Prova 1 - 18 de dezembro 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 submeta seu pdf na Tarefa (ou Assignment) desta prova, no Team de PAA.
- **O prazo limite para recebimento do arquivo é às 16h30. A tarefa será encerrada após este horário, e não será possível o envio de arquivo.** Recomendo veementemente que você encerre a prova as 16h, para ter tempo e calma para envio do arquivo.

Questão 1 (5 pontos)**.** Considere o seguinte algoritmo.

Algorithm 1 Girassol(A)

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

Saída: Número inteiro.

```
n = A.length;
key = A[1];
for j = 2 to n do
    if A[j] < key then
        key = A[j];
return key;
```

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

(b) Para demonstrar que este algoritmo realmente funciona, pode-se utilizar a técnica de indução matemática. Para tal, qual hipótese indutiva, ou loop invariante, você utilizaria?

Questão 2 (6 pontos)**.** Sejam $f, g : \mathbb{Z} \to \mathbb{R}$, e tais que $f(n) \geq 0$ e $g(n) \geq 0$ para todos $n \geq 0$. Defina função $h(n) = \min\{f(n), g(n)\}$, ou seja, para todo $n \in \mathbb{Z}$, $h(n) = f(n)$ se $f(n) \leq g(n)$, e $h(n) = g(n)$ se $g(n) < f(n)$.

(a) Matematicamente, o que significa dizer que $f(n) + g(n) \in \Omega(h(n))$?

(b) Demonstre que $f(n) + g(n) \in \Omega(h(n))$.

(c) É verdade que $f(n) + g(n) \in O(h(n))$? Se sim, demonstre. Se não, ofereça um contra-exemplo e verifique que de fato ele é contra-exemplo.

Questão 3 (6 pontos)**.** Nos casos abaixo, uma função $T(n)$ é apresentada de forma recursiva. Encontra, em cada caso, uma função $h(n)$, apresentada de forma não recursiva, tal que $T(n) \in \Theta(h(n))$. Justifique sua resposta apropriadamente, por exemplo usando indução, ou desenhando uma árvore de recorrência e analisando seu custo, ou usando o Teorema Mestre.

(a) $T(n) = 3T(n-2) + 1$. (Você pode assumir que n é ímpar, e que $T(1) = 1$).

(b) $T(n) = 2T(n/4)+T(n/2)+n$. (Você pode assumir que n é potência de 4, e que $T(1) = 1$).

(c) $T(n) = 5T(n/2) + n^2 \log(n)$. (Você pode assumir que n é potência de 2, e que $T(1) = 1$).

Questão 4 (5 pontos)**.** Dada uma permutação de um vetor A com n posições, dizemos que a "(i, j)-transposição" ocorreu se os elementos originalmente nas posições i e j trocaram de lugar.

(a) Seja $X_{i,j}$ a variável aleatória indicadora que, aplicada em uma permutação, é igual a 1 se a (i, j)-transposição ocorreu, e é igual a 0 caso contrário. Determine $\mathbb{E}[X_{i,j}]$ supondo que a permutação é gerada de modo uniformemente aleatório.

(b) Seja X a variável aleatória que conta o número de transposições em uma permutação do vetor A. Determine a relação entre X e os $X_{i,j}$s da letra anterior.

(c) Calcule o número esperado de transposições em uma permutação gerada de modo uniformemente aleatório.

(d) Considere o seguinte algoritmo, que se propõe a gerar uma permutação de uma lista com n elementos.

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 = Random(0,1);
    if j = 1 then
        swap A[i] with A[i + 1];    (assuma A[n + 1] = A[1])
    if j = 0 then
        do nothing;
```

Este algoritmo gera uma permutação de modo uniformemente (pseudo)-aleatório da lista A? Justifique com uma demonstração, ou com um contra-exemplo.

(e) Bonus (2 pontos): qual o número esperado de (i, j)-transposições obtidas ao final da execução do algoritmo Permuta ?

Questão 5 (5 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) executa:
   - 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.

(c) Bonus (1 ponto): Considere a situação inversa, em que duas filas FIFO F_1 e F_2 são usadas para implementar uma pilha LIFO P, de acordo com o seguinte esquema:

1. Push(x, P) executa:
   - Enqueue(x, F_2).
   - Sequencialmente faça y = Dequeue(F_1) e Enqueue(y, F_2), até que F_1 fique vazia.
   - Troque os nomes de F_1 e F_2.
2. Pop(P) executa Dequeue(F_1).

Há custo amortizado para as operações Push e Pop que indique que n dessas operações tem custo limitado em $O(n)$? Justifique.