Questão 8 - a) Uma fila de prioridade first in first out
A operação Insira sempre insere no fim da fila
e a operação remova sempre extrai o 1º elemento
da fila levando tempo $O(1)$

b) Um Max-Heap, onde a tarefa mais antiga
é a maior e a recém-chegada é a de menor
valor.
A operação Insira busca o elemento no heap
em $O(\log n)$, caso não encontre insere um
novo elemento no Max-Heap que leva t_b tempo
$O(\log n)$.
A operação remova retira o elemento
da folha levando tempo $O(1)$

c) A estrutura é um vetor indexado de 0 a n
onde cada elemento inteiro i será armazenado
em $v[i]$.
A operação insira 1º verifica se $v[i]$ está
ocupado em tempo $O(1)$ e caso não esteja insere
o elemento i em $v[i]$ em tempo $O(1)$.
A operação remova remove o elemento
apontado por j sendo j um apontador para o
último elemento inserido, não precisando
dessa forma procurar uma posição na fila
levando t_b tempo $O(1)$.

Prova 2002/2

Questão 1 - $p-1$ $r=2$ 1 vez $n = r-p+1$
$q-1$ $p-1$ $r=3$ 2 vezes

$$\boxed{\begin{aligned} T(n) &= 0 \quad n=1 \\ T(n) &= T(n-1)+1 \end{aligned}}$$ recorrência

$n-i=1$
$i=n-1$

$$\begin{aligned} T(n) &= T(n-1)+1 \\ &= T(n-2)+1+1 \\ &= T(n-3)+1+1+1 \\ &= T(n-i)+i \\ &= T(1)+n-1 \\ &= n-1 \end{aligned}$$

A fórmula exata é $T(n) = n-1$

Vou provar que a fórmula está certa
Prova por indução em n:
base - $n=1$
$T(1) = 0 = 1-1 = 0$ OK!
passo $n>1$

$$\begin{aligned} T(n) &= T(n-1)+1 \\ &\overset{H.I.}{=} (n-1)-1+1 \\ &= n-1 \quad // \end{aligned}$$

$q(n) = n-1 \leq 1 \cdot n$ para $n \geq 1$ e
$n-1 \geq \frac{1}{2} n$ para $n \geq 1$

Assim concluímos que $T(n) = \Theta(n)$

Questão 2 - Falsa. O algoritmo build-Heap que recebe um vetor $A[1..n]$ e devolve um Heap com n elementos consome tempo $O(n)$ fazendo uma análise cuidadosa.

PanAmericana

Questão 3. Se NP ≠ co-NP significa que NP não é fechada sob complemento e como P é fechada sob complemento $P \subseteq NP \cap co\text{-}NP$ ou seja $P \neq NP$

Questão 4- Se $f(n) = O(n)$
$g(n) = O(n^2)$

A função $g(n)$ cresce mais rápido.
Justificativa
$f(n) \leq c_1 n$ para $n \geq n_0$
$g(n) \leq c_2 n^2$ para $n \geq n_0$

$f(n) \leq c_1 \cdot g(n)$ para $n \geq n_0$
$n \leq 1 \cdot n^2$ para $n \geq 1$

Vou provar que $n \leq 1 \cdot n^2$ para $n \geq 1$
$f(n) \leq n$
$\leq n \cdot n$
$\leq n^2 = g(n)$
Assim $g(n)$ cresce mais rápido que $f(n)$

Questão 5 - Algoritmo
1- Constrói um Max-Heap
2- Extraia os $\lfloor \sqrt{n} \rfloor$ maiores elementos através de EXTRACT-MAX
3- Imprima cada elemento extraído

Análise de tempo	total $O(n)$
1- $O(n)$ - build-heap	
2- $O(\sqrt{n}) \cdot \log n$	
3- $\lfloor \sqrt{n} \rfloor$	

Correção
O algoritmo Build-Heap constrói um Max-Heap onde a raiz possui o maior elemento do Heap
Se a cada extração o heap for reajustado, a raiz permanece com o maior valor garantindo portanto a corretude do algoritmo.

Questão 7- Pg 89
INTERCALA(A)
1- Pegue o 1º vetor A_0 e execute um merge com o vetor crescente A_1 resultando em um vetor parcial $B[1..2^2-1]$ com $2^0+2^1 = 2^1+1$ ou 2^2-1 elementos
2- Execute um merge de $B[1..2^{i+1}-1]$ que é o resultado da operação anterior com o próximo vetor A_{i+1}
3- Execute o passo 2 até A_k
4- Demarca 3

Consumo
Cada merge consome tempo $2^{i+1}-1$

$$\sum_{i=1}^{k} 2^{i+1}-1 = \sum_{i=1}^{k} 2^{i+1} - \sum_{i=1}^{k} 1 = \sum_{i=2}^{k+1} 2^i - k = \sum_{i=1}^{k+1} 2^i \cdot 2 - k$$

$$\frac{2^{k+2}-1}{2-1} - 2 - k = 2^{k+2} - 1 - 2 - k = 2^{k+2} - k - 3 =$$

$$2^{\lfloor \log n+1 \rfloor + 2} - \lfloor \log n+1 \rfloor - 3 = 2^{\log 2} \cdot 2^{\log n+1} - \log n+1 - 3 = 4n - \log n - 3$$

$$4n - (\log n + 3) \leq 4n = O(n)$$

PanAmericana

Questão 8 - ~~Pag 88~~ Pag 108

inserir - O(1)

remover - O(K) sendo K o nº de elementos em D.

O máximo de inserções será de n elementos então a operação remover só poderá mover n elementos da pilha D para a pilha E, portanto no máximo terá um consumo de O(n)

$O(n) + O(n) = 2O(n) = O(n)$

Seja T(n) o custo amortizado de todas as operações para n elementos e $T(n) = O(n)$ então o custo amortizado de cada operação será

$\frac{T(n)}{2} = O(1)$

Questão 6 Não vi Fazer

Árvore B

/

LISTA 2 - ALAIR

Questão 16 - Regra de Horner

$P_n(x) = a_n x^n + \dots + a_1 x^1 + a_0 x^0$

$q_{n-1}(x) = a_n x^{n-1} + \dots + a_2 x + a_1$ e $p_n(x) = x\, q(x) + a_0$

Algo (A, p, r, x)

Se p = r

então se r = 0

devolva A[r]

Se r = 1

devolva A[r].x

i ← 2 v ← x

enquanto i ≤ r

v ← v . x

i ← i + 1

devolva A[r].v

senão q ← ⌊(r+p)/2⌋

devolva Algo(A, p, q, x) + Algo(A, q+1, r, x)

Quantas adições faz?

Sendo n = r - p + 1 adições n - 1

Quantas multiplicações faz?

$\sum_{k=1}^{n} k-1 = \frac{n.(n-1)}{2} = \frac{n^2-n}{2}$

Regra Horner

$q_{n-1}(x) = a_5 x^4 + a_4 x^3 + a_3 x^2 + a_2 x + a_1$

$\sum_{k=1}^{n-2} k - 1 = \frac{1}{2}.(n-2).(n-3)$

$= \frac{1}{2} n^2 - 3n - 2n + 6 = \frac{1}{2} n^2 - \frac{5}{2} n + 6$

$P_n(x) = x.q(x) + a_0$

$= 1 + \frac{1}{2} n^2 - \frac{5}{2} n + 6$ multiplicações

adições

n - 1 adições

O algoritmo faz $\frac{n^2-n}{2}$ enquanto que o método faz $1 + \frac{1}{2} n^2 - \frac{5}{2} n + 6 \leq \frac{n^2-n}{2}$