Trabalho de estrutura de dados

1. Tempo e consumo de memória.

2) Porque o tempo de execução não depende somente do algoritmo.

3) A medição de tempo é feita através da quantidade de passos que o algoritmo precisa para finalizar sua execução. A quantidade de passos é expressa em forma de uma função matemática f(n), onde n representa o tamanho da entrada.

4) Base 2, porque o código é binário.

5) Seja M uma máquina de turing determinística que pára sobre todas as entradas. O tempo de execução ou complexidade de tempo de M é a função f: N → N, onde f(n) é o número máximo de passos que M usa sobre entradas de comprimento n. É um costume generalizado usar n para representar o comprimento da entrada.

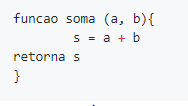
6) O algoritmo que tiver a função com o menor valor representa uma solução melhor, ou seja, o algoritmo precisa de menos tempo ou passos para resolver o determinado problema. O algoritmo com a menor complexidade de tempo é o melhor.

7) É o tamanho da entrada.

8) Atribuição de valores a variáveis, Chamadas de métodos, Operações aritméticas (por exemplo, adição de dois números), Comparação de dois números, Acesso a um arranjo, Seguimento de uma referência para um objeto, Retorno de um método.

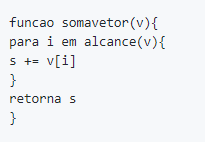
9) Para cada operação primitiva se atribui o valor 1.

10)



f(n) = 2

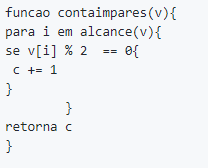
11)



f(n) = n + n + 1

f(n) = 2n + 1

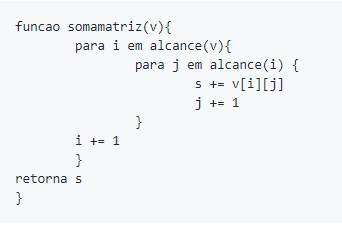
12)



f(n) = n + n + n + 1

f(n) = 3n + 1

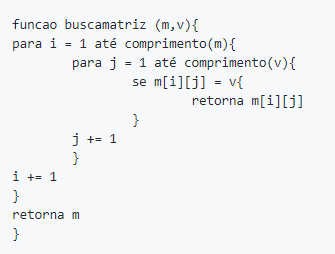
13)

****

f(n) = n + n*n + n*n + n\*n + n + 1

f(n) = 3n² + 2n + 1

14)

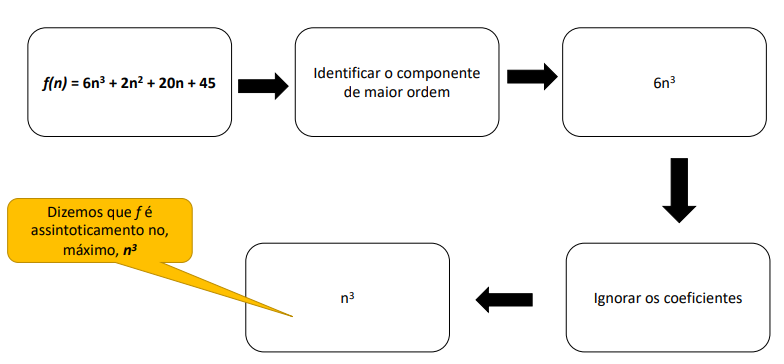


f(n) = n + n*n + n*n+ n\*n +n + 1 + 1

f(n) = 3n² + 2n + 2

15) Análise assintótica é um método de descrever o comportamento de limites. com objetivo de compreender o tempo de execução para entradas grandes. Ou seja, é um método de descrever o comportamento de limites.

16) Compreender o tempo de execução para entradas grandes. Ou seja, é um método de descrever o comportamento de limites. Para isso, é considerado apenas os componentes de maior ordem de uma expressão. Os termos de baixa ordem é ignorado. Os coeficientes são também ignorados.



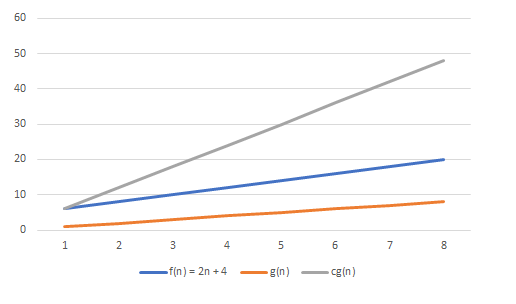
17) É uma notação para descrever uma fórmula matemática que se importa apenas com o maior componente da fórmula e ignora os fatores menores ou constantes.

18) Uma função é menor que ou igual a outra função g(n). F é limitada superiormente por g assintoticamente

19) Sejam f e g funções f,g: N → R+. Digamos que f(n) = O(g(n)) se existem inteiros positivos c e n0 tais que para todo inteiro n ≥ n0.Quando f(n) = O(g(n)) dizemos que g(n) é o limite superior para f(n) ou, mais precisamente, que g(n) é um limitante assintótico para f(n), para enfatizar que estamos suprimento fatores constantes.

20) n0 = 1

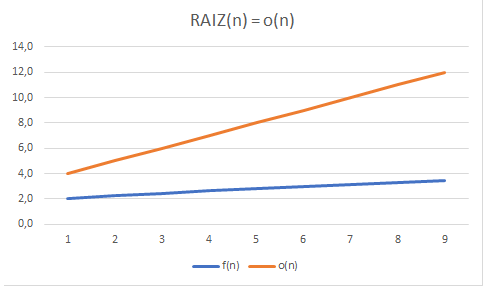
constante = 6



21) Uma é função é menor que a outra função g(n). F é denominado de g assintoticamente.

22) Sejam f e g funções f,g: N → R+. Digamos que f(n) = o(g(n)) se Em outras palavras, f(n) = O(g(n)) significa que, para qualquer número real c > 0, existe um número n0, onde f(n) < cg(n) para todo n ≥ n0.

23)



24)

#### **A F(n) = n + 1**

O (n)

o(n logn)

#### **B F(n) = 8**

O (1)

o (1)

#### **C F(n) = 2n 2 − 1**

O(n2)

o(2^n)

#### **D F(n) = nlogn**

O (nlogn)

o(n^2)

#### **E F(n) = 3n! + 2n**

O (n!)

o(n.n!)

#### **F F(n) = 3n 3 + 2n 2 + 4n + 6**

O (n3)

o(n!)

#### **G F(n) = 5^n + 11**

O (n)

o (n logn)

#### **H F(n) = 3logn**

O (n)

o(n)

25)

#### **10) f(n) = 2**

O(1)

#### **11) f(n) = 2n + 1**

O(n)

#### **12) f(n) = 3n + 1**

O(n)

#### **13)f(n) = 3n² + 2n + 1**

O(n2)

#### **14) f(n) = 3n² + 2n + 2**

O(n2)