Prova 2

Total points 18/20

As justificativas das questões de múltipla escolha e as respostas das questões dissertativas devem ser enviadas em um documento separado (formato PDF).

The respondent's email (**henrique.padula@ufv.br**) was recorded on submission of this form.

Considere o problema de ordenação, onde as listas (de tamanho n) a serem ordenadas possuem $\lfloor n/2 \rfloor$ valores iguais a um número real x e $\lceil n/2 \rceil$ valores iguais a um outro número real y (estes números estão distribuídos aleatoriamente na lista). Assinale as afirmativas verdadeiras. Justifique suas respostas (em uma folha separada). *2/4

I) No pior caso, o algoritmo *quicksort* (com a partição de *Hoare*) ordenará esse tipo de listas em tempo $O(n \log_2 n)$.

II) Essas listas podem ser ordenadas por um algoritmo de tempo $O(n)$ (diga qual seria esse algoritmo).

III) No melhor caso, o algoritmo *mergesort* ordenará esse tipo de listas em tempo $O(n)$.

IV) No pior caso, o algoritmo *quicksort* (com a partição de *Hoare*) ordenará esse tipo de listas em tempo $O(n^2)$.

- ○ I e II
- ○ III e IV
- ◉ II e IV
- ○ I e III
- ○ Somente IV

Individual feedback

Esse tipo de listas corresponde ao melhor casso do Quicksort. A partição de Hoare sempre dividirá a lista em partes iguais (sublistas balanceadas).

Para um array A[1 .. n] de n números inteiros, o que determina (retorna) o seguinte algoritmo? Qual é a complexidade do algoritmo? Justifique suas respostas (em uma folha separada). *4/4

```
ALGORITMO(A[i . . r])
#Entrada: um subarray de A[1 . . n], onde i e r são os índices inicial e final.
    if i == r : return A[i]
    else:
        m = ⌊ (i + r)/2 ⌋
        a = ALGORITMO (A[i . . m])
        b = ALGORITMO ( A[m+1 . . r])
        if a > b :   return b
        else: return  a
```

- ○ Determina o menor número do array. A complexidade do algoritmo é O(n log2 n).
- ○ Determina a mediana. A complexidade do algoritmo é O(n log2 n).
- ◉ Determina o menor número do array. A complexidade do algoritmo é O(log2 n).
- ○ Determina o maior número do array. A complexidade do algoritmo é O(n log2 n).
- ○ Determina o maior número do array. A complexidade do algoritmo é O(log2 n).

Individual feedback

A complexidade é O(n), pois T(n) = 2T(n/2) + 1

Relacione a coluna da esquerda com a coluna da direita com base nos métodos de projeto de algoritmos. *

	(A) Divide o problema em partes menores e combina sua solução em uma solução global.	(B) Geralmente testa todas as possibilidades para se encontrar uma solução. É uma abordagem baseada diretamente na definição do problema e nos conceitos envolvidos.	(C) Calcula a solução para subproblemas, dos problemas menores para os maiores, armazenando os resultados parciais durante o processo, reutilizando-os assim que possível.	(D) O método sugere a construção de uma solução através de uma sequência de passos. Em cada passo escolhe o item/elemento mais atrativo que vê pela frente para fazer parte da solução atual.	Score
(II) Divisão e conquista	◉	○	○	○	1/1
(I) Força bruta	○	◉	○	○	1/1
(III) Programação dinâmica	○	○	◉	○	1/1
(IV) Método guloso	○	○	○	◉	1/1

Para esta questão, o item a) deve ser respondido neste formulário. As respostas dos outros itens devem ser feito em uma folha separada. *4/4

Aplicando Programação Dinâmica (PD) deseja-se determinar o elemento de menor valor de uma lista L[1..n] com n elementos não ordenados.

a) Apresente uma **relação de recorrência** para determinar o elemento de menor valor de forma *bottom-up*.

b) Para a lista L[1..8] = {10, 8, 4, 2, 20, 12, 6, 5}, monte a tabela utilizada pelo método de PD e mostre como é obtida elemento de maior valor.

c) Escreva o **pseudocódigo** do algoritmo de PD e apresente sua **complexidade**.

menor_valor[i] = min(menor_valor[i-1], L[i]), com menor_valor[0] = "∞"

Individual feedback

Se a lista tem um único elemento (n=1), então menor_valor[1] = L[1] (o único elemento).

Para esta questão, o item a) deve ser respondido neste formulário. As respostas dos outros itens devem ser feito em uma folha separada. *4/4

Dada uma barra de metal de comprimento n metros, deseja-se cortar esta barra em pedaços menores de comprimentos $p_1, \dots, p_m$ (inteiros); sendo que p_1 = 1m e $p_i < n$, $\forall i \geq 2$. O objetivo é obter o menor número de pedaços. Note que, a pior solução é cortar a barra em n pedaços, cada pedaço de comprimento 1 metro.

a) Escreva uma **estratégia gulosa** para cortar a barra obtendo o menor número de pedaços.

b) Aplique sua estratégia para resolver a seguinte instância:
Comprimento da barra $n = 20$, comprimento dos possíveis pedaços: $p_1 = 1$, $p_2 = 7$, $p_3 = 5$, $p_4 = 2$, $p_5 = 4$.

c) Escreva o pseudocódigo do seu algoritmo guloso. O algoritmo deve determinar/imprimir as barras menores obtidas no corte. Calcule a **complexidade** do algoritmo.

Cortar sempre o maior pedaço possivel, desde que esse nao ultrapasse o tamanho atual da barra