Errado é só usar Max-Heap-Delete mas para Min-Heap.

Questão 7 - | 1 | 3 | 5 | 7 | 9 | 11 |

BUILDHEAP1(A, i)

1 $x \leftarrow A[i]$

2 $H \leftarrow [1..n] \leftarrow 0$ // eu posso apenas trocar os elementos em A

3 BUILDMAX(A, H, 1, n, 1)

4 REMOVEDEHEAP(H, x)

Questão 7 errada! Veja solução do Jesús.

BUILDMAX(A, H, p, r, j)

1 se $p \le r$

2 então $H[j] \leftarrow A[r]$

3 $q \leftarrow \lfloor r-1+p/2 \rfloor$

4 BUILDMAX(A, H, p, q, 2j)

5 BUILDMAX(A, H, q+1, r-1, 2j+1)

Consumo

BuildHeap1		BuildMax	
Linha		Linha	
1,	$O(1)$	1, 2, 3	$O(1)$
3	$O(\lg n)$	4	$T(\lceil n/2 \rceil)$
4	$O(\lg n)$	5	$T(\lfloor n/2 \rfloor)$

$T(n) = 0 \quad n = 1$

$T(n) = 2T(n/2) + 1$

$= \log n$

O algoritmo BuildHeap1 cria um Max-heap a partir de um vetor ordenado crescentemente e depois remove o elemento A[i].

A rotina Remove de heap é uma operação básica de Heap que remove o elemento e restabelece o Heap tendo um consumo de $O(\lg n)$

24/06/16

Corretude: Invariante $H[1..j]$ é um Max Heap.

Início - Na 1ª chamada $H[1]$ recebe o maior elemento de $A[1..n]$ sendo trivialmente um Max-heap com 1 elemento.

Manutenção: Vou provar que o invariante se mantém na recursão.

$H[j]$ que é próximo índice vazio recebe o maior valor do vetor $A[p..r]$ que é o último já que A está crescentemente ordenado.

As linhas 4 e 5 subdividem o vetor em $A[p..q]$ e $A[q+1..r-1]$ e envia o próximo índice a ser preenchido 2j ou 2j+1 que restabelece o invariante.

Término: A última chamada da recursão será $j = n$ e $H[j]$ receberá elemento de menor valor do vetor $A[n/2+1..n-1]$ caracterizando assim que todos os elementos de H seguem a propriedade $H[i] \le H[\lfloor i/2 \rfloor]$. Portanto, um Max-heap.

Prova 2000/01

Questão 1 - São árvores de busca binária que tem um bit extra que armazena a cor: que pode ser vermelho ou preto. As folhas Nil tem cor preta e a raiz também. Se a raiz for rubra então não é uma árvore rubro-negra.

Perguntar
b)

$n^{\log_2 3 - \epsilon}$

Questão 2 -

a) $T(0) = T(1) = 1$

$T(n) = 3T(\lfloor n/2 \rfloor) + n^2$

Usando o Teorema Mestre temos que

$a = 3$, $b = 2$ e $f(n) = n^2$

Pelo caso 1

$f(n) \leq 1 \cdot n^{\log_2 3 - \epsilon}$ para $\epsilon > 0$ e $n \geq 1$

então $T(n) = \Theta(n^{\log_2 3})$

b) $T(0) = T(1) = 1$

$T(n) = 2T(n-2) + 1$ para n ímpar

Desenrolando a recorrência temos:

