Curso de Tecnologia em Sistemas de Computação
Disciplina: Estrutura de Dados e Algoritmos
Gabarito AP1 - Segundo Semestre de 2019

1. **Defina:**

a. **(1,0) Complexidade de pior caso de um algoritmo.**
Resposta: Seja A um algoritmo e E = {E_1, E_2, ..., E_n} o conjunto de todas as entradas possíveis de A. Dada a entrada E_i, seja t_i o número de passos efetuados por A, para $1 \leq i \leq n$. Podemos definir a *complexidade de pior caso* como MAX $_{Ei \in E}$ $\{t_i\}$

b. **(1,0) Algoritmo ótimo.**
Resposta: Seja *g* um limite inferior para um problema P. Um algoritmo ótimo A que resolve P é tal que sua complexidade é dada por $f = O(g)$. Dessa forma, o algoritmo A possui complexidade de pior caso $\Omega(g)$ e $O(g)$. Em outras palavras, um algorítmo é ótimo se sua complexidade de pior caso é dada pelo limite inferior para o problema.

2. **(1,5) Considere os algoritmos ORDENAÇÃO POR SELEÇÃO e ORDENAÇÃO PELO MÉTODO DA BOLHA. Qual dos dois efetua menos TROCAS de elementos quando a lista a ser ordenada encontra-se em ordem inversa de ordenação? (Ex: 10 9 8 7 6 5 4 3 2 1.) JUSTIFIQUE SUA RESPOSTA.**
Resposta: A **ordenação por seleção** executa $\left[\frac{n}{2}\right]$ trocas, uma vez que os elementos trocados ocupam suas posições finais no vetor. Isso pode ser verificado para a primeira torca no exemplo com 10 elementos, onde os números 1 e 10 trocam de lugar. Como 1 é o menor elemento do vetor e 10 o maior, os mesmos não mudam de posição novamente. Podemos aplicar o mesmo raciocínio nos demais passos.

Por outro lado, **a ordenação pelo método bolha** executa um número quadrático de trocas, já que em cada iteração, a bolha é trocada com todos os elementos que a precedem até o primeiro elemento menor que a mesma. Ou seja, executa-se $(n-1) + (n-2) + \cdots + 1 = \frac{n(n-1)}{2}$ trocas.

Portanto, a **ordenação por seleção** executa menos trocas que a **ordenação pelo método bolha**.

3. **(2,0) Considere uma lista simplesmente encadeada *L* com *n* nós, que armazenam números inteiros. Elabore um algoritmo que crie uma nova lista L' contendo somente nós com os números ímpares que ocorrem em *L*. Os números devem aparecer em L' na mesma ordem em que aparecem em *L*. Por exemplo, se *L* contiver os números 1, 8, 4, 5, 7, 8, 6, 3, nesta ordem, então L' conterá os números 1, 5, 7, 3, nesta ordem. Qual a complexidade do seu algoritmo?**

Resposta: A complexidade do algoritmo a seguir é Θ (n), pois percorre a lista L apenas uma vez, e para cada nó, L executa um número constante de passos.

```
pt := L          % considerando que L não tem nó cabeça
L' := λ
ultimo := λ
enquanto pt ≠ λ faça
    se pt↑.info mod 2 ≠ 0 então
            ocupar(novo)
            novo↑.info := pt↑.info
            novo↑ .prox := λ
            se ultimo ≠ λ então      % L' já contém algum nó
                    ultimo↑.prox := novo
            senão
                    L' := novo
                    último := novo
    pt := pt↑.prox
```

4. **(1,5) Considere uma pilha *P* contendo 5 posições de 1 a 5. A variável *topo* marca a posição do topo da pilha. No início, a pilha *P* encontra-se vazia, e a variável *topo* vale zero.**

Usamos a notação *R* para denotar a operação de remoção de um elemento da pilha *P*, e a notação *I*(*X*) para denotar a operação de inserção de um elemento *X* na pilha *P*.

Considere a seguinte sequência de operações em *P*:

***I*(*A*), *I*(*B*), *I*(*C*), *R*, *I*(*D*), *R*, *I*(*E*), *I*(*G*), *I*(*H*), *R*, *R*, *R*, *R*, *I*(*J*)**

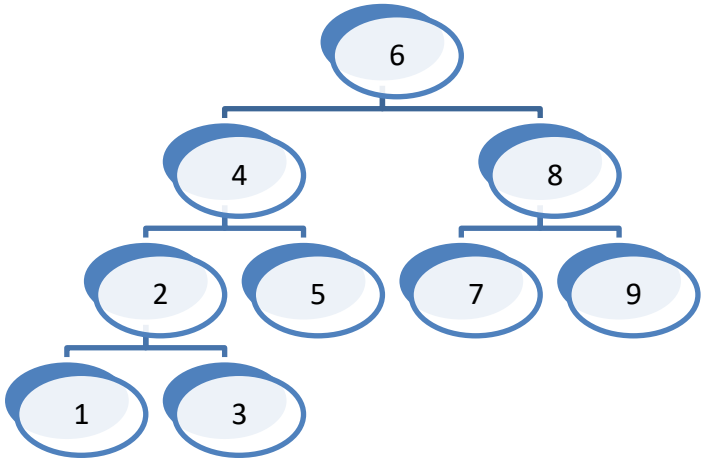
Desenhe como fica a pilha *P* após a sequência de operações acima, e forneça o valor final da variável *topo*. Use um traço (–) para denotar as posições vazias. Como um exemplo de configuração, poderíamos ter: *P* = [*C D H* – –], onde *topo* neste caso vale 3.

Resposta: Ao fim da execução desta sequência, a pilha ficaria com a seguinte configuração:

P = [A J – – –], topo = 2.

5. **Para cada item abaixo, desenhe uma árvore binária de busca *T* de altura 4 (colocando valores de chaves nos respectivos nós) que atenda às condições descritas:**

a. **(1,5) *T* é uma árvore completa, estritamente binária e com um número mínimo de nós.**
Resposta: Uma árvore binária completa é aquela em que todos os nós com alguma subárvore vazia estão no último ou penúltimo nível. Uma árvore estritamente binária é aquela em que todos os nós possuem exatamente 0 ou exatamente 2 filhos. Abaixo é apresentada uma árvore que atende ao solicitado.



b. **(1,5) *T* é uma árvore binária, não é cheia e tem um número máximo de nós.**
Resposta: Uma árvore cheia é aquela em que todos os nós com alguma subarvore vazia estão no último nível. Abaixo, é apresentada uma árvore que atende ao solicitado.