Primeira Avaliação PAA

João Pedro Pires Bertonedo
RA 112.650

1 - a) $1^3 + 2^3 + 3^3 + \dots + n^3 = (n^2/4) \cdot (n+1)^2 \quad \forall n \geq 1$

Base: $1^3 = (1^2/4) \cdot (1+1)^2$

$1 = 1$ ✓

Hipótese: Fórmula para $n = k$

Tese: $1^3 + 2^3 + 3^3 + \dots + k^3 = (k^2/4) \cdot (k+1)^2$

$$\underbrace{1^3 + 2^3 + 3^3 + \dots + k^3} + (k+1)^3$$

$$(k^2/4) \cdot (k+1)^2 + (k+1)^3 = \frac{k^2}{4} \cdot (k+1)^5$$

Igualando, temos:

$$\frac{k^2}{4} \cdot (k+1)^2 + (k+1)^3 = ((k+1)^2/4) \cdot ((k+1)+1)^2$$

$$k^2 \cdot (k+1)^2 + 4(k+1)^3 = (k+1)^2 \cdot (k+2)^2$$

$$k^2 \cdot (k^2 + 2k + 1) + 4 \cdot (k^3 + 3k^2 + 3k + 1) = (k^2 + 2k + 1) \cdot (k^2 + 4k + 4)$$

$$k^4 + 2k^3 + k^2 + 4k^3 + 12k^2 + 12k + 4 = k^4 + 4k^3 + 4k^2 + 2k^3 + 8k^2 + 8k + k^2 + 4k + 4$$

$$k^4 + 6k^3 + 13k^2 + 12k + 4 = k^4 + 6k^3 + 13k^2 + 12k + 4$$

$$0 = 0$$

b) base: para $n = 1$

$$1 = \frac{1}{2} \cdot (3 - 1)$$

$1 = 1$ ✓

hipótese: fórmula para $n = k$

tese: $1 + 4 + 7 + \ldots + (3k - 2) = \frac{k}{2} \cdot (3 \cdot k - 1) = \frac{(3k^2 - k)}{2}$

$$\underbrace{1 + 4 + 7 + \ldots + (3k - 2)} + (3k + 1)$$

$$\frac{k(3k - 1)}{2} + (3k + 1) = \frac{(3k^2 - k)}{2} + (3k + 1)$$

c) base: para $n = 1$

$1^3 = (1)$

$1 = 1$

hipótese: fórmula para $n = k$

tese: $1^3 + 3^3 + 5^3 + \ldots + (2k - 1)^3 = k^2 (2k^2 - 1)$

$$\underbrace{1^3 + 3^3 + 5^3 + \ldots + (2k - 1)^3} + (2k + 1)^3$$

$$k^2 \cdot (2k^2 - 1) + (2k + 1)^3 = 2k^4 - k^2 + (2k + 1)$$

=

Portanto, temos:

$$k^2 \cdot (2k^2 - 1) + (2k+1)^2 = (k+1)^2 (2(k+1)^2 - 1)$$

$$2k^4 - k^2 + (2k^3 + 3 \cdot 2k^2 + 3 \cdot 2k + 1) = (k+1)^2 (2(k^2+2k+1) - 1)$$

$$2k^4 + k^2 + 8k^3 + 12k^2 + 6k + 1 = (k^2 + 2k + 1)(2k^2 + 4k + 1)$$

$$2k^4 + 8k^3 + 11k^2 + 6k + 1 = 2k^4 + 4k^3 + k^2 + 4k^3 + 8k^2 + 2k + 2k^2 + 4k + 1$$

$$2k^4 + 8k^3 + 11k^2 + 6k + 1 = 2k^4 + 8k^3 + 11k^2 + 6k + 1$$

d) base: para $n = 1$

$$1^2 = \frac{(2-1) \cdot (2+1)}{3}$$

$$1 = 1$$

hipótese: Iremos para $n = k$

tese: $1^2 + 3^2 + 5^2 + \ldots + (2k-1)^2 = \frac{k(2k-1)(2k+1)}{3}$

$$\underbrace{1^2 + 3^2 + 5^2 + \ldots + (2k-1)^2}_{} + (2k+1)^2$$

$$\frac{k(2k-1)(2k+1)}{3} + (2k+1)^2$$

portanto, temos:

$$\frac{k(2k-1)(2k+1)}{3} + (2k+1)^2 = \frac{(k+1)((k+1)-1)(2(k+1)+1)}{3}$$

$$(2k^2 - k)(2k+1) + 3(2k+1)^2 = (k+1)(2k+1)(2k+3)$$

$$4k^2 - k + 3(4k^2 + 2 \cdot 2k + 1) = (2k^2 + 3k + 1)(2k+3)$$

$$4k^3 + 12k^2 + 11k + 3 = 4k^3 + 6k^2 + 2k + 6k^2 + 9k + 3$$

$$4k^3 + 12k^2 + 11k + 3 = 4k^3 + 12k^2 + 11k + 3$$

e) base: para $n = 4$

$2^4 > 4^2$

$16 > 16$

não é verdadeiro, portanto, não há como provar.