$n - 2i = 1$

$n = 2i + 1$

$i = \frac{n-1}{2}$

$T(n) = 2T(n-2) + 1$

$= 2^2 T(n-4) + 2 + 1$

$= 2^3 T(n-6) + 2^2 + 2 + 1$

$= 2^i T(n - 2i) + \sum_{k=0}^{i-1} 2^k$

$= 2^{\frac{n-1}{2}} \cdot 1 + 2^{\frac{n-1}{2}} - 1$

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

Prova por indução em n

base: $n = 1$

$T(1) = 1 = 2^{\frac{1-1}{2}} + 2^{\frac{1-1}{2}} - 1 = 1$

passo: $n > 1$

$T(n) = 2T(n-2) + 1$

$\overset{H.I}{\leq} 2 \cdot (2^{\frac{n-3}{2}} + 2^{\frac{n-3}{2}} - 1) + 1$

$= 2(2^{\frac{n-3}{2}+1} - 1) + 1$

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

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

???? Não sei

c) $T(0) = T(1) = 1$

$T(n) = T(n-1) + n^2$

$n - i = 1$

$i = n - 1$

$T(n) = T(n-1) + n^2$

$= T(n-2) + (n-1)^2 + n^2$

$= T(n-3) + (n-2)^2 + (n-1)^2 + n^2$

$= T(n-i) + \sum_{k=0}^{i-1} (n-k)^2$

$2 - 1 = 1$

$3 - 1 = 2$

0	n-1	n-2
2	1	0

$= T(1) + \sum_{k=0}^{n-2} n^2 - 2nk + k^2$

$= 1 + \sum_{k=0}^{n-2} n^2 - 2n \sum_{k=0}^{n-2} k + \sum_{k=0}^{n-2} k^2$

$= 1 + (n-1) \cdot n^2 - 2n \cdot \frac{1}{2} (n-1) \cdot (n-2) + \frac{1}{6} (n-2) \cdot (n-1) \cdot (2(n-2)+1)$

$= 1 + \cancel{n^3} - n^2 - \cancel{n^3} + 2n^2 + \cancel{n^2} - 2n + \frac{1}{6}(n^2 - n - 2n + 2) \cdot (2n - 3)$

$= 2n^2 - 2n + 1 + \frac{1}{6}(n^2 - 3n + 2) \cdot (2n - 3)$

$= 2n^2 - 2n + 1 + \frac{1}{6} \cdot (2n^3 - 6n^2 + 4n - 3n^2 + 9n - 6)$

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

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

Prova por indução em n

base: $n = 1$

$T(1) = 1 = \frac{1}{3} + \frac{1}{6} + \frac{1}{2} = 1$

passo: $n > 1$

$T(n) = T(n-1) + n^2$

$\overset{H.I}{\leq} \frac{1}{3}(n-1)^3 + \frac{1}{6}(n-1) + \frac{1}{2}(n-1)^2$

$= \frac{1}{3}(n-1) \cdot (n-1) \cdot (n-1) + \frac{n}{6} - \frac{1}{6} + \frac{1}{2}(n^2 - 2n + 1)$

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

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

$6 - 2 \,|\, 2$

$3 - 1 \,|\, 3$

$1 - 1$

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

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

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

A recorrência b) não pode ser limitada por uma função polinomial.

Análise amortizada

Uma sequência de n operações é realizada numa estrutura de dados. A i-ésima operação custa $\sqrt{i}$, se i é um quadrado perfeito, e 1 se caso contrário. Determine o custo amortizado por operação.

$\sqrt{i}$ | se i é quadrado perfeito
1 | caso contrário

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		25
$\sqrt{1}$	1	1	$\sqrt{4}$	1	1	1	1	$\sqrt{9}$	1	1	1	1	1	1	$\sqrt{16}$		$\sqrt{25}$
1			2					3							4		5

$$T(n) = \sum_{i=1}^{n} c_i \leq \sum_{i=1}^{\lfloor\sqrt{n}\rfloor} i + n - \lfloor\sqrt{n}\rfloor$$
$$\leq \sum_{i=1}^{\sqrt{n}} i + n - \sqrt{n}$$
$$= \frac{1}{2}(\sqrt{n}\cdot(1+\sqrt{n})) + n - \sqrt{n}$$
$$= \frac{1}{2}(\sqrt{n}+n) + n - \sqrt{n}$$
$$= \frac{1}{2}\sqrt{n} + \frac{1}{2}n + n - \sqrt{n}$$
$$= \frac{3}{2}n - \frac{1}{2}\sqrt{n}$$
$$\leq \frac{3}{2}n$$

$T(n) = O(\frac{3}{2}n)$

custo amortizado $= \dfrac{O(\frac{3}{2}n)}{n} = O(1)$ constante

Método contábil

Se quadrado perfeito 1,5 reais
caso contrário 0

Se i é quadrado perfeito custa 1 real e 0,5 vai para crédito. Como a quantidade de quadrados perfeitos não é negativa o nº de créditos nunca é negativo

