

# List of Beatriz Saife

01)  $f(n) = 3n + 6$

```
int main() {  
    int n, f; → 2  
    scanf("%d", &n); → 1  
    while (n > 0) { → 1+n  
        f = f * n; → n  
        n--; → n  
    }  
    printf("%d\n", f); → 1  
    return 0; → 2  
}
```

02)  $f(n) = 3n + 8$

```
int main() {  
    int x = 2, n = 5, p = 1; → 6  
    while (n != 0) { → 1+n  
        p = p * x; → n  
        n--; → n  
    }  
    printf("%d\n", p); → 1  
}
```

$$3) f(n) = 3n + 9$$

```

Int main(){
    Int n, x=4; → ⊥
    Int A[7] = {2,5,8,4,6,7}; → ⊥
    n = sizeof(A) / sizeof(int); → ⊥
    While (n != 0){ → ⊥ + n
        If (A[n] == x){ → n
            printf("Encontrado %d\n", A[n]); → ⊥
        }
        n--; → n
    }
    return 0; → ⊥
}

```

$$4) f(n) = n \cdot m + 4m + 13$$

```

Int main(){
    Int n=1, m=1, i=0, j; → ⊤
    Scanf("%d", &m); → ⊥
    Scanf("%d", &n); → ⊥
    Int A[m][n]; → ⊥
    While (i < m){ → ⊥ + m
        While (j < n){ → m + n \cdot m
            A[i][j] = 0; → n \cdot m
            j++; → n \cdot n
        }
        i++; → m
    }
    j = 0; → m
    return 0; → ⊥
}

```

# Listão 02

Beatriz Ta. de

O) A.  $7n^2 \in O(n)$

falso, pois não é possível encontrar uma constante,  $7n^2 \leq C \cdot n$

B.  $7n^2 \in O(n^2)$

verdadeiro, pois  $7n^2$  satisfaç  $O(n^2)$

C.  $7n^2 \in O(n^3)$

verdadeiro, satisfaç  $O(n^3) = 7n^2 \leq n^3 \cdot C$

d.  $7n^2 \in \Omega(n)$

verdadeiro, onde satisfaç a definição

$$\Omega(n^2) = 7n^2 \geq n \cdot C$$

e.  $7n^2 \in \Omega(n^2)$

verdadeiro, satisfaçendo  $\Omega(n^2) = 7n^2 \geq n^2 \cdot C$

f.  $7n^2 \in \Omega(n^3)$

falso, pois não encontra uma constante

$$\Omega(n^3) = 7n^2 \geq n^3 \cdot C$$

g.  $7n^2 \in \Theta(n)$

falso, pois não encontra  $C_1$  e  $C_2$  que

satisfazem a definição  $\Theta(n)$

$$C_1 \cdot n \leq 7n^2 \leq C_2 \cdot n$$

$$h \circ 7n^2 \in \Theta(n^2)$$

verdadeiro, satisfazendo  $\Theta(n^2) = C_1 \cdot n^2 \leq 7n^2 \leq C_2 \cdot n^2$

$$I \circ 7n^2 \in \Theta(n^3)$$

falso, pois não encontro a constante que satisfaça as constantes  $C_1, C_2$  de  $\Theta(n^3)$

$$C_1 \cdot n^3 \leq 7n^2 \leq C_2 \cdot n^3$$

$$2) f(n) = 12n^3 + 7n + 25 \in O(n^3) ?$$

$$12n^3 + 7n + 25 \leq n^3 \cdot C$$

$$12n^3 \leq C \cdot n^3 \quad \rightarrow \quad C = 13 / n = 1$$

$$12n^3 \leq 13n^3$$

pois,  $f(n)$  pertence a  $O(n^3)$

$$3) f(n) = 5n^2 + 2n \quad / \quad g(n) = n^3 + 14 \in \omega(g(n)) ?$$

$$5n^2 + 2n \geq n^3 + 14 \cdot C$$

$$5n^2 \geq C \cdot n^3$$

não, pois não encontro um  $C$  que satisfaça  $\omega(g(n))$

$$4) f(n) = 5^{n+1} \quad / \quad g(n) = 5^n \in O(g(n)) ?$$

$$5^{n+1} \leq 5^n \cdot C$$

$$5^n \cdot 5 \leq C \cdot 5^n$$

$$C = 5^1$$

pois,  $f(n)$  é limitado superiormente por  $g(n)$  pois foi encontrado uma constante.

