Código de honra e conduta discente:
Eu, Lucas Mateus Fernandes , matrícula 0035411, prometo pela minha honra que fui honesto e não trapaceei nessa avaliação passando ou recebendo cola.

Formiga, MG, 11 de março de 2021.

Devido a verbosidade da resolução da prova e os debates levantados durante a ultima aula estou enviando o arquivo em forma digital.

OBS:Estou sem nota na prova 1, Enviei no enunciado prova1, porfavor verifique final arq.

Questão 1)

Há duas formulas a serem analisadas

- $A(n) = 5A(n/2)+n^2$
- $B(n) = kB(n/8) + n^2$

Usando o teorema mestre há 3 possíveis resultados na seguinte ordem de crescimento

- $Teta(n^k) \rightarrow Teta(n^k * \log n) \rightarrow Teta(n^{\log_b a})$

Em A tem se que (a=5 b=2 c=1 k=2) e como $a>b^k$ aplicasse tal regra e chegamos a $Teta(n^{\log_2 5})$

Em B tem se que (a=K b=8 c=1 k=2) ou seja há 3 possibilidades 'a' pode ser 63 ou 64 ou um valor maior que 64:

$a<b^k \rightarrow a<64$
$a=b^k \rightarrow a=64$
$a>b^k \rightarrow a>64$

Como estou procurando o maior k entrarei na regra $a>b^k$ ou seja a>64 Para achar um valor de 'a' tal que B(n) < A(n) é o equivalente a achar um valor de k tal que $\log_8 K < \log_2 5$ ou seja é necessário achar o valor de K tal que $\log_8 K = \log_2 5$

$\log_8 K = \log_2 5$
$8^{\log_2 5} = k$
$125 = k$

Como quero um valor que faça B(n) ser menor então é necessário um valor menor que 125 e o maior inteiro menor que 125 é 124.

K = 124

Questão 2 a) $T(n) = 10T(n/2) + n^3$

De acordo com o teorema mestre a:10 b:2 c:1 k:3

$b^k \rightarrow 2^3 \rightarrow 8$

- 10 > 8 True
- 10 = 8 Falso
- 10 < 8 Falso

Portanto entra na regra 1

$\Theta(n^{\log_2 10})$

Questão 2 b) $T(n) = T(n - 1) + 5$

Fazendo pelo método da substituição:

$T(n - 1) + 5$	$\rightarrow$	$T(n - 1) + 5*1$
$[T(n - 2) + 5]+5$	$\rightarrow$	$T(n - 2) + 5*2$
$[T(n - 3) + 5]+5+5$	$\rightarrow$	$T(n - 3) + 5*3$
	.	
	.	
	.	
$[(n-(n-3))+5)] +(n-3)*5$	$\rightarrow$	$(n-(n-3)) + 5*(n-3)$
$[(n-(n-2))+5)] +(n-2)*5$	$\rightarrow$	$(n-(n-2)) + 5*(n-2)$
$[(n-(n-1))+5)] +(n-1)*5$	$\rightarrow$	$(n-(n-1)) + 5*(n-1)$

Ou seja a cada iteração reduz 1 do n portanto o T(x) sera chamado 'n-1' vezes parando no caso base onde x=1 e a cada chamada de iteração faz 5 operações ou seja

T(n) :

Se n = 1 então 1

Se n != 1 então ((n-1)*5) + T(1)

que equivale a

Se n != 1 então 5n-4

$\Theta(5n-4) \rightarrow \Theta(n)$

Questão 2 c) $T(n) = 2T(n/2) + (\log n) * n$

De acordo com o teorema mestre a:2 b:2 c:(log n) k:1

$b^k \rightarrow 2^1 \rightarrow 2$

- 2 > 2 Falso
- 2 = 2 True
- 2 < 2 Falso

Portanto entra na regra 2

$\Theta(n^1 * \log n)$

Questão 2 d) $T(n) = 2T(n/4) + T(n/2)$

Fazendo a expansão telescopia temos:

$T(n/2^0) = 2T(n/2^2) + T(n/2^1)$
$T(n/2^2) = 2T(n/2^4) + T(n/2^3) + T(n/2^1)$
$T(n/2^4) = 2T(n/2^8) + T(n/2^5) + T(n/2^3) + T(n/2^1)$
$T(n/2^8) = 2T(n/2^{16}) + T(n/2^9) + T(n/2^5) + T(n/2^3) + T(n/2^1)$
$T(n/2^{16}) = 2T(n/2^{32}) + T(n/2^{17}) + T(n/2^9) + T(n/2^5) + T(n/2^3) + T(n/2^1)$
$T(n/2^{32}) = 2T(n/2^{64}) + T(n/2^{33}) + T(n/2^{17}) + T(n/2^9) + T(n/2^5) + T(n/2^3) + T(n/2^1)$
.
.
.
$T(n/2^{\log_2 n/2}) = 2T(n/2^{\log_2 n/4}) + T(n/2^{\log_2 n/4}) \ldots + T(n/2^9) + T(n/2^6) + T(n/2^3) + T(n/2^1)$
$T(n/2^{\log_2 n}) = 1 + T(n/2^{\log_2 n/2}) + T(n/2^{\log_2 n/4}) \ldots + T(n/2^9) + T(n/2^6) + T(n/2^3) + T(n/2^1)$

