Pontifícia Universidade Católica de Minas Gerais
Instituto de Ciências Exatas e Informática
Algoritmos e Estruturas de Dados 1
Professor Cristiano Rodrigues

Prova de Reavaliação (100 pontos)

Nome: _____ Nota: _____

Orientações:

1. Esta atividade avaliativa é de caráter individual e sem consulta.
2. A correta interpretação das questões apresentadas é parte integrante do processo de avaliação.
3. As respostas fornecidas devem ser coerentes com os conteúdos abordados durante as aulas da disciplina e os códigos escritos na linguagem C ou C++.
4. A avaliação levará em consideração a corretude das respostas e a inclusão de comentários relevantes para o entendimento do código.

Q1. **Somatório (20 pontos)**

Escreva uma função que receba um número inteiro n e um número real x. A função deve retornar o somatório dos n primeiros termos da sequência abaixo:

$$+\frac{x}{2}, -\frac{x}{4}, +\frac{x}{6}, -\frac{x}{8}, +\frac{x}{10}, \ldots$$

Em seguida, escreva um programa principal que leia um número inteiro n e um número real x digitados pelo usuário, utilize a função implementada para calcular o somatório dos n primeiros termos da sequência, e imprima o resultado.

Exemplo de execução:

```
Digite o valor de n: 5
Digite o valor de x: 2.0
Resultado: 1.3
```

Q2. **Vetores (20 pontos)**

Escreva um programa que crie um vetor de tamanho 1000 e o preencha com valores inteiros gerados aleatoriamente no intervalo de 1 a 99 ($1 \leq$ valor ≤ 99).

O programa deve, então, solicitar ao usuário que informe um número a ser pesquisado no vetor. Em seguida, o programa deve buscar no vetor o número informado pelo usuário, imprimir as posições (índices) onde o valor foi encontrado, e informar quantas vezes o valor aparece no vetor.

Caso o valor não seja encontrado, informe ao usuário que o valor não está presente no vetor.

Q3. **Manipulação de Matrizes em C usando ponteiros (20 pontos)**

Considere que em um programa principal exista uma matriz quadrada preenchida com números reais. Escreva uma função em C para realizar a operação descrita a seguir. A função deve utilizar aritmética de ponteiro.

Escreva uma função que receba como parâmetros a matriz quadrada e seu tamanho (número de linhas ou colunas). A função deve retornar a soma dos elementos da diagonal secundária e deve obedecer à seguinte definição:

```
double somaDiagonalSecundaria(double *matriz, int tamanho)
```

Dicas

- Lembre-se de que a diagonal secundária de uma matriz quadrada $n \times n$ é composta pelos elementos $matriz[i][n-i-1]$ para $0 \leq i < n$.
- Utilize aritmética de ponteiros para acessar os elementos da matriz.

Q4. **Gerenciamento de Dados de Carros com Structs (20 pontos)**

Desenvolva um programa em C para gerenciar os dados de um grupo de carros. Cada carro é representado por uma estrutura que contém as seguintes informações:

- Modelo do carro
- Marca do carro
- Ano de fabricação do carro
- Preço do carro
- Data da venda do carro (outra estrutura contendo dia, mês e ano)

a) Crie uma estrutura chamada `Data` contendo três campos: `dia`, `mês` e `ano`.

b) Crie uma estrutura chamada `Carro` para representar um carro, que deve conter os campos mencionados acima: modelo, marca, ano de fabricação, preço e data da venda.

c) Suponha que as estruturas criadas anteriormente tenham escopo global e possam ser usadas no programa principal e nas demais funções do código. Escreva uma função chamada `carroMaisCaro` que receba como parâmetro uma lista contendo os dados de n carros (structs do tipo `Carro`) e mostre na tela o preço e a data da venda do carro mais caro da lista. A função deve obedecer à seguinte definição:

```
void carroMaisCaro(Carro lista[], int n);
```

Q5. **Cálculo de Salário de Vendedores (20 pontos)**

Dada a classe `Empregado` com os seguintes atributos e métodos:

```
class Empregado {
private:
    String nome;
    double salarioBase;
    double imposto;

// Construtores padrão e com parâmetros
public:
    Empregado() : nome(""), salarioBase(0.0), imposto(0.0) {}

    Empregado(String nome, double salarioBase, double imposto) {
        this->nome = nome;
        this->salarioBase = salarioBase;
        this->imposto = imposto;
    }

    // Métodos getters e setters
    String getNome() {
        return nome;
    }

    void setNome(String nome) {
        this->nome = nome;
    }

    double getSalarioBase() {
        return salarioBase;
    }

    void setSalarioBase(double salarioBase) {
        this->salarioBase = salarioBase;
    }

    double getImposto() {
        return imposto;
    }

    void setImposto(double imposto) {
        this->imposto = imposto;
    }
};
```

a) Implemente a classe `Vendedor` como subclasse da classe `Empregado`. Um determinado vendedor tem, além dos atributos da classe `Empregado`, os seguintes atributos adicionais:

- `double valorVendas` (correspondente ao valor monetário dos artigos vendidos)
- `double comissao` (porcentagem do `valorVendas` que será adicionada ao salário base do vendedor)

Implemente os construtores e os métodos para manipular os atributos (`get` e `set`) da classe `Vendedor`.

b) Na classe `Vendedor`, implemente um método `calcularSalario` que retorne o salário líquido do vendedor, incluindo a comissão no cálculo do salário e deduzindo os impostos. O cálculo do salário líquido pode ser representado pela fórmula:

$$salarioLiquido = (salarioBase + (comissao \times valorVendas)) \times (1 - imposto)$$

c) Escreva um programa de teste para a classe `Vendedor`. O programa deve criar duas instâncias de `Vendedor`, uma usando o construtor padrão e outra usando o construtor com parâmetros. O programa deve calcular o salário usando o método `calcularSalario` e exibir os resultados.

Exemplos de vendedores:

- Vendedor 1:
  - Nome: Mariano
  - Salário base (R$): 1000.00
  - Imposto: 0.14 (equivalente a 14%)
  - Valor de vendas (R$): 80000.00
  - Comissão: 0.05 (equivalente a 5%)
- Vendedor 2:
  - Nome: Joana
  - Salário base (R$): 1500.00
  - Imposto: 0.12 (equivalente a 12%)
  - Valor de vendas (R$): 120000.00
  - Comissão: 0.06 (equivalente a 6%).