Universidade Estadual do Oeste do Paraná – UNIOESTE
Campus de Cascavel
Colegiado de Ciência da Computação

Projeto e Análise de Algoritmos
Lista 2 – 14/12/2021

1) Dado o código fonte a seguir encontre:
a) A sua função de recorrência
b) A sua complexidade assintótica. Para tanto, adote o método da substituição.

```
int Peso (int *V, int ini, int fim)
{
        int i, P=0;
        for(i=ini;i<=fim;i++)
                P += V[i];
        return P;
}
```

```
int Falsa (int *V, int ini, int fim)
{
        if (ini == fim)
                return ini;
        else
        {
                int meio = (ini+fim)/2;
                if (Peso(V,ini,meio) < Peso(V,meio+1,fim))
                        return Falsa(V,ini,meio);
                else
                        return Falsa(V,meio+1,fim);
        }
}
```

2) Resolva as seguintes recorrências adotando o **teorema mestre** e determine qual a complexidade assintótica das funções. Deve constar na resposta em qual caso do teorema a recorrência se encaixa.

a) $$\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 3T\left(\frac{n}{3}\right) + 12n + 3 & \text{para } n > 1 \end{cases}$$

b) $$\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 4T\left(\frac{n}{2}\right) + 4n & \text{para } n > 1 \end{cases}$$

c) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 3T\left(\frac{n}{4}\right) + n & \text{para } n > 1 \end{cases}$

d) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 2T\left(\frac{n}{2}\right) + 7n + 2 & \text{para } n > 1 \end{cases}$

e) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 2T\left(\frac{n}{2}\right) + 3n^2 - 12n + 2 & \text{para } n > 1 \end{cases}$

3) Considere os seguintes algoritmos recursivos que resolvem o mesmo problema em uma entrada de tamanho n:
Algoritmo 1: Divide o problema em 3 partes de tamanho n/4 cada e gasta um tempo adicional de O(1) por chamada.
Algoritmo 2: Divide o problema em 3 partes de tamanho n/2 cada e gasta um tempo adicional de O(n^2) por chamada.
Algoritmo 3: Divide o problema em 3 partes de tamanho n/3 cada e gasta um tempo adicional de O(n) por chamada.
A complexidade dos algoritmos 1, 2 e 3 é, respectivamente:

A) $\Theta(n^{log_4 3}), \Theta(n^2), \Theta(n\ log\ n)$

B) $\Theta\left(\frac{n}{4}\right), \Theta\left(\frac{n}{2}\right), \Theta\left(\frac{n}{3}\right)$

C) $\Theta(1), \Theta(n^2), \Theta(n)$

D) $\Theta(n^4), \Theta(n^2), \Theta(n^3)$

E) $\Theta(n^{log_4 3}), \Theta(n^{log_2 3}), \Theta(n^{log_3 3})$

4) Resolva as seguintes recorrências através do método iterativo e determine qual a complexidade assintótica das funções.

a) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = T(n-1) + 4 & \text{para } n > 1 \end{cases}$

b) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = T(n-1) + n & \text{para } n > 1 \end{cases}$

c) $\begin{cases} T(n) = 1 & \text{para } n = 1 \\ T(n) = 2T\left(\frac{n}{2}\right) + 7n + 2 & \text{para } n > 1 \end{cases}$

5) O tempo de execução T(n) de um algoritmo, em que n é o tamanho da entrada, é dado pela equação de recorrência $T(n) = 8T(n/2)+q*n$ se $n > 1$. Dado que $T(1) = p$, e que p e q são constantes arbitrárias, a complexidade do algoritmo é:

a) $O(n)$.
b) $O(n \log n)$.
c) $O(n^2)$.
d) $O(n^3)$.
e) $O(n^n)$

6) O algoritmo ALGSORT ordena vetores de números inteiros distintos usando apenas comparações. Nesse algoritmo, a função menor(V, i, j) retorna o índice l, tal que V[l] é o menor número no vetor V[i..j]. O custo de tempo de pior caso de menor(V, i, j) é igual a $j - i$ comparações. De forma similar, a função maior(V, i, j) retorna um índice g, tal que V[g] é o maior número no vetor V[i..j], também com custo de execução de $j - i$ comparações no pior caso. Para ordenar o vetor X[1..n], ALGSORT (V, i, j) é chamado com os parâmetros, V = X, i = 1 e j = n.

```
ALGSORT (V,i,j);
(1)     Se j-i=0 então
        retorne;
(2)     Se j-i=1 então
                Se V[j] < V[i] então
                        Troque(V[j],V[i]);
                Fim;
                retorne;
        Fim;
(3)     l = menor(V,i,j);
(4)     Troque(V[i],V[l]);
(5)     g = maior(V,i,j);
(6)     Troque(V[j],V[g]);
(7)     ALGSORT (V,i+1,j-1);
```

A função que caracteriza o custo de tempo de pior caso, T(n), para a chamada ALGSORT (X, 1, n) é dada por:

a) $T(n) = T(n-1) + 2n - 2$
b) $T(n) = T(n-2) + 2n - 2$
c) $T(n) = T(n-2) + n - 1$
d) $T(n) = T(n-2) + (n - 1)^2$
e) $T(n) = T(n/2) + 2n$

7) Através do método interativo, determine a complexidade assintótica do algoritmo ALGSORT.

8) Obtenha a complexidade assintótica do algoritmo a seguir utilizando a árvore de recursão

```
void Imprime (int *V, int tam, int ini, int fim)
{
        int i;
        if (tam == 1)
                printf(V[ini]);
        else
        {
                for(i=0;i<tam;i++)
                        printf("%d",V[i]);
                int meio = (ini+fim)/2;
```

```
            Imprime(V,tam/2,ini,meio);
            Imprime(V,tam/2,meio+1,fim);
      }
}
```