Universidade Federal de Pernambuco
Centro de Informática
Bach. em Ciência da Computação

IF672cc Algoritmos e Estr. de Dados 2019.2
Prof. Paulo Fonseca

Primeira Prova — 26 de Setembro de 2019

■ QUESTÃO 1 (2,0pt)

Considere o TAD X implementado através de duas pilhas E e S, e com duas operações *fe* e *fs*, como se segue.

Algoritmo *fe*
Entrada te, ts : ponteiros para os topos das pilhas E e S (com sentinela),
v : inteiro
Saída ???
1 $stack\_push(te, v)$
2 **devolva** te, ts
fim

Algoritmo *fs*
Entrada te, ts : ponteiros para os topos das pilhas E e S (com sentinela)
Saída ???
1 **se** $ts \rightarrow next = \perp$ **então**
2 **enquanto** $te \rightarrow next \neq \perp$ **faça**
3 $te, v \leftarrow stack\_pop(te)$
4 $ts \leftarrow stack\_push(ts, v)$
5 **fim faça**
6 **fim se**
7 $ts, v \leftarrow stack\_pop(ts)$
8 **devolva** te, ts, v
fim

a) Ilustre a execução das operações abaixo, representando as duas pilhas E, S após cada operação:

1. *fe* 1	4. *fs*	7. *fs*
2. *fe* 2	5. *fe* 4	8. *fs*
3. *fe* 3	6. *fe* 5	9. *fs*

b) Explique sucintamente (máx 05 linhas) *o que* as operações *fe* e *fs* fazem.

c) Indique o custo assintótico no pior caso da operação *fe*. Justifique sucintamente (máx 03 linhas)

d) Indique o custo assintótico *agregado* de m operações *fs* no pior caso. Justifique sucintamente (máx 05 linhas)

■ QUESTÃO 2 (2,0pt)

O Prof. Sogupa propôs uma variação do Algoritmo Quicksort visto em aula com a seguinte função *qsort*.

Algoritmo *qsort*
Entrada $V = (v_0, \ldots, v_{n-1})$; $0 \leq l \leq r \leq n-1$
1 **se** $r - l > 1$ **então**
2 $p \leftarrow partition(V, l, r)$
3 $pred \leftarrow p$
4 **enquanto** $l \leq pred$ **e** $V[pred] = V[p]$ **faça**
5 $pred \leftarrow pred - 1$
6 **fim faça**
7 $suc \leftarrow p$
8 **enquanto** $suc \leq r$ **e** $V[suc] = V[p]$ **faça**
9 $suc \leftarrow suc + 1$
10 **fim faça**
11 $qsort(V, l, pred)$
12 $qsort(V, suc, r)$
13 **fim se**
fim

Supondo que o pivô escolhido é o elemento mais à esquerda do trecho particionado, responda:

a) Qual o custo assintótico do algoritmo no *pior* caso? Dê um exemplo de entrada de pior caso para $n = 10$.

b) Qual o custo assintótico do algoritmo no *melhor* caso? Dê um exemplo de entrada de pior caso para $n = 10$.

■ **QUESTÃO 3** (2,0pt)

Considere o problema em que, dados um array de inteiros $V = (v_0, \ldots, v_{n-1})$ e um inteiro S, queremos encontrar dois elementos v_i e v_j de V tais que $v_i + v_j = S$. Esse problema pode ser resolvido com auxílio de uma tabela de dispersão (*Hashtable*) da seguinte maneira. Inicie com uma hashtable H vazia. Para cada $i = 0, \ldots, n-1$, procure $d = S - v_i$ em H. Caso encontre, retorne (v_i, d). Caso não encontre, insira v_i em H e continue. Se, após haver inserido todos os elementos de V, ainda não tiver encontrado um par de soma S, é porque tal par não existe.

a) Considerando uma hashtable fechada de tamanho $m = 2n + 1$, com política de resolução de colisões por sondagem linear (*linear probing*), ilustre a execução do algoritmo sobre a entrada

$$V = (28, 54, 25, 60, 14, 44, 91) \quad S = 69,$$

exibindo a tabela após cada inserção realizada.

b) Qual o custo assintótico desse algoritmo

i. No melhor caso? Justifique sucintamente (máx 05 linhas).
ii. No pior caso? Justifique sucintamente (máx 05 linhas).

■ **QUESTÃO 4** (2,0pt)

a) Desenhe a árvore AVL cuja enumeração em pós-ordem é

1, 4, 7, 6, 3, 10, 20, 8.

b) Ilustre as inserções sucessivas dos valores

9 e 5

nessa árvore, representando-a logo após cada inserção (antes das rotações), assim como após todas as rotações necessárias.

■ **QUESTÃO 5** (2,0pt)

Seja $\mathcal{A}$ o conjunto de pares do tipo (S, F), onde S é uma string e F um inteiro positivo. Definimos a relação $\geq$ ("maior ou igual") em $\mathcal{A}$ dizendo que $(S_1, F_1) \geq (S_2, F_2)$ se:

i. $S_1 = S_2$ e $F_1 = F_2$; ou
ii. $F_1 > F_2$; ou
iii. $F_1 = F_2$ e S_1 vem *depois* de S_2 na ordem alfabética.

a) Ilustre a construção *offline* da *max-heap* sobre o vetor de entrada

$$H = \langle (\texttt{C}, 1), (\texttt{D}, 1), (\texttt{B}, 2), (\texttt{R}, 2), (\texttt{A}, 5) \rangle.$$

b) Ilustre passo-a-passo a execução do algoritmo abaixo sobre a max-heap resultante da letra (a), exibindo-a ao final de cada iteração.

Algoritmo *HC*
Entrada H: max-heap sobre $\mathcal{A}$
1 **enquanto** *heap_size*$(H) > 1$ **faça**
2 $\quad (S_1, F_1) \leftarrow$ *heap_extract*(H)
3 $\quad (S_2, F_2) \leftarrow$ *heap_extract*(H)
4 $\quad$ *heap_insert*$(H, (S_1 \cdot S_2, F_1 + F_2))$
5 **fim faça**
fim

No algoritmo acima, $S_1 \cdot S_2$ representa a concatenação das strings S_1 e S_2.

■ ■ ■