Questão 3 a)

- Estou assumindo que o range(N) é uma instrução linear que gasta N instruções
- Estou assumindo no for também é uma instrução linear que gasta N instruções

Portanto um for i in range(N) gasta 2*N que assintoticamente é O(N)

Análise linha a linha:

Linha7: Atribuição gasta 1 e a multiplicação gasta 1 que assintoticamente é O(1)
Linha6: Atribuição gasta 1 e a soma gasta 1 que assintoticamente é O(1)
Linha5:Vai de 1 até N porem k dobra a cada instancia logo é O($\log_2 N$)
Linha4: Atribuição é uma constante que assintoticamente é O(1)
Linha3:Parse para int gasta 1 e o for rodara n/3 vezes que assintoticamente é O(N)
Linha2:O for rodara N vezes que assintoticamente é O(N)
Linha1:Rodara 1 vezes que assintoticamente é O(1)

T(N) equivale a $((((((2+2)*(\log_2 N))+1)*(n/3)))*n)$
Portanto o algorítimo inteiro tem um custo que assintoticamente é: O($n^2 * \log_2 N$)

Questão 3 b)

- Estou assumindo que o N é limitado ou seja 0<=N<10 ou seja não pode ser analisado tendendo ao infinito.
- Estou assumindo que o range(N) é uma instrução linear que gasta N instruções.
- Estou assumindo que random.sample(N) gasta N instruções.
- Estou assumindo que random.choice gasta 1
- Estou assumindo que escolhidos é uma lista duplamente encadeada ou seja append gasta 1

M é o tamanho do vetor definido na linha 2
N é o valor escolhido na linha 1

Linha9: Gasta 1 que é apenas atribuir no final da fila encadeada
Linha8: Gasta no máximo M
Linha7: Desconsidero o else pois será contabilizado no ‘if’
Linha6: Gasta 2 (calcular o valor e atribuir no final da fila encadeada)
Linha5: Gasta 2 (comparar e fazer o ‘go to’)
Linha4: Gasta no máximo N
Linha3: Gasta 2 (atribuição mais choice)
Linha2: Gasta (M+1) + N + 1 (range,sample,atribuição)
Linha1: Gasta (N+1) + 2 (range,choice,atribuição)

T(N) equivale a ((1)*M)+(2*N)+2+((M+1) + N + 1)+((N+1) + 2)
que assintoticamente equivale a: O (M+N)

Porem como N é limitado a [0,10[e M sé limitado a [0,100[temos que o big O é :
O (198+36+7) → O (241) ou seja é uma constante

Questão 3 c)

- Estou assumindo que o range(N) é uma instrução linear que gasta N instruções
- Estou assumindo no for também é uma instrução linear que gasta N instruções portanto um for i in range(N) gasta 2*N que assintoticamente é O(N)

Linha6: Gasta 1
Linha5: T(N/2) + 2 (recursão, soma, atribuição)
Linha4: O(n) (laço de repetição linear)
Linha3: O(1)Verificação da condição de parada
Linha2: O(1)Atribuição
Linha1 : T(N)

Aplicando o teorema mestre em " T(N/2) + 2 " temos as seguintes variáveis a:1 b:2 c:2 k:0
$b^k \rightarrow 2^0 \rightarrow 1$

- 1 > 1 Falso
- 1 = 1 Verdadeiro
- 1 < 1 Falso

Portanto entra na regra 2 → $\Theta(n^0 * \log n) \rightarrow \Theta(\log n)$

Teoricamente a linha 5 gasta assintoticamente O(log N)
portanto a função M(n) equivale a (1 + (n*(log(n)) +1 +1)

Portanto o algorítimo inteiro tem um custo assintoticamente de: O(n*log N)

Questão 3 d)

Linha6: Gasta 1
Linha5: T(N/2) + 2 (recursão, soma, atribuição)
Linha4: Gasta 3 (divisão, parse, atribuição)
Linha3: $O(\log_2 n)$ Até chegar na condição de parada pois a cada iteração n cai metade
Linha2: Gasta 1 (Atribuição)
Linha1 : T(N)

T(n) :

Se n == 1 então custo é 3
Se n >= 1 então $((1+(((T(n/2)+2)+3))*\log_2 n)+1)$

Usando a substituição na linha 5 temos que:
$T(N/2^0) \rightarrow T(N/2^1) + 2 \rightarrow T(N/2^2) + 3 \rightarrow T(N/2^3) + 4 \rightarrow \ldots\ldots \rightarrow T(N/2^{\log_2 n}) + (1 + \log_2 n)$

$T(N/2^{\log_2 n})$ equivale a T(1) que é o caso base

Então a linha 5 equivale a $((T(1) + (1+\log_2 n)) + 2)$

T(n) :

Se n == 1 então custo é 3

Se n >= 1 então $((1+((((3 + (1+\log_2 n)) + 2) + 3)) * \log_2 n) + 1)$

Assintoticamente T(n) :

Se n == 1 então custo é 3

Se n >= 1 então $(\log_2 n)^2$

OBS: Estou sem nota na prova 1, Enviei no enunciado prova1, porfavor verifique