f) base: para $n = 7$

$7! > 3^7$

$5.040 > 2.187$

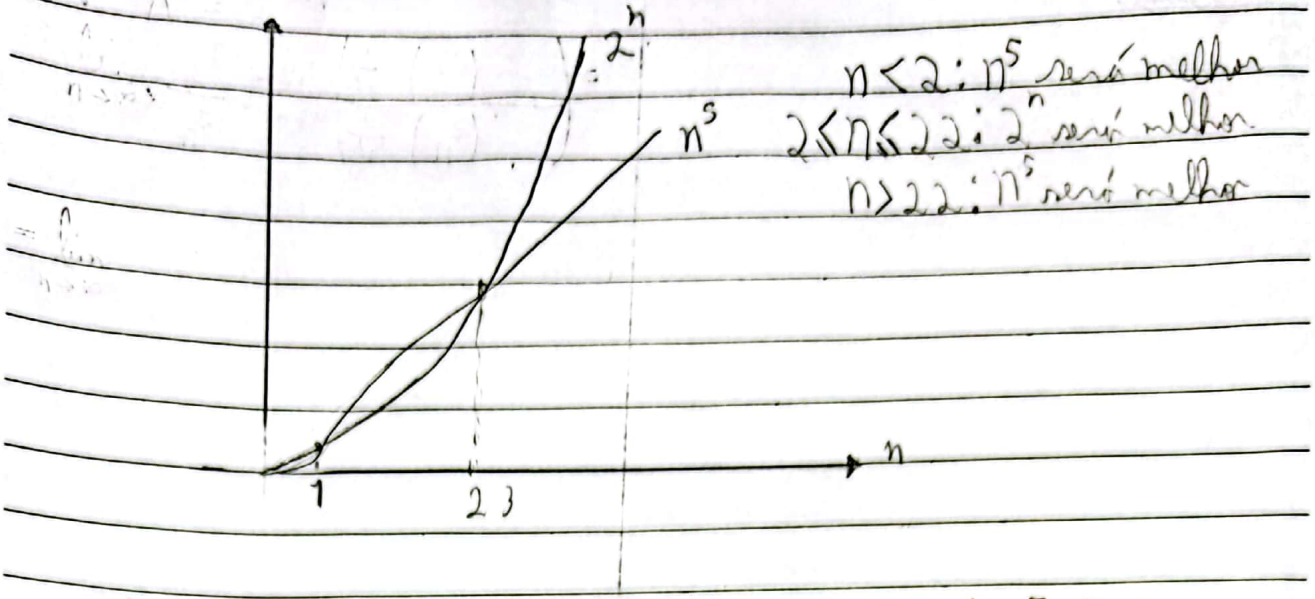
hipótese: função para $n = K$

tese: $K! > 3^K$

$(K+1)! > 3^{(K+1)}$

portanto, temos que $K! < (K+1)!$

4 - Temos, portanto, que para $n<2$ e $n>22$, n^5 terá um melhor desempenho, e para $2 \leq n \leq 22$, 2^n terá um melhor desempenho portanto:



$n<2$: n^5 será melhor
$2 \leq n \leq 22$: 2^n será melhor
$n>22$: n^5 será melhor

temos, como prova, a seguinte tabela de testes:

	2^n	n^5
20	1.048.576	3.200.000
21	2.097.152	4.084.101
22	4.194.304	5.153.632
23	8.388.608	6.436.343
24	16.777.216	7.962.624

5 - A notação $O(g(n))$ indica que a função $f(n)$ nunca será pior que a função $g(n)$ em determinado ponto indicado.

A notação $\Omega(g(n))$ indica que a função $f(n)$ nunca será pior que $f(n)$ em determinado ponto indicado.

(a) A partir do ponto n_0, a função $f(n)$ é $O(C_2 g(n))$ e $\Omega(C_1 g(n))$.

(b) A partir do ponto n_0, a função $f(n)$ é $O(C g(n))$

(c) A partir do ponto n_0, a função $f(n)$ é $\Omega(C g(n))$

3) ITERATIVO

```
#include <STDIO.H>
int iterativo (int X[][], int m, int n){
    int i=0, j=0, simetrico;
    for(i=0; i<n; i++){

        for (j=0; j<m; j++){
            if (X[i][j] == X[j][i]){
                simetrico = 1;
            }else{
                simetrico = 0;
            }
        }
    }
Return simetrico
```

Recursivo

```
#include <STDio.h>
int recursivo (int X[][], int i, int j, int m, int n){
if (X[i][j] != X[j][i]){
    Return 0;} else{
    if (i < n){
        if (X[i][j] == X[j][i]){
            return recursivo (X, i++, j, m, n);
        } else{
            return 0;
        }
    } else if (j < m){
        if (X[i][j] == X[j][i]){
            return recursivo (X, i, j++, m, n);
    } } else{
            return 0;
```

3)...

}

}

return 1;

a) iterativo:

melhor caso: quando uma matriz for 1x1, e será O(1).

caso médio: quando a matriz tiver qualquer tamanho de n, e será $O(n^2)$

pior caso: Quando a matriz for infinito, será $O(n^2)$

Recursivo:

Melhor caso: quando a matriz for 1X1, o algoritmo será O(1).

caso médio: com valores de n quaisquer, teremos um algoritmo $O(n^2)$

pior caso: quando a matriz for infinito, teremos $O(n^2)$

b) V