

1ª Lista de Exercícios – 2015.1

Questão 1:

Qual a ordem entre as funções 2^n , 1 , n , n^2 , n^3 , n^4 , $n!$, $n \cdot \log_2 n$ e $\log_2 n$ para valores de n grandes?

Questão 2:

- Apresente um algoritmo de busca em um vetor de tamanho N e a sua complexidade em notação O .
- Suponha que o vetor está ordenado, apresente um algoritmo de busca eficiente e a sua complexidade.
- Suponha que você mantém um vetor de tamanho máximo N , ordenado e com M posições preenchidas. Desta forma você pode usar um algoritmo otimizado para busca no vetor ordenado. Apresente um algoritmo de inserção eficiente para inserir um novo elemento neste vetor, mantendo-o e indique qual a complexidade deste algoritmo de inserção. Dica: um algoritmo de ordenação pode ser base de sua solução.

Questão 3:

Os algoritmos QUICKSORT e MERGESORT são descritos brevemente abaixo:

```
quickSort(A[], esq, dir) {
    if (dir > esq) {
        r = particiona(A, esq, dir);
        quickSort(A, esq, r - 1);
        quickSort(A, r + 1, dir);
    }
}

mergeSort(A[], ini, fim) {
    if (fim != ini) {
        mid = (fim - ini + 1) / 2;
        mergeSort(ini, mid);
        mergeSort(mid, fim);
        merge(A[], ini, fim);
    }
}
```

- Considere que o pivô é escolhido o da primeira posição. No caso médio este algoritmo executa $N \cdot \ln(N)$ passos, mas o pior caso é de N^2 passos.
- Explique com um exemplo o porquê deste pior caso.
- O MERGESORT executa $N \cdot \log_2(N)$ passos em qualquer caso, e $N \cdot \ln(N) = 1,39 \cdot N \cdot \log_2(N)$. Ou seja, o MERGESORT tem menos passos que o caso médio do QUICKSORT, apresente as razões para se utilizar o QUICKSORT ao invés do MERGESORT.
- Compare o melhor caso do MERGESORT com o do QUICKSORT.

Questão 4:

O algoritmo MERGE do MERGESORT funciona de modo que dados os vetores ordenados A e B , cada um de tamanho n , retorne um vetor C de tamanho $2n$ com todos os elementos de A e B em ordem crescente.

Exemplo:

A: 1, 5, 9, 19
B: 7, 12, 18, 30
C: 1, 5, 7, 9, 12, 18, 19, 30

- Apresente uma versão em C para o algoritmo MERGE e indique qual a sua complexidade de pior caso em notação O .
- Apresente uma versão modificada do algoritmo que ordene de forma decrescente o vetor C .

Questão 5:

Um algoritmo de ordenação é estável se não altera a posição relativa de elementos com mesmo valor. Por exemplo, se o vetor $v[0..n-1]$ tiver dois elementos iguais a 222, primeiro um sublinhado e depois um sublinhado duplo, um algoritmo de ordenação estável mantém o 222 sublinhado simples antes do sublinhado duplo.

original:	444	555	555	666	777	333	999	222	111	222	888
ordenado:	111	<u>222</u>	<u>222</u>	333	444	555	555	666	777	<u>888</u>	999

- Os algoritmos BUBBLESORT, INSERTION SORT, SELECTION SORT, QUICKSORT e MERGESORT são algoritmos de ordenação estável? Justifique sua resposta com argumentação e exemplos em cada caso.

Questão 6:

Considere o cálculo da série de *Fibonacci* por meio da função recursiva abaixo:

```
fibonacci(n) {
    if (n <= 1) {
        return 1;
    }
    else {
        return fibonacci(n-1) + fibonacci(n-2);
    }
}
```

- Execute um teste de mesa para $n=10$, apresentando o número de passos gasto.
- Qual o problema em termos de eficiência (número de passos) desta abordagem? Calcule a complexidade.
- Apresente uma solução mais eficiente.

Questão 7:

Considere os algoritmos de ordenação INSERTION (inserção) e MERGESORT para ordenar um vetor com N posições e as seguintes observações:

– O algoritmo INSERTION tem número de passos médio igual a $N^2/4$ e de pior caso, $N^2/2$.

– O algoritmo MERGESORT tem número de passos $N \cdot \log_2 N$.

- Considerando o caso médio, qual destes algoritmos deve-se utilizar para um vetor de 10 posições? Qual a justificativa?
- E ainda considerando o caso médio, qual destes algoritmos deve-se utilizar para um vetor de 50 posições? Qual a justificativa?
- Proponha um algoritmo de ordenação que utilize o melhor método de acordo com o tamanho do vetor.

Questão 8:

Teorema Mestre:

Sejam a, b, c, k constantes não negativas com $a > 0$, $b > 1$, a solução para $T(n) = aT(n/b) + cn^k$ é :

$T(n)$:

$O(n^{\log_b a})$, $\log_b a > k$

$O(n^k \log n)$, $\log_b a = k$

$O(n^k)$, $\log_b a < k$

Dados os algoritmos abaixo indique a complexidade no pior caso na forma de $T(n)$ e de notação O. Considere que na notação O deve ser indicada dentre as funções 2^n , 1 , n , $n!$, n^2 , n^3 , n^4 , $n \cdot \log n$, $\log n$, a que melhor expresse o comportamento dos algoritmos.

```
proc1 (n, A) {
for (i = 0; i < n/3; i++)
  for (j = n/3-1; j > i; j--)
    troque(A[i], A[j]);
}
```

```
proc2(n) {
if (n == 1)
  return 1;
else {
  valor = proc2(n / 2);
  Para i = 1 até n faça
    valor = valor + i;
  result = valor + proc2(n /
2);
  return result;
}
}
```

```
hanoi (n, a, b, c) {
if (n > 1) {
  hanoi(N-1, a, c, b);
  move(1, a, b); // considere
complexidade de 1 passo
  hanoi(N-1, c, b, a);}
}
```

```
proc3 (n) {
if (n < 1) {
  return 0; }
else
  return n + proc3(n / 3);
}
```

```
proc4 () {
for (i = 0; i < n; i++) {
  for (j = i+1; j < n; j++) {
    for (k = 1; k < n; k++) {
      A[i][j] = A[i][j]+k;
    } } }
}
```

```
selecao(A[], N) {
for (i=1; i < n-1; i++) {
  min = i
  for (j = i+1; j < n; j++)
    if A[j] < A[min] then
      min = j
  troca(A, min, i )
}
```

Questão 9:

Compare o algoritmo INSERTION SORT com o algoritmo SELECTION SORT. Indique os pontos positivos e negativos de cada um, a quantidade de trocas e comparações no caso médio, pior e melhor caso e em quais cenários cada um é adequado.

Questão 10:

Considere um vetor A com n elementos, onde o maior valor permitido aos elementos é n e o menor valor é 1. Apresente um algoritmo de ordenação que utilize essa informação de forma eficiente e não utilize nenhum método de ordenação estudado até então.

Questão 11:

Dois algoritmos A e B para o mesmo problema possuem complexidade $5 \cdot n^5$ e 2^n , respectivamente. Explique, com exemplos, em que condições deve-se utilizar o algoritmo B ao invés do algoritmo A.

Questão 12:

Explique o conceito de complexidade de espaço, apresente um exemplo de algoritmo para ilustrar sua explicação.