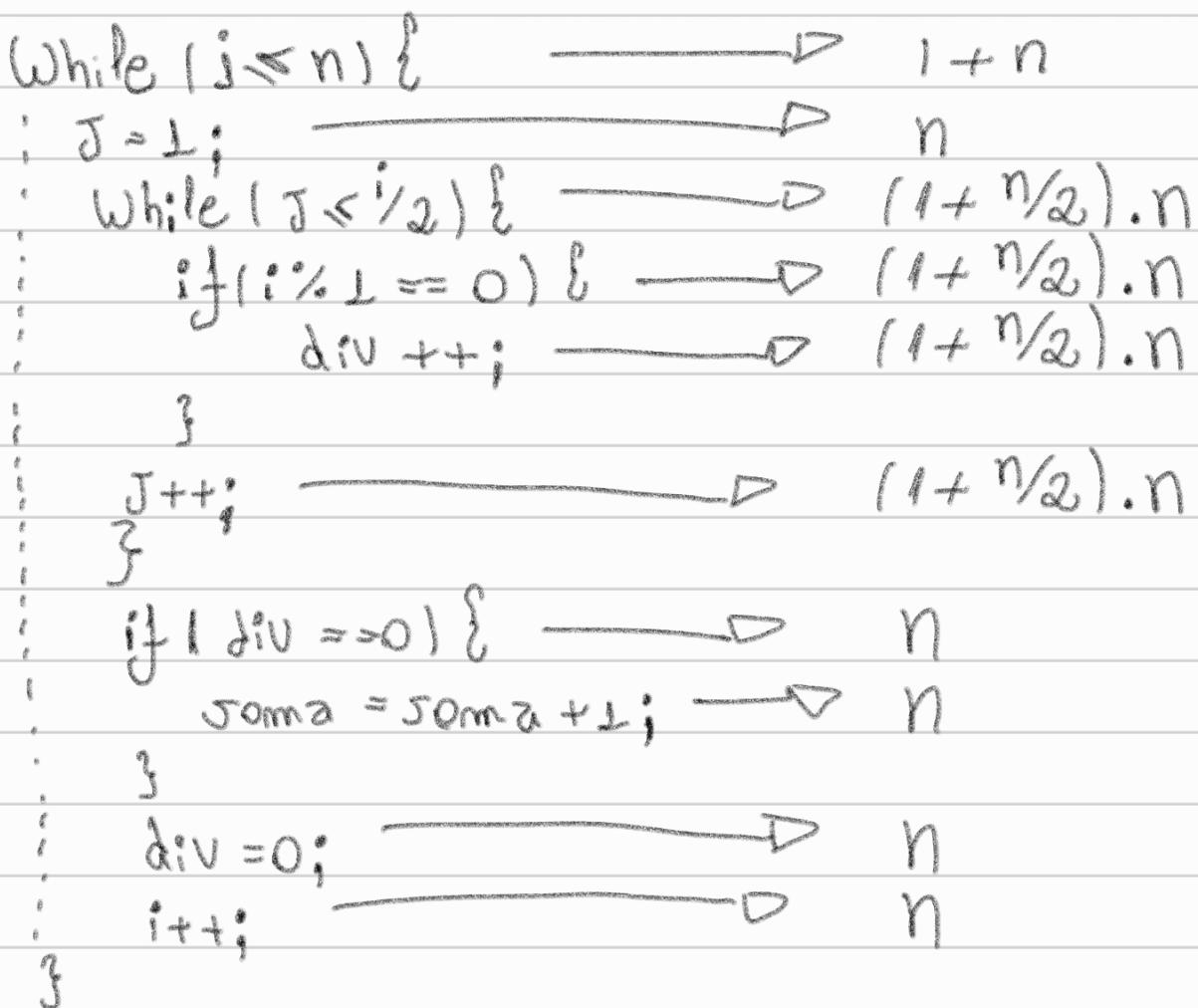
$$5) f(n) = 5^{2^n} / g(n) = 5^n \in O(g(n))?$$

$$5^{2^n} \leq C \cdot 5^n$$

$$5^n \cdot 5^n \leq C \cdot 5^n$$

falso, pois para ser valido a função tem que ser maior que  $5^n$ .

