Prova 1 - 10 de agosto de 2020

Leia as instruções abaixo com atenção:

- Responda às questões a caneta, em papel branco. Não use o verso se for um papel fino.
- Identifique com clareza a questão que está sendo respondida. Preferencialmente use páginas diferentes para questões diferentes.
- Fotografe ou escaneie as páginas ao final da prova, e envie um único email para paadccufmg@gmail.com com todas as páginas de respostas. Escreva "Prova 1 - [seu nome]" no assunto do email.
- **O prazo limite para recebimento do email é às 15h15.** Recomendo veementemente que interrompa a resolução da prova às 15h, para ter alguma folga no processo de organização e envio do email.

Questão 1 (9 pontos)**.**

(a) Seja $T(n)$ uma função definida recursivamente, com $T(1) = 1$, e $T(n) = T(n-1) + n$. Demonstre, por indução, que
$$T(n) \in \Theta(n^2).$$

(b) Seja $S(n)$ uma função definida recursivamente por $S(0) = S(1) = 1$, e $S(n) = 2S(n-2) + 3$.

   (i) Esboce a árvore de recorrência desta função.

   (ii) Quantos níveis esta árvore possui?

   (iii) Qual o custo do k-ésimo nível desta árvore? (conte de cima para baixo, com o primeiro nível sendo $k = 0$)

   (iv) Encontre uma função $f(n)$, definida explicitamente, tal que $S(n) \in \Theta(f(n))$ (mas não é necessário mostrar por indução que seu chute funciona).

(c) Se $R(n)$ definida recursivamente por $R(n) = 4R(n/2) + n^2 \log n$. Use o Teorema Mestre, e encontre uma função $f(n)$ tal que $R(n) \in \Theta(f(n))$.

Questão 2 (6 pontos)**.** Considere o algoritmo abaixo.

Algorithm 1 Girassol(A)

Entrada: Lista $A = A[1 \cdots n]$, com n inteiros

```
n = A.length;
for j = n − 1 to 1 do
    key = A[j]
    i = j + 1
    while i ≤ n and A[i] < key do
        A[i − 1] = A[i]                (∗)
        i = i + 1
    A[i − 1] = key
```

(a) O que este algoritmo realiza na lista A?

(b) Para se demonstrar que `Girassol` funciona corretamente, a técnica de *indução matemática* é a mais apropriada. Para tal, qual hipótese indutiva, ou loop invariante para o `for`, você utilizaria?

(c) Suponha que a entrada da lista acima é uma lista com todos os números de 1 até n, e que foi gerada de modo uniformemente aleatório dentre todas as permutações de n elementos. Qual a quantidade total esperada de operações da linha $(*)$ este algoritmo executará?

Dica: Denote por X a variável aleatória que conta o total de operações, e por X_i a variável aleatória que conta o total de operações na i-ésima iteração do `for`.

Questão 3 (6 pontos)**.**

Algorithm 2 Permuta(A)

Entrada: Lista $A = A[1 \cdots n]$, com n inteiros, $n > 1$.

```
n = A.length;
for i = 1 to n do
    j = número escolhido de modo uniformemente aleatório de 1 a n, menos o i
    swap A[i] with A[j]
```

Para cada uma das frases abaixo, julgue se é verdadeira ou falsa. Se for verdadeira, ofereça uma demonstração. Se for falsa, ofereça um contra-exemplo.

(a) Este algoritmo gera uma permutação da lista A que nunca é igual à permutação identidade.

(b) Com exceção da permutação que inverte exatamente a ordem dos elementos de A, este algoritmo gera qualquer uma das outras permutações com probabilidade constante e uniforme.

Questão 4 (6 pontos)**.** Deseja-se implementar uma fila F (first-in first-out) usando duas pilhas P e Q (last-in first-out), que possuem as ações de Push e Pop. Em linhas gerais, esta implementação funciona assim:

1. Enqueue(x, F) executa Push(x, P).

2. Dequeue(F) é determinada pelas opções:

   - Caso Q não esteja vazia, Dequeue(F) = Pop(Q).
   - Caso Q esteja vazia e P não esteja, efetuamos de modo iterado x =Pop(P) e em seguida Push(x, Q), até que P fique vazia. Ao final, fazemos Dequeue(F) = Pop(Q).
   - Caso ambas as pilhas estejam vazias, nada acontece.

Assuma que cada operação Push ou Pop nas pilhas P ou Q têm custo constante igual a 1.

Uma sequência de n operações do tipo Enqueue ou Dequeue é realizada na fila F. Apesar de potencialmente uma operação de Dequeue poder ser muito custosa caso a pilha Q esteja vazia e a pilha P tenha muitos elementos, você desconfia que o custo dessas n operações está limitado em $O(n)$.

(a) Em no máximo 3 linhas de texto, argumente por que você acha isso.

(b) Determine custos amortizados para Enqueue e Dequeue em F que permitam concluir que as n operações tem custo limitado em $O(n)$. Explique sua escolha.