Crédito = custo amortizado − custo real > 0

Custo amortizado $= \frac{3}{2}$ reais

Função potencial

Φ − custo de um nº quadrado perfeito

Sabemos que a distância entre 2 elementos quadrado perfeito é: $(i+1)^2 - (i)^2 = i^2 + 2i + 1 - i^2$
$= 2i+1$

Assim, podemos amortizar o custo de uma operação cara, mediante uma carga constante nas operações baratas.

$$c(2i+1) \geq (i+1)^2$$
$$c \geq \frac{(i+1)^2}{2i+1}$$
$$c \geq \frac{(i+1)^2}{(i+1)^2}$$
$$c \geq \frac{1}{2}\left(\frac{i+1}{i+1}\right)^2 \quad \text{então} \quad c \geq \frac{1}{2}$$

Função potencial

$\Phi(i) = 0$ se $i = 0$

$\Phi(i) = \frac{1}{2}i - \frac{1}{2}\lfloor\sqrt{i}\rfloor^2$ se $i > 0$

Quando $i > 1$ não é quadrado perfeito
$\lfloor\sqrt{i}\rfloor = \lfloor\sqrt{i-1}\rfloor$

$$\hat{c}_i = c_i + \Phi(i) - \Phi(i-1)$$
$$= 1 + \left[\frac{1}{2}i - \frac{1}{2}\lfloor\sqrt{i}\rfloor^2\right] - \left[\frac{1}{2}(i-1) - \frac{1}{2}\lfloor\sqrt{i-1}\rfloor^2\right]$$
$$= 1 + \frac{1}{2}i - \frac{1}{2}\lfloor\sqrt{i}\rfloor^2 - \frac{1}{2}i + \frac{1}{2} + \frac{1}{2}\lfloor\sqrt{i-1}\rfloor^2$$
$$= 1 + \frac{1}{2}$$
$$= \frac{3}{2} \quad \text{constante}$$

/ /

Quando $i \geq 1$ é quadrado perfeito $i = j^2$

$$\hat{c}_i = c_1 - \Phi(i) - \Phi(i-1)$$

$$= \sqrt{i} - \left[\frac{1}{2} i - \frac{1}{2} \lfloor \sqrt{i} \rfloor^2\right] - \left[\frac{1}{2}(i-1) - \frac{1}{2} \lfloor \sqrt{i-1} \rfloor^2\right]$$

$$= \sqrt{i} - \frac{1}{2} i + \frac{1}{2} \lfloor \sqrt{i} \rfloor^2 - \frac{1}{2} i + \frac{1}{2} + \frac{1}{2} \lfloor \sqrt{i-1} \rfloor^2$$

$$= \sqrt{i} - i + \frac{1}{2} + \frac{1}{2} \lfloor \sqrt{i} \rfloor^2 + \frac{1}{2} \lfloor \sqrt{i-1} \rfloor^2$$

/ /

Questão 4 -

Para a opção 1. temos uma busca binária que tem um tempo máximo de $O(\lg n) = \lg 10000 = 13,29$

Análise Amortizada

Vamos supor M. acessos

$$\underbrace{1\ 2\ 3\ 4\ 5\ 6}_{4000}\ \underbrace{7\ 8\ 9\ 10}_{6000}$$

Como todos os clientes estão em um único vetor ordenado

/ /

Questão 5 -

```
1 BuildHeap(A, n)
2   para i = ⌊n/2⌋ decrescendo até 1
3       MaxHeapify(A, n, i)
```

Tamanho da instância = n, consumo $T(n)$

O algoritmo MaxHeapify leva tempo $O(h)$ sendo h a altura da árvore

O loop da linha 2 consome tempo $O(n/2)$

Numa análise grosseira BuildHeap teria consumo $O(n \lg n)$

Numa análise mais cuidadosa, suponha uma árvore completa e de altura h, $n = 2^{h+1} - 1$

