\infty} \frac{2\log(n)}{\log(n)} \leq \lim_{n\to\infty} c_2\left(\frac{\log(n)}{\log(n)} + \frac{5}{\log(n)}\right)$$

$\Rightarrow c_1 \cdot 1 \leq 2 \leq c_2$, tome $c_1 = 1$ e $c_2 = 3$ e a sentença é verdadeira.

4-a) O algoritmo calcula, seja b a quantidade de divisões que b faz por 2, $2 \cdot n \cdot a$ na pior hipótese e caso b seja ímpar, vai ser $\sum_{i=1}^{n} a_i$, sendo $a_i = 2 \cdot a_{i-1}$.

b) A invariante de laço que o algoritmo mantém é dado pela quantidade sucessiva de vezes que o b é dividido por 2 até chegar 1 mais 1 vez para $b=0$.

c) Para b par, ele irá rodar até 0 e retornará o valor de x, se b for ímpar, ele também irá rodar até 0, retornando o valor de x de acordo com as operações, ou seja, o algoritmo está correto de acordo com sua proposta.

5- A função retorna, para o valor de r em função de n:

$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \sum_{k=1}^{j} 1 = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} j$, iremos calcular $\sum_{j=i+1}^{n} j$, portanto, para $n=5$, temos:

$i=1 \Rightarrow 2+3+4+5=14$

$i=2 \Rightarrow 3+4+5=12$, ou seja,

$i=3 \Rightarrow 4+5=9$

$$S_n=\frac{((i+1)+n)(n-i)}{2}$$

$$S_n=\frac{\cancel{ni}+n-i^2-i+n^2-\cancel{ni}}{2}$$

$$S_n=\frac{n^2+n-i^2-i}{2}$$, voltando no

somatório, temos:

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

$$\sum_{i=1}^{n-1}\frac{-i}{2}+\sum_{i=1}^{n-1}\frac{-i^2}{2}$$, calculando $\sum_{i=1}^{n} i = 1+2+3+\dots+n-1+n=\frac{n(n+1)}{2}$,

portanto $\sum_{i=1}^{n-1} i=\frac{n(n+1)}{2}-n=\frac{n^2-n}{2}=\frac{n(n-1)}{2}$. Agora no outro

somatório, temos:

$$\sum_{i=1}^{n} i^2=\frac{n\cdot(n+1)(2n+1)}{6}$$, ou seja, $\sum_{i=1}^{n-1} i^2=\frac{n(n+1)(2n+1)}{6}-n^2$

$$=\frac{2n^3+n^2+2n^2+n-6n^2}{6}=\frac{n(2n^2-3n+1)}{6}=\sum_{i=1}^{n-1} i^2$$, ou seja,

voltando no somatório original,

$$\frac{n(n-1)}{2}+\frac{n^2(n-1)}{2}-\frac{n(n-1)}{4}-\frac{n(2n^2-3n+1)}{12}=$$

$$=\frac{6n^2-6n+6n^3-6n^2-3n^2+3n-2n^3+3n^2-n}{12}=\frac{4n^3-4n}{12}$$ //