

Universidade Federal de Uberlaˆndia - UFU

Faculdade de Computac¸a˜o - FACOM

Lista de exerc´ıcios de estrutura de dados em linguagem C

Exerc´ıcios: Ana´lise de complexidade

1. Complexidade da função g(n) é proporcional a função f(n)
2. Suponha um algoritmo A e um algoritmo B com func¸o˜es de complexidade de tempo *a*(*n*) = *n*2 *n* + 549 e *b*(*n*) = 49*n* + 49, respectivamente. Determine quais sa˜o os valores de *n* pertencentes ao conjunto dos nu´meros naturais para os quais A leva menos tempo para executar do que B.

*—*

1. Expresse a func¸a˜o 10*n*3 *−* 5*n*2 *−* 10*n* + 3 em termos da notac¸a˜o Θ.
2. E´ verdade que 2*n*3 + 5 = Θ(*n*3)? Explique.
3. Dois algoritmos A e B possuem complexidade *n*5 e 2*n* respectivamente. Voceˆ utilizaria o algoritmo B ao inve´s do A, em qual caso? Explique.
4. Qual a ordem de complexidade no pior caso de:
   1. 2*n* + 10
   2. (1*/*2)*n*(*n* + 1)
   3. *n* + *√n*
   4. *n/*1000
   5. (1*/*2)*n*2
   6. (1*/*2)*n*2 *−* 3*n*
5. Quais as grandezas f´ısicas que influenciam a eficieˆncia de tempo de um algoritmo na pra´tica?
6. Para o ca´lculo da complexidade de algoritmos na˜o recursivos, existe um conjunto de regras bastante simples de serem seguidas. Cite e descreva estas regras.
7. Explique que tipos de problemas ou algoritmos costumam ter complexidade da ordem de

*n* log *n* e como os identificamos.

1. Quais problemas que possuem geralmente complexidade da ordem de *logn*?
2. Quais problemas que costumam ser exponenciais?
3. Calcule a complexidade, no pior caso, do fragmento de co´digo abaixo:
4. **i n t** i , j , k ;
5. **for** ( i = 0 ; i *<* N; i ++){

3 **for** ( j = 0 ; j *<* N; j ++){

4 R[ i ] [ j ] = 0 ;

5 **for** ( k = 0 ; k *<* N; k ++)

6 R[ i ] [ j ] += A [ i ] [ k ] ∗ B [ k ] [ j ] ;

7 }

8 }

1. Calcule a complexidade, no pior caso, do fragmento de co´digo abaixo:
2. **i n t** i , j , k , s ;
3. **for** ( i = 0 ; i *<* N−1; i ++)
4. **for** ( j = i + 1 ; j *<* N; j ++)
5. **for** ( k = 1 ; k *<* j ; k ++)

5 s = 1 ;

1. Calcule a complexidade, no pior caso, do fragmento de co´digo abaixo:
2. **i n t** i , j , s ;
3. s = 0 ;
4. **for** ( i = 1 ; i *<* N−1; i ++)

4 **for** ( j = 1 ; j *<* 2∗N; j ++)

5 s = s + 1 ;

1. Obtenha a equac¸a˜o matema´tica referente a´ ana´lise do pior e melhor caso do fragmento de co´digo abaixo:
2. **for** ( i = 0 ; i *<* N; i ++)
3. p r i n t f ( ”%d ” , i ) ;
4. Obtenha a equac¸a˜o matema´tica referente a´ ana´lise do pior e melhor caso do fragmento de co´digo abaixo:

1 **for** ( i = 0 ; i *<* N; i = i +2)

2 p r i n t f ( ”%d ” , i ) ;

1. Obtenha a equac¸a˜o matema´tica referente a´ ana´lise do pior e melhor caso do fragmento de co´digo abaixo:

1 **for** ( i = 0 ; i *<* N; i = i +2) {

1. p r i n t f ( ”%d ” , i ) ;
2. i −−;

4 }

1. Obtenha a equac¸a˜o matema´tica referente a´ ana´lise do pior e melhor caso do fragmento de co´digo abaixo:

1 **for** ( i = 0 ; i *<* N; i = i +2) {

2 **for** ( j = N−i ; j *>*=0; j −−){

3 **i f** ( V [ i ] *<* V [ j ] ) {

1. p r i n t f ( ”%d ” , i ) ;

5 }

6 }

7 }

1. Obtenha a equac¸a˜o matema´tica referente a´ ana´lise do pior e melhor caso do fragmento de co´digo abaixo:

1 **for** ( i = 1 ; i *<*= N; i =2∗ i )

2 p r i n t f ( ”%d ” , i ) ;

1. Escreva um algoritmo que receba valores em um vetor e imprima “ORDENADO” se o vetor estiver em ordem crescente. Qual e´ func¸a˜o de custo de pior caso e sua ordem de complexidade *O*?
2. Escreva um algoritmo que receba um vetor ordenado e um nu´mero extra e insira esse nu´mero na sua posic¸a˜o correta no vetor ordenado, deslocando os outros nu´meros se necessa´rio. Quais sa˜o sua func¸a˜o de custo e ordens de complexidade *O* e Ω?
3. Escreva um algoritmo que procure por um dado nu´mero em vetor ordenado. Quais sa˜o sua func¸a˜o de custo e ordens de complexidade *O* e Ω?
4. Escreva um algoritmo eficiente que procure por um dado nu´mero em vetor ordenado. Quais sa˜o sua func¸a˜o de custo e ordens de complexidade *O* e Ω?