$2^h = \frac{1}{2}(n+1)$

altura	cada nó consome	quantos nós
1	1	2^{h-1}
2	2	2^{h-2}
3	3	2^{h-3}
⋮	⋮	⋮
h	h	2^{h-h}

$$\sum_{k=1}^{h} T(k) =$$

$$= 1 \cdot 2^{h-1} + 2 \cdot 2^{h-2} + \dots + h \cdot 2^0$$

$$= (1 \cdot 2^{-1} + 2 \cdot 2^{-2} + \dots + h \cdot 2^{-h}) \cdot 2^h$$

$$= \left(1 \cdot \left(\frac{1}{2}\right)^1 + 2 \cdot \left(\frac{1}{2}\right)^2 + \dots + h \cdot \left(\frac{1}{2^h}\right)\right) \cdot \frac{1}{2}(n+1)$$

$$= \left(\frac{1/2}{(1-1/2)^2}\right) \cdot \frac{1}{2}(n+1)$$

$$= 2 \cdot \frac{1}{2}(n+1) = (n+1)$$

$$= O(n).$$

NOME:

E-MAIL

MATÉRIA

PROFESSOR

SALA

HORÁRIOS

HORÁRIO	SEGUNDA	TERÇA	QUARTA	QUINTA	SEXTA	SÁBADO

PROVAS/NOTAS

TRABALHOS

ANOTAÇÕES

Questão 6 - $k = r - p + 1$

Merge (n, p, r, A)
  se $p = r$
    devolva A_1
  se $p < r$ $q \leftarrow \lceil r+p/2 \rceil$
    devolva INTERCALA (Merge(A, p, q), , Merge(A, q+1, r))

$T(n, k) = 1$ para $k = 1$
$T(n, k) = 2T(k/2, n) + n$ para $k \geq 2$
$= 2^2 T(k/4, n) + n + n$
$= 2^3 T(k/8, n) + n + n + n$
$= 2^i T(k/2^i, n) + i n$
$= k \cdot 1 + \log k \cdot n$
$= O(n \log k)$

Questão 7 - o loop é controlado por a e b

Para $n = 3$

1	1	0	1	2	3
2	2	1	1	1	1

$\underbrace{1 \quad 2}_{p>0} \; \underbrace{3 \quad 4}_{p<0} \; \underbrace{5 \quad 6}_{p>0}$

A soma é incrementada independentemente de p
O máximo que se consegue é incrementar n-1 vezes a em seguida decrementá-lo n-1 vezes controlado por b e novamente incrementado n vezes totalizando $n-1 + n-1 + n = 3n-2$ vezes.

Seqüência
$n-1$ - $p>0$
$n-1$ $p<0$
n $p>0$

Questão 8 - $\{x_1, x_2, \ldots x_n\}$

Recorrência
$c[i, j] \leftarrow 0$ se $i = 0$ ou $j = 0$ ou $i = j$
$c[i, j] \leftarrow c[i-1, j] + 1$ se $x_j - x_i = 1$ e $j - i \neq 1$
$c[i, j] \leftarrow c[i-1, j]$ se $x_j - x_i \neq 1$

$c[i, j] \leftarrow 0$
$c[$

$x_1 \; x_2 \; x_3 \; x_4$

Pontos(X, n)
Para $i = 0$ até n
  $c[i, 0] \leftarrow 0$
  $c[0, i] \leftarrow 0$
  $c[i, i] \leftarrow 0$
Para $l \leftarrow 2$ até n
  Para $i \leftarrow 1$ até $n - l + 1$
    $j \leftarrow i + 1$
    se $x_j - x_i = 1$ e $j - i \neq 1$
      então $c[i, j] \leftarrow c[i-1, j] + 1$ $O(n^2)$
        $K[v, 0] \leftarrow X_i$
        $K[v, 1] \leftarrow X_j$
        $v \leftarrow v + 1$
      senão $c[i, j] \leftarrow c[i-1, j]$

$x_1 \; x_4$
$x \leq x$

Outra idéia Guloso
Ordena pontos não-decrescentemente $O(n \lg n)$
$D \leftarrow x_1$
Para $i \leftarrow 2$ até n
  se $x_i - D > 1$
    então $K[v, 0] \leftarrow D$
      $K[v, 1] \leftarrow X_{i-1}$
      $v \leftarrow v + 1$
      $D \leftarrow x_i$
devolva K