Estrutura de Dados - 1º. período de 2016

Primeira Avaliação a Distância

1. (1,0) Escreva as seguintes funções em notação Θ: $3^n + n^3 \log n$, $n^3 + n^2 \log n$, $n! + 2^n$, $\sqrt{n} + \log n$.

   Resposta: $3^n + n^3 \log n = \Theta(3^n)$, $n^3 + n^2 \log n = \Theta(n^3)$, $n! + 2^n = \Theta(n!)$, $\sqrt{n} + \log n = \Theta(\sqrt{n})$.

2. Para cada item abaixo, responda "certo" ou "errado", justificando:

   a. (0,5) Se a complexidade de pior caso de um algoritmo for $\Theta(n^2)$, então o melhor caso deste algoritmo é $O(n^2)$.

   Resposta: Certo. A complexidade de pior caso determina um limite superior para a complexidade de melhor caso. Porém, isso não significa que a complexidade de melhor caso seja $\Theta(n^2)$.

   b. (0,5) Se a complexidade de pior caso de um algoritmo for f, então o algoritmo é $\Theta(f)$.

   Resposta: Errado. Neste caso o algoritmo é $O(f)$, já que f é a complexidade de pior caso. Porém, nada se pode afirmar quanto a complexidade de melhor caso.

   c. (0,5) Seja P um problema com limite inferior $n \log n$. Se um algoritmo resolve P em tempo $\Omega(n \log n)$, então este algoritmo é ótimo.

   Resposta: Errado. Um algoritmo é ótimo quando sua complexidade de **pior caso** é igual ao limite inferior para o problema.

   d. (0,5) Seja P um problema com limite inferior ℓ. Então qualquer algoritmo que resolve P tem complexidade de melhor caso $\Omega(\ell)$.

   Resposta: Errado. O problema de ordenação é um exemplo em que o limite inferior é $n \log n$, mas sua complexidade de melhor caso é $\Omega(n)$.

3. (2,0) Sejam L_1 e L_2 duas listas encadeadas ordenadas, com nó cabeça, tais que $|L_1| = m$ e $|L_2| = n$. Elabore um algoritmo que imprima os elementos que pertencem a apenas uma das listas. Seu algoritmo deverá executar em $\Theta(n + m)$ passos.

   Resposta: O Algoritmo 1 representa tal algoritmo. Das linhas 3–24 é feita a comparação dos elementos de ambas as listas até que uma delas se torne vazia. As linhas 6–12 representam o caso em que o primeiro elemento de L_1 é o menor dentre os elementos ainda não analisados, enquanto as linhas 13–19 são análogas em relação a L_2. As linhas 9–12 e 16-19 tratam o caso em que as listas possuem elementos repetidos, onde os mesmos são ignorados uma vez que o primeiro deles já foi selecionado para impressão através do ponteiro menor. As linhas 29–37 representam o mesmo das linhas 9–12 e 16-19 em relação ao restante da lista ainda não vazia. Como cada lista é percorrida apenas uma vez, o algoritmo é $\Theta(n + m)$.

Algoritmo 1: Imprime_Sem_Repetição(PTlista_1, PTlista_2)

Entrada: Nós cabeça PTlista_1 e PTlista_2 das listas ordenadas e simplesmente encadeadas L_1 e L_2, respectivamente.

```
pt_1 ← PTlista_1↑.prox;
pt_2 ← PTlista_2↑.prox;
enquanto pt_1 ≠ λ e pt_2 ≠ λ faça
    menor ← λ;  // Ponteiro para o menor elemento das duas listas ainda não impresso.
    sair ← falso;
    se pt_1↑.info < pt_2↑.info então
        menor ← pt_1;
        pt_1 ← pt_1↑.prox;
        enquanto pt_1 ≠ λ e sair = falso faça
            se pt_1↑.info = menor↑.info então
                pt_1 ← pt_1↑.prox;
            senão sair ← verdadeiro;
    senão se pt_1↑.info > pt_2↑.info então
        menor ← pt_2;
        pt_2 ← pt_2↑.prox;
        enquanto pt_2 ≠ λ e sair = falso faça
            se pt_2↑.info = menor↑.info então
                pt_2 ← pt_2↑.prox;
            senão sair ← verdadeiro;
    se menor ≠ λ então
        Imprime menor↑.info;
    senão
        pt_1 ← pt_1↑.prox;
        pt_2 ← pt_2↑.prox;
se pt_1 = λ então
    pt ← pt_2 ;  // Ponteiro para o primeiro elemento da lista não vazia.
senão
    pt ← pt_1;
enquanto pt ≠ λ faça
    Imprime pt↑.info;
    menor ← pt;
    sair ← falso;
    pt ← pt↑.prox;
    enquanto pt ≠ λ e sair = falso faça
        se pt↑.info = menor↑.info então
            pt ← pt↑.prox;
        senão sair ← verdadeiro;
```

