Curso de Tecnologia em Sistemas de Computação
Disciplina: Estrutura de Dados e Algoritmos
Gabarito da AP1 - Primeiro Semestre de 2017

Nome -
Assinatura -

Observações:

1. Prova sem consulta e sem uso de máquina de calcular.
2. Use caneta para preencher o seu nome e assinar nas folhas de questões e nas folhas de respostas.
3. Você pode usar lápis para responder as questões.
4. Ao final da prova devolva as folhas de questões e as de respostas.
5. Todas as respostas devem ser transcritas nas folhas de respostas. As respostas nas folhas de questões não serão corrigidas.

---

1. (2,0) Considere as seguintes estruturas de dados: Lista Sequencial Não Ordenada, Lista Sequencial Ordenada e Lista Encadeada Não Ordenada. Suponha que o número de elementos nas listas é igual a n.

   (a) Com relação ao algoritmo de busca, determine a complexidade de pior caso deste algoritmo para cada uma destas estruturas. (Justifique com exemplos.)

   Resposta: Em cada uma das listas a seguir consideramos a busca pelo elemento 1.
   –Lista Sequencial Não Ordenada: complexidade de pior caso igual a $O(n)$, dado que é necessário percorrer todo a lista no pior caso.
   Ex: 8 7 6 5 4 3 2 1
   –Lista Sequencial Ordenada: complexidade de pior caso igual a $O(\log n)$, uma vez que é possível efetuar uma busca binária por um elemento qualquer.
   Ex: 1 2 3 4 5 6 7 8
   –Lista Encadeada Não Ordenada: complexidade de pior caso igual a $O(n)$, visto que é necessário percorrer toda a lista no pior caso.
   Ex: 8 7 6 5 4 3 2 1

   (b) Suponha que você já localizou na lista um elemento determinado, o qual você deseja remover. Qual é o pior caso desta operação para cada uma destas estruturas? (Justifique com exemplos.)

   Resposta: Em cada uma das listas a seguir consideramos a busca pelo elemento 1.
   –Lista Sequencial Não Ordenada: complexidade de pior caso igual a $O(1)$, dado que podemos trocar de posição o último elemento da lista na posição que contém o último elemento a ser removido, além de considerar a lista resultante da posição inicial até a posição final menos uma unidade.
   Ex: lista inicial 1 7 6 5 4 3 2 8 e lista resultante 8 7 6 5 4 3 2
   –Lista Sequencial Ordenada: complexidade de pior caso igual a $O(n)$, uma vez que é necessário copiar todos os elementos posteriores à posição do elemento a ser removido em uma unidade a menos cada uma para que a lista resultante continue ordenada.
   Ex: lista inicial 1 2 3 4 5 6 7 8 e lista resultante 2 3 4 5 6 7 8
   –Lista Encadeada Não Ordenada: complexidade de pior caso igual a $O(1)$. Sejam N_1 e N_2 nós da lista tais que N_1 precede N_2 na lista e N_2 contém o elemento a ser removido. É necessário apenas trocar o ponteiro $N_1.\uparrow prox$ para $N_2.\uparrow prox$ e desalocar o espaço de memória ocupado por N_2. O mesmo ocorre caso N_2 seja o primeiro nó da lista, ou seja, basta fazer com que o ponteiro para a cabeça da lista aponte para $N_2.\uparrow prox$.
   Ex: lista inicial 8 7 6 1 4 3 2 5 e lista resultante 8 7 6 4 3 2 5

2. (2,0) Os números 1, 2, 3, 4, 5 são inseridos em uma pilha nesta ordem. Porém, estas cinco operações de inserção são intercaladas com operações de remoção, e a cada remoção é impresso o número desempilhado. Considere os exemplos abaixo, onde I representa uma operação de inserção, e R uma operação de remoção.

Exemplo 1: Se a sequência de operações na pilha for I R I R I I I R R R, a sequência de números impressos será 1 2 5 4 3.

Exemplo 2: Se a sequência de operações na pilha for I I I I I R R R R R, a sequência de números impressos será 5 4 3 2 1.

(a) Escreva a sequência de números impressos quando a sequência de operações na pilha é I I R R I I R R I R.

