PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Exatas e Informática
1ª Avaliação

Curso : *Engenharia de Software*
Disciplina : *Algoritmos e Estruturas de Dados II*
Professora : *Eveline Alonso Veloso*
Valor : *20 pontos*
Data : *10/04/2024*

Nome: Luca Ferrari Agalin Nota: 19,0

1) (2 pontos) O estudo de algoritmos envolve dois aspectos básicos: correção e análise.

- Correção: exatidão do método empregado (prova matemática).
- Análise: avaliar a eficiência do algoritmo em termos dos recursos (memória e tempo de execução) utilizados.

Por isso, uma habilidade necessária ao projetista de *software* é saber contar o número de operações realizadas pelo algoritmo. O trecho de código abaixo realiza algumas operações.

```
if (n > a + 5) && (n < b + 4)
    l *= 5;
else {
    l *= 2;
    k *= 3;
    m *= 5;
}
for (int i = 0; i < n; i++) {
    for (int j = 0; j < (n - 1); j++) {
        l = (a * 2) + (b * 5) + (c * 4);
    }
}
```

n
n-1
3
n·(n-1)·3
3n(n-1)
3n² - 3n

Considerando o trecho de código acima, assinale a opção que apresenta sua função de complexidade *f*(*n*), para o pior caso, considerando o número de operações de multiplicação realizadas.

a) $n(n - 1) + 3$
b) $3n(n - 1) + 3$
c) $3n(n - 1)$
d) $n^2 + 3$
e) $3n^2$

2) (3 pontos) Para criar ou utilizar um algoritmo é importante determinar seu custo computacional. O trecho de código abaixo realiza algumas operações.

```
int x = 2;
for (int i = n; i > 0; i--) {
      for (int i = n; i > 0; i /= 2) {
            x++;
      }
}
```

Considerando o trecho de código acima, assinale a opção que apresenta sua função de complexidade $f(n)$, considerando o número de operações de soma realizadas.

a) $n * (\lfloor \log_2 n \rfloor + 1)$
b) $n * (\log_2 (n) + 1)$
c) $n * \lceil \log_2 n \rceil$
d) $n * (\lfloor \log_2 n \rfloor - 1)$
e) $\log_2 (n) + 1$

Valor de n	Entra em	
4	4, 2, 1	$\log(n)+1$
8	8, 4, 2, 1	
3	3, 1	$\lfloor \log(n) \rfloor + 1$
9	9, 4, 2, 1	

3) (2 pontos) Considere um arranjo unidimensional X, contendo milhares de números inteiros não ordenados; e um algoritmo que faça a contagem dos números iguais a zero presentes em X, da forma mais eficiente possível. Assuma ainda que a função de complexidade de tempo para o número de comparações desse algoritmo seja $f(n)$.

Leia e analise as afirmativas a seguir:

I. $f(n) = n^2$. F
II. $f(n)$ é $\Theta(n)$. V
III. $f(n) = O(n^2)$. V
IV. $f(n)$ é $\Omega(n^2)$, pois n^2 é um limite superior para $f(n)$. F

É correto o que se afirma apenas em:

a) I e II
b) I e IV
c) I, III e IV
d) II e III
e) II, III e IV

4) (2 pontos) Em computação, ordenar é o ato de se colocar os elementos de uma sequência de informações, ou dados, em uma ordem predefinida.
Um estudante necessita ordenar um vetor de 5 posições. Considere que os percentuais foram inseridos nesse vetor, na seguinte sequência: 25.33, 27.72, 27.10, 26.90 e 27.31. Ao ordenar esse vetor, o método desenvolvido pelo estudante realizou os seguintes passos no processo de ordenação:

Passo 1:	25.33	27.72	27.10	26.90	27.31
Passo 2:	25.33	27.10	27.72	26.90	27.31
Passo 3:	25.33	26.90	27.10	27.72	27.31
Passo 4:	25.33	26.90	27.10	27.31	27.72

Trata-se do método de ordenação:

a) *Bubble Sort*.
b) *Selection Sort*.
c) *Insertion Sort*.
d) *Quicksort*.
e) *Heapsort*.

Respostas:

9,0

1	2	3	4
B ✓	A ✓	D ✓	C ✓

5) (5 pontos) Considere o código abaixo e escreva, utilizando a notação Σ, o somatório do número de multiplicações em função da entrada *n*. Em seguida, encontre a fórmula fechada do somatório em função apenas de *n*.

```
public void calcula (int n) {
    for (int i = 1; i <= n; i++) {
        a *= 3;
        for (int j = 1; j <= i; j++) {
            b *= 2;
            c *= 5;
        }
    }
}
```

$3 + \sum_{1}^{3} 2 \cdot i$

15

$n + \sum_{1}^{n} 2i$

Dica: Lembre-se da fórmula fechada para o somatório dos termos de uma P.A. (Progressão Aritmética) de razão = 1 e termo inicial = 0:

$$S_n = \sum_{0 \le i \le n} i = \frac{n \cdot (n+1)}{2}$$

Luca Ferrari Azalim

Questão 5:

Função: $\boxed{n + \sum_{1}^{n}(2i)}$

Fórmula fechada: 5,0

$$n + \sum_{1}^{n}(2i) =$$

$$n + 2 \cdot \sum_{1}^{n} i =$$

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

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

6) (6 pontos) Implemente um método, em Java, denominado combinador, que receba duas *strings*. Esse método deve ser capaz de combinar as duas *strings* recebidas como parâmetros, alternando as letras de cada *string* recebida a cada dois caracteres, da seguinte forma: comece com as duas primeiras letras da primeira *string*, seguida pelas duas primeiras letras da segunda *string*, continue com a terceira e a quarta letras da primeira *string*, e assim sucessivamente. As letras restantes da cadeia mais longa devem ser adicionadas ao fim da *string* resultante, que deve ser retornada.
Por exemplo, se as duas *strings* recebidas como parâmetros forem: **Too** e **pCder**, a *string* resultante será: **TopCoder**

String ~~void~~ combinar (String a, String b){

```
String c = "";
int pa = 0, pb = 0;
while(pa < a.length && pb < b.length){
    if(pa < a.length - 1){
        c += a.charAt(pa++);
        c += a.charAt(pa++);
    }
    if(pb < b.length - 1){
        c += b.charAt(pb++);
        c += b.charAt(pb++);
    }
}
while(pa < a.length)
    c += a.charAt(pa++);
while(pb < b.length)
    c += b.charAt(pb++);
return c;
}
```

5,0

Não deixa no resultado após um caractere.