4. Considere a lista: 15 7 10 2 80 24 1. Desenhe as trocas de elementos e determine o número de trocas efetuadas, utilizando:

   a. (1,0) Ordenação por seleção

   Resposta: São efetuadas 4 trocas.
   15 7 10 2 80 24 1 (Vetor Inicial.)
   1 7 10 2 80 24 15 (15 $\rightleftarrows$ 1)
   1 2 10 7 80 24 15 (7 $\rightleftarrows$ 2)
   1 2 7 10 80 24 15 (10 $\rightleftarrows$ 7)
   1 2 7 10 80 24 15 (Não há troca.)
   1 2 7 10 15 24 80 (80 $\rightleftarrows$ 15)
   1 2 7 10 15 24 80 (Não há troca.)
   1 2 7 10 15 24 80 (Não há troca.)

   b. (1,0) Ordenação por bolha

   Resposta: A bolha é marcada com *. São efetuadas 12 trocas.
   15* 7 10 2 80 24 1 (Vetor Inicial.)
   15 7* 10 2 80 24 1
   7* 15 10 2 80 24 1 (15 $\rightleftarrows$ 7)
   7 15 10* 2 80 24 1
   7 10* 15 2 80 24 1 (15 $\rightleftarrows$ 10)
   7 10 15 2* 80 24 1
   7 10 2* 15 80 24 1 (15 $\rightleftarrows$ 2)
   7 2* 10 15 80 24 1 (10 $\rightleftarrows$ 2)
   2* 7 10 15 80 24 1 (7 $\rightleftarrows$ 2)
   2 7 10 15 80* 24 1
   2 7 10 15 80 24* 1
   2 7 10 15 24* 80 1 (80 $\rightleftarrows$ 24)
   2 7 10 15 24 80* 1
   2 7 10 15 24 80 1*
   2 7 10 15 24 1* 80 (80 $\rightleftarrows$ 1)
   2 7 10 15 1* 24 80 (24 $\rightleftarrows$ 1)
   2 7 10 1* 15 24 80 (15 $\rightleftarrows$ 1)
   2 7 1* 10 15 24 80 (10 $\rightleftarrows$ 1)
   2 1* 7 10 15 24 80 (7 $\rightleftarrows$ 1)
   1* 2 7 10 15 24 80 (2 $\rightleftarrows$ 1)
   1 2 7 10 15 24 80 (Vetor Final.)

5. (2,0) Escreva um algoritmo que inverte uma lista simplesmente encadeada com nó cabeça. Exemplo: se a lista contém os elementos $1, 3, 5, 9, 2$, nesta ordem, então a lista resultante contém os elementos $2, 9, 5, 3, 1$. Você deve utilizar uma pilha auxiliar para resolver este problema.

   Resposta: O Algoritmo 2 representa a inversão de uma lista utilizando uma pilha P.

Algoritmo 2: Inverte_Lista_Encadeada(PTlista)

Entrada: Nó cabeça PTlista da lista simplesmente encadeada L.
Saída: Nova lista simplesmente encadeada L' com elementos de L em ordem inversa.

pt $\leftarrow$ PTlista$\uparrow$.prox;
topo $\leftarrow 0$;
enquanto pt $\neq \lambda$ **faça**
    topo $\leftarrow$ *topo* $+ 1$;
    P[*topo*] $\leftarrow$ pt$\uparrow$.info;
    pt $\leftarrow$ pt$\uparrow$.prox;
Ocupar(PTLista$'$);
// Cria L' vazia com nó cabeça apontado por PTLista$'$.
PTLista$'$ $\uparrow$.prox $\leftarrow \lambda$;
ultimo $\leftarrow$ PTLista$'$;
enquanto *topo* > 0 **faça**
    Ocupar(pt); // Cria novo nó enderaçado por pt.
    pt$\uparrow$.info $= P$[*topo*];
    pt$\uparrow$.prox $= \lambda$;
    ultimo$\uparrow$.prox $\leftarrow$ pt;
    ultimo $\leftarrow$ pt;
    topo $\leftarrow$ *topo* $- 1$;
retorna PTLista$'$;

---

6. (1,0) Desenhe uma árvore binária de busca completa, estritamente binária, de altura 4 e com o mínimo de nós. Lembre-se de colocar os valores dentro dos nós.

   Resposta: A Figura 1 representa tal árvore.

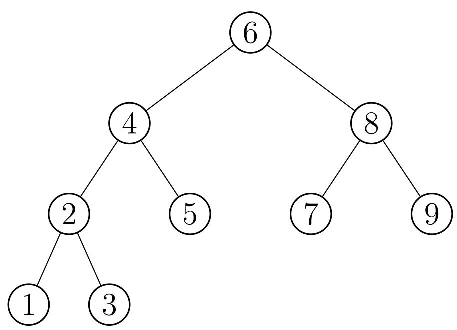

Figura 1: Árvore binária de busca completa, estritamente binária, de altura 4 e com o mínimo de nós.