Algoritmos e Estruturas de Dados II
Prova P1 – 2022/2
07/12/2022

Nome completo: _____

Matrícula:_____

Questão 1 (1 ponto)
Resolva as seguintes recorrências:

a) $T(n) = 81T\left(\frac{n}{9}\right) + n^4 \log n$

Resposta: $T(n) = \Theta(n^4 \log n)$ pelo Caso 3 do Teorema Mestre, já que $n^{\log_b a} = n^2$ e $f(n) = n^4 \log n$.

b) $T\ (n) =\ 4T\left(\frac{n}{2}\right) + \log n$

Resposta: $T(n) = \Theta(n^2)$ pelo Caso 1 do Teorema Mestre, já que $n^{\log_b a} = n^2$ e $f(n) = \log n$.

Critério de correção: 0,5 pontos para cada item com resposta correta.

Questão 2 (2 pontos)
Escreve um algoritmo de Divisão e Conquista para determinar se todos os elementos em um array fornecido são iguais. Seu algoritmo deve retornar verdadeiro, se todos os elementos forem iguais, ou falso, caso contrário.

Resposta:

```
boolean iguais(int A[], int ini, int fim){
   if (ini == fim)
      return true;
   if (A[ini] != A[fim])
      return false;
   int meio = (ini+fim)/2;
   return iguais(A, ini, meio) && iguais(A, meio+1, fim);
}
```

A chamada inicial seria iguais (A, 0, A.length-1).

Critério de correção: 2 pontos para o algoritmo correto usando divisão e conquista e 1 ponto para algoritmo que resolve o problema sem uso de divisão e conquista.

Questão 3 (2 pontos)

Marque V ou F para cada uma das seguintes afirmações e explique brevemente o porquê. Quanto melhor o seu argumento, maior a sua nota. Sua justificativa vale mais pontos do que sua designação de verdadeiro ou falso.

a) ( ) Verdadeiro ( ) Falso

Dado um conjunto de inteiros, o número de comparações necessárias para encontrar o elemento máximo é $n-1$ e o número de comparações necessárias para encontrar o elemento mínimo é $n-1$. Portanto, o número de comparações necessárias para encontrar simultaneamente o menor e o maior elemento é $2n-2$.

Resposta: Falso. Na aula nós vimos um algoritmo que usa $\frac{3n}{2}$ comparações para encontrar simultaneamente o menor e o maior elemento.

c) ( ) Verdadeiro ( ) Falso

Uma implementação de um problema de programação dinâmica usando a abordagem *top-down* não produz nenhuma melhoria assintótica (para qualquer instância do problema) em tempo de execução em relação a uma solução iterativa.

Resposta: Falso. Em alguns casos a abordagem *top-down* pode permitir que a solução "pule" certos subproblemas.

Critério de correção: para cada item, 0,3 pontos para a escolha correta de verdadeiro ou falso e até 0,7 pontos para a justificativa correta.

Questão 4. (2 pontos)

Quais são os princípios básicos dos algoritmos de programação dinâmica? Ou seja, quais características eles compartilham?

Resposta: Algoritmos de programação dinâmica geralmente empregam os seguintes princípios:

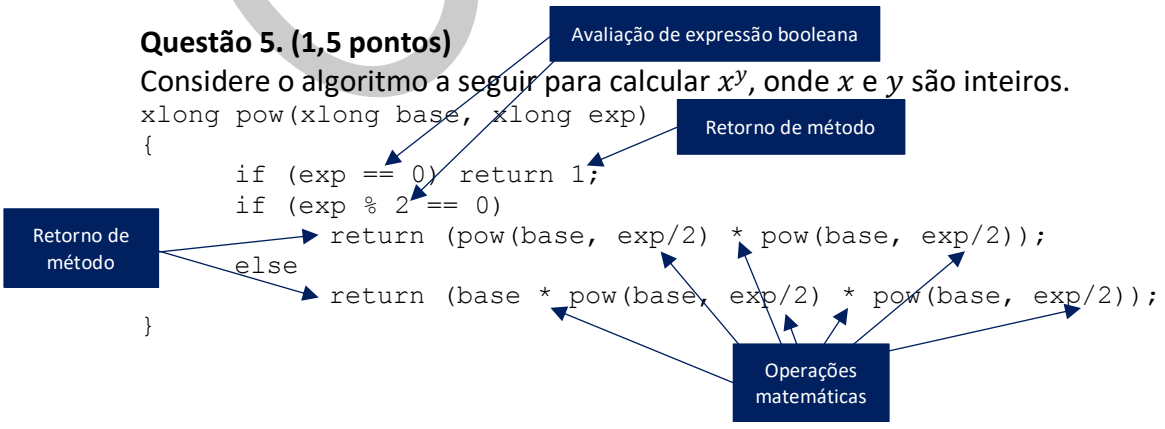
1. Subestrutura ótima: soluções ótimas para problemas são construídas a partir de soluções ótimas para subproblemas.
2. Exploração em torno de uma configuração: para determinar a melhor opção, experimente todas e selecione a melhor. Não emprega heurística.
3. Armazenamento das soluções: as respostas aos subproblemas são lembradas para evitar cálculos repetidos da mesma coisa.

Critério de correção: pontuação proporcional a completude da resposta.

Questão 5. (1,5 pontos)

Considere o algoritmo a seguir para calcular x^y, onde x e y são inteiros.

```
xlong pow(xlong base, xlong exp)
{
      if (exp == 0) return 1;
      if (exp % 2 == 0)
            return (pow(base, exp/2) * pow(base, exp/2));
      else
            return (base * pow(base, exp/2) * pow(base, exp/2));
}
```



a) Qual o paradigma utilizado no algoritmo?

Resposta: Divisão e Conquista

b) Identifique as operações primitivas realizadas pelo algoritmo?

Resposta: Marcadas no algoritmo

Critério de correção: 0,7 pontos no item a) e até 0,8 pontos no item b).

Questão 6. (1,5 pontos)
Considere o problema da mochila visto em aula que recebe como entrada uma mochila com capacidade de até C quilos e n itens a serem adicionados à mochila. Cada item i tem um peso w_i e um valor v_i, com w_i e v_i sendo inteiros. O objetivo é encontrar o valor máximo que não ultrapasse o limite C de peso da mochila. Um algoritmo que resolve o problema é mostrado a seguir.

```
int mochila(int C, int n){
     if(memo[w][n] != -1)
         return memo[w][n];
     if (w ==0 || n==0){
         return memo[w][n] = 0;
     if (peso[n-1] > C){
         return memo[w][n] = mochila(C, n-1);
     return memo[w][n] =
        max(mochila(C, n-1), valor[n-1] + mochila(C - peso[n-1], n-1));
}
```

Mostre o preenchimento da matriz a seguir realizado pelo algoritmo com a mochila tendo capacidade $w = 12$ e os 3 (três) itens a serem transportados: $O(w, v) = \{(4, NLPN), (3, NLUS), (5,9)\}$, onde NLPN é o número de letras do seu primeiro nome e NLUS é o número de letras do seu último sobrenome. Mostre a solução dada pela matriz para $w = 12$.

	0	1	2	3	4	5	6	7	8	9	10	11	12
1													
2	0												
3	0												

Resposta: ANULADA. No algoritmo, o parâmetro de entrada C deveria ser w.

Critério de correção: Todos os alunos que fizeram a prova receberão 1,5 pontos para essa questão.