Resposta: 2 1 4 3 5.

(b) Determine a sequência de operações na pilha que resulta na seguinte sequência de impressão: 3 5 4 2 1.

Resposta: $I\ I\ I\ R\ I\ I\ R\ R\ R\ R$.

3. (2,0) É dada uma Lista Sequencial Não Ordenada contendo n números distintos, implementada em um vetor V com n posições indexadas de 1 até n. Suponha que $n \geq 2$. Escreva um algoritmo que encontra os dois maiores elementos desta lista, percorrendo o vetor V uma única vez da esquerda para a direita (isto é, percorrendo uma única vez as posições de 1 até n).

Resposta: O Algoritmo 1 retorna o segundo maior (SM) e o maior (M) valores, respectivamente.

Algoritmo 1: $Maior_e_SegundoMaior(V)$.

Entrada: Vetor V não ordenado composto por n valores distintos.
Saída: Os valores do maior e segundo maior elementos de V.

```
1  se V[1] < V[2] então
2  |  SM ← V[1];
3  |  M ← V[2];
4  senão
5  |  SM ← V[2];
6  |  M ← V[1];
7  para i ← 3, ..., n faça
8  |  se V[i] > SM então
9  |  |  se V[i] > M então
10 |  |  |  SM ← M;
11 |  |  |  M ← V[i];
12 |  |  senão
13 |  |  |  SM ← V[i];
14 retorna SM e M;
```

4. (2,0) Considere os seguintes algoritmos de ordenação: Ordenação por Seleção e Ordenação pelo Método da Bolha.

   (a) Determine todas as TROCAS de elementos efetuadas por estes métodos quando o vetor de entrada é $V = [9, 4, 10, 3, 5, 1]$.

   Resposta:–Trocas pela Ordenação por Seleção: são efetuadas 6 trocas.
   9 4 10 3 5 1^* Vetor inicial
   1 4 10 3^* 5 9
   1 3 10 4^* 5 9
   1 4 4 10 5^* 9
   1 4 4 5 10 9^*
   1 4 4 5 9 10^*
   1 4 4 5 9 10

   –Trocas pelo Método da Bolha: são efetuadas 11 trocas.
   9 4 10 3 5 1^* Vetor inicial
   9 4 10 3 1^* 5
   9 4 10 1^* 3 5
   9 4 1^* 10 3 5
   9 1^* 4 10 3 5
   1^* 9 4 10 3 5
   1 9 4 10 3^* 5
   1 9 4 3^* 10 5
   1 9 3^* 4 10 5
   1 3^* 9 4 10 5
   1 3 9 4^* 10 5
   1 3 4^* 9 10 5
   1 3 4 9 10 5^*
   1 3 4 9 5^* 10
   1 3 4 5^* 9 10

   (b) Verdadeiro ou falso? Para qualquer vetor de entrada com 6 elementos, a Ordenação por Seleção sempre faz MENOS trocas que a Ordenação pelo Método da Bolha.

   Resposta: Falso. Para um vetor ordenado de forma crescente com elementos distintos o número de trocas no algoritmo por Seleção é maior que o número de trocas da ordenação pelo Método Bolha, dado que no Método por Seleção é feita uma troca do menor elemento com ele próprio a cada passo, enquanto que no Método da Bolha não é feita qualquer troca neste caso.

5. (2,0) Assinale V ou F, justificando:

   (a) Se A é o melhor algoritmo conhecido para resolver um certo problema P, então A é um algoritmo ótimo.

Resposta: Falso. Não necessariamente o melhor algoritmo conhecido para resolver um problema P é ótimo. Um algoritmo é ótimo quando sua complexidade de pior caso é limitada superiormente pelo limite inferior para P. A multiplicação de matrizes é um exemplo em que o melhor algoritmo conhecido não é limitado superiormente pelo limite inferior para o problema.

(b) Qualquer algoritmo para buscar um elemento x numa Lista Sequencial Ordenada com n elementos precisa fazer n comparações no pior caso.

Resposta: Falso. Em uma lista sequencial ordenada é possível efetuar uma busca binária que possui complexidade de pior caso igual a $O(\log n)$.