$$6) f(n) \in \Theta(n^2)?$$



$$f(n) = 1 + n + n + 4((1 + n/2).n) + 4n$$

$$6n + 1 + 4n(1 + n/2)$$

$$6n + 1 + 4n + 2n^2$$

$$2n^2 + 10n + 1$$

# Lijda 03

Beatriz Saito Gobira

01) Big O

a)  $f(n) = n^3$

$$n^3 \leq C \cdot n^3$$

$$C = 1$$

$$R = O(n)$$

b)  $f(n) = 133n + 1500$

$$133n + 1500 \leq C \cdot n$$

$$133n + 1500 \leq 133n + 1500n$$

$$133n + 1500 \leq 1633$$

$$C = 1633$$

$$R = O(1)$$

c)  $f(n) = 700$

$$700 \leq C \cdot 1$$

$$C = 700$$

$$R = O(1)$$

02) Omega

a)  $f(n) = 33n^3 + 33n^2 + 33n + 33$

$$33n^3 + 33n^2 + 33n + 33 \geq C \cdot n^3$$

$$C = 1$$

$$R = \Omega(n^3)$$

b)  $f(n) = n + 5000$

$$n + 5000 \geq C \cdot n$$

$$C = 1$$

$$R = \Omega(n)$$

$$C) f(n) = 10^8 n^7 / 7 + 10^6 n + 10^4$$

$$10^8 n^2 / 7 + 10^6 n + 10^4 \gg C \cdot n^2$$

$$C = 1$$

$$R = \alpha(n^2)$$

3) theta:

$$2) f(n) = n^2 + 10^3$$

$$C_1 \cdot n^2 \leq n^2 + b^3 \leq C_2 \cdot n^2$$

$$C_1 > 1$$

$$\leq \eta^2(1+10^3)$$

$$C_2 = 1 + 10^3$$

$$b) f(n) = n^3 + n^2 + n + 1$$

$$C_1 \cdot n^3 \leq n^3 + n^2 + n + 1 \leq C_2 \cdot n^3$$

$$G = \mathbb{Z}/\pi$$

$$\leq n^3 + n^3 + n^3 + n^3$$

$$\sqrt{5} \cdot 4n^3$$

$$\sqrt{C_2} = 4$$

$$c) f(n) = 27n + 27 \cdot 10^{13}$$

$$C_1 \cdot n \leq 27n + 27 \cdot 10^{13} \leq C_2 \cdot n$$

$$C_1 = 1 \text{ K}$$

2

110

$$\leq n(27 + 27 \cdot 10^{13})$$

$$C_2 = 27 + 27 \cdot 10^{13}$$

```

4) for(i=0; i<n; i++) { // 2+2n
    for(j=0; j<n; j++) { // n(2+2n)
        cout << m[i][j]; // n.n
    }
}

```

$$\int |N| \in \Theta(n^2)$$

# Lista 04

## Beatriz Jardó

1) int somar (vet[], n) {  
 if (N == 0) {  
 return 0;  
 } else {  
 return vet[n-1] + somar(vet, N-1);  
 }

Relación  
Referencias:  
 $t(n) = t(N-1) + 1$

2) int palindromo (n, reverso) {  
 if (n == 0) {  
 return reverso;  
 } else {  
 int dig = N % 10;  
 reverso = reverso \* 10 + dig;  
 return palindromo (N/10, reverso);  
 }

Relación Referencias:  $t(n) = t(N/10) + O(1)$

3) int binario (N, b) {  
 if (N == 0) return b;  
 else {  
 int dig = N % 2;  
 b = b \* 10 + dig;  
 return binario (N/2, b)  
 }

Relación de  
Referencias:  
 $t(N) = t(N/2) + O(1)$

4) Relación de Recurrencia:  $t(A, B) = t(A-1, B) + t(A, B-1) + O(1)$

# List 05

## Beatriz Soito

D1) P: algoritmos polinomiais eficientes, como por exemplo algoritmos de ordenação;

Não necessariamente porque um problema X, tem N formas e todas elas são expressões, isso coloca obrigatoriamente  $X \in P$ .

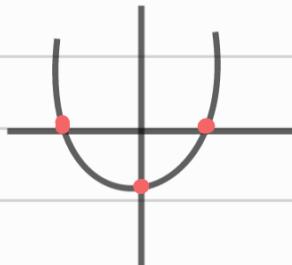


gráfico polinomial

NP: são algoritmos que podem ter resultados por soluções não determinísticas polinomiais.

Como por exemplo, caminho Hamiltoniano de um grafo não orientado

NP-completo: uma classe especial dos NP onde um problema de X se transforma facilmente em um problema de decisão Y.  
Ex: problemas de satisfação booleana

$$(A \vee \neg B) \wedge (\neg A \vee B \vee C) \wedge (C \vee \neg A)$$

O problema validade é j para A, b e c que formam todas expressões verdadeiras.

```

Q2) Int buscaBinaria(int arr[], int l, int r, int x) {
    while (l <= r) {
        int meio = l + (r + 1) / 2;
        if (arr[meio] == x) return meio;
        if (arr[meio] < x) l = meio + 1;
        else r = meio - 1;
    }
    return -1;
}

```

O problema encontra-se na classe P  
 para seu tempo de execução é O(log n)  
 sendo polinomial em relação aos dados  
 de entrada