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

Prova 2 (30 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. O uso de conceitos e informações não apresentados (ainda) em sala de aula não é permitido.
4. Todos os códigos devem ser escritos na linguagem C padrão (Ansi C). Respostas escritas em pseudo-código serão penalizadas e consideradas apenas parcialmente.
5. A avaliação levará em consideração não apenas a corretude das respostas, mas também a inclusão de comentários relevantes para o entendimento do código e a adequada identação.

Q1. **Vetores (6 pontos)**

Escreva um programa que crie um vetor de tamanho 100 e o preencha com valores inteiros gerados aleatoriamente no intervalo de 1 a 20 ($1 \leq valor \leq 20$).

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 e imprimir as posições (índices) onde o valor foi encontrado.

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

Dicas

- Utilize a função `rand()` da biblioteca `stdlib.h` para gerar os números aleatórios.
- Lembre-se de inicializar o gerador de números aleatórios usando a função `srand()` com uma semente adequada, como a função `time(NULL)` da biblioteca `time.h`.
- Para cada valor encontrado, imprima a posição (índice) correspondente no vetor.

Solução da questão 1

Q2. **Alocação Dinâmica de Memória e Recursividade (8 pontos)**

Escreva um programa em C que utilize alocação dinâmica de memória para criar uma lista contendo n números reais.

a) O programa deve solicitar ao usuário que informe o tamanho n de uma lista de números reais e deve alocar memória dinamicamente para armazenar os n elementos dessa lista. A lista deve ser preenchida com números digitados pelo usuário.

b) Implemente uma função recursiva para encontrar o maior elemento da lista.

Q3. **Manipulação de Matrizes em C usando ponteiros (8 pontos)**

Considere que em um programa principal exista uma matriz quadrada preenchida com números reais. Escreva duas funções em C para realizar as operações descritas a seguir. As funções devem utilizar aritmética de ponteiro.

a) Escreva uma função que receba como parâmetros a matriz quadrada e seu temanho. A função deve retornar a soma dos elementos abaixo da diagonal principal e deve obedecer à seguinte definição:

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

b) Escreva uma função que receba como parâmetros a matriz quadrada e seu tamanho. A função deve retornar a soma dos elementos da diagonal principal e deve obedecer à seguinte definição:

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

Q4. **Gerenciamento de Dados de Pessoas com Structs (8 pontos)**

Desenvolva um programa em C para gerenciar os dados de um grupo de pessoas. Cada pessoa é representada por uma estrutura que contém as seguintes informações:

- Nome da pessoa
- Altura da pessoa
- Data de nascimento da pessoa (outra estrutura contendo dia, mês e ano)

a) Crie uma estrutura chamada Data para representar a data de nascimento. Esta estrutura deve conter três campos: dia, mes e ano.

b) Crie uma estrutura chamada Pessoa para representar uma pessoa.

c) Suponha que as estruturas criadas anteriormente tenham escopo global e possam ser usadas no programa principal e nas demais funções do código. Escreva um procedimento chamado pessoaMaisAlta que receba como parâmetro uma lista contendo os dados de n pessoas e mostre na tela o nome, altura e ano de nascimento da pessoa mais alta. O procedimento deve obedecer à seguinte definição:

```
void pessoaMaisAlta(Pessoa lista[], int n)
```