

BCC202 – Estruturas de Dados I (2020-02)

Departamento de Computação - Universidade Federal de Ouro Preto - MG

Professora: **Prof. Pedro Silva**

Prova 01 - 10 pontos (Peso 1, 5) (24/06/2021 às 10:10)

Nota:

Nome:

Matrícula:

Leia com atenção as instruções abaixo antes de iniciar a solução da prova:

- A interpretação das questões faz parte da avaliação, por isso faça as observações que achar necessário, por escrito;
- Esta prova é **individual, sem consulta** e tem duração de 1 hora e 40 minutos (das 10:10 às 11:50).
- O Moodle será fechado às 12:20 (30 min extras).
- As soluções para as questões devem ser especificadas em papel e posteriormente fotografadas para submissão via Moodle.
- As folhas de respostas devem ser organizadas de maneira razoavelmente clara e coerente num único arquivo em formato *pdf* a ser submetido no Moodle. Se sua matrícula é 15.1.1234. O formato entregue deve ser *PrimeiroNome_1511234.pdf*.
- No caso de soluções idênticas, as pessoas envolvidas terão suas notas zeradas enquanto a situação não for devidamente esclarecida.
- Respostas misteriosas não receberão crédito total. Uma resposta correta sem explicação, ou desenvolvimento não receberá crédito. Uma resposta incorreta apoiada por explicações substancialmente corretas pode receber crédito parcial.
- **Boa Prova!**

Questao 1 (1.0 pontos)

Considere que você tem um problema para resolver e duas opções de algoritmos. O primeiro algoritmo tem custo $a(n) = 5n^2 + 15$ no pior caso e $a(n) = 10n$ no melhor caso. Já o segundo algoritmo tem custo $b(n) = 200n$ no pior caso e $b(n) = 30n + 10$ no melhor caso. Considerando que o pior caso ocorre 40% das vezes em que você executa o programa enquanto o melhor caso ocorre 60% das vezes, qual algoritmo você escolheria? Justifique a sua resposta em função do tamanho da entrada.

Questao 2 (4.0 pontos)

Considerando um tipo abstrato de dados (TAD) TCirculo, que representa um círculo, faça as questões a seguir:

- a) (0.5) Apresente a estrutura (struct, em C) utilizada pelo TAD. O ponto central do círculo deve ser um ponteiro para a TAD TPonto descrita abaixo.

```
1 | typedef struct {  
2 |     int x, y;  
3 | } TPonto;
```

- b) (1.0) Apresente o código (em C) de uma função que cria um novo TCirculo (assinatura da função dada abaixo) a partir do ponto central e do raio deste círculo. Esta função deve alocar memória para a estrutura a ser preenchida.

```
1 | void criaCirculo(TCirculo **c, int x, int y, int raio);
```

- c) (1.0) Crie uma função que desaloca um Círculo alocado dinamicamente.

- d) (0.5) Apresente o código (em C) da função a seguir, sabendo que esta função retorna 1 se o ponto passado por parâmetro estiver dentro da área delimitada pelo círculo e 0 caso contrário. Utilize a assinatura dada abaixo:

```
1 | int checarPontoCirculo(TCirculo *c, TPonto *ponto);
```

- e) (0.5) Crie uma função que retorna se um ponto se encontra dentro da área delimitada por n círculos. Esta função deve receber um array de ponteiros para círculos, um inteiro indicando o tamanho do array e o ponto que será avaliado.
- f) (0.5) Crie um método main para testar sua TAD, alocando/liberando variáveis e fazendo chamadas às funções criadas.

Questao 3 (4.0 pontos)

Resolva as questões a seguir:

- a) (1.0) Escreva uma função recursiva *imprimir* que recebe dois parâmetros (um array de inteiros e um inteiro) e imprime todos os valores do array em ordem inversa (do último elemento ao primeiro do vetor). Utilize a linguagem C e explique como fica a chamada desta função.
- b) (1.0) Reescreva a função recursiva *imprimir* de forma que os elementos sejam impressos em ordem (do primeiro elemento ao último do vetor).
- c) (1.0) Mostre e resolva a equação de recorrência da função “imprimir” apresentada na questão (a). Qual a complexidade assintótica desta função? Ela pode ser melhor implementada de outra forma? Se sim, como? Justifique sua resposta.
- d) (1.0) Calcule a função de complexidade e a complexidade assintótica dos códigos a seguir. Mostre o critério que utilizou na análise do código (exemplo: número de atribuições a uma determinada variável, número de comparações, etc.).

```
1 | void funcao_a (int *A, int n) {
2 |     int i, j;
3 |     int aux;
4 |
5 |     for (j = 0; j < n; j++) {
6 |         for (i = 0; i < n - 1; i++) {
7 |             if (A[i] > A[i+1]) {
8 |                 aux = A[i];
9 |                 A[i] = A[i+1];
10 |                A[i+1] = aux;
11 |            }
12 |        }
13 |    }
14 | }
```

```
1 | int funcao_b (int n) {
2 |     int i, s = 0;
3 |     for (i = 0; i < n; i++)
4 |         s += funcao_b (n-1);
5 |     return s;
6 | }
```

Questao 4 (1.0 pontos)

Seja o código da função abaixo:

```
1 | void funcao(int n) {
2 |     int i;
3 |     if (n > 1) {
4 |         for (i = 0; i < n; i++)
5 |             print("%d ", i);
6 |         funcao(n/3);
7 |     }
8 | }
```

- (a) (0.4) Formule a equação de recorrência considerando apenas a operação de impressão na tela.
- (b) (0.4) Resolva a equação definida usando o método de expansão.
- (c) (0.2) Defina a ordem de complexidade da função.

Somatórios úteis:

$$\sum_{i=m}^n 1 = n + 1 - m$$

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

$$\sum_{i=1}^n i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=0}^k 2^i = 2^{k+1} - 1$$

$$\sum_{i=0}^k \frac{1}{2^i} = 2 - \frac{1}{2^k}$$

$$\sum_{i=1}^k a^i = \frac{a^{k+1} - 1}{a - 1}$$

Somatório de uma PG finita

$$a_1 \frac{1 - q^n}{1 - q}$$

Lembre-se que $\text{for}(i = 1; i \leq n; i++) = \sum_{i=1}^n = (n + 1 - 1) = n$.

Se $\text{for}(i = 0; i < n; i++)$, então o somatório vai até $n - 1$, ou seja, $\sum_{i=0}^{n-1} = ((n - 1) + 1 - 0) = n$.