ATENÇÃO!

- A nota desta avaliação é composta de **5 pontos** sobre a nota da avaliação continuada **A1**.
- A avaliação deve ser realizada **individualmente**.
- **Não é permitido consulta** a nenhum material, seja analógico ou digital.
- **Não é permitida** a **troca de materiais** entre colegas.
- Qualquer **ato** considerado **suspeito** pelo professor **anulará a avaliação dos alunos envolvidos**.
- Cada questão já está previamente delimitada com seu nome de rotina; não é necessário criar a função main() e nem incluir as bibliotecas #include
- A **endentação** será rigorosamente **avaliada** nesta prova, portanto utilize as tabulações pré-impressas para o devido deslocamento dos comandos subordinados.

Questão 01 – (Peso da questão: 1,0)

Crie um programa que: 1) declare um vetor com 10 posições de inteiros; 2) utilizando estrutura de repetição, leia valores para todas as posições do vetor; 3) solicite um número inteiro n para o usuário; 4) utilizando estrutura de repetição, percorra o vetor e ao final informe quantas vezes o número informado n existe no vetor.

Questão 02 – (Peso da questão: 0,5):

A ordenação por inserção funciona de modo semelhante à forma como algumas pessoas ordenam cartas de baralho. Inicia-se com a mão esquerda vazia e as cartas empilhadas na mesa. Remove-se da pilha uma carta de cada vez, inserindo-a na posição correta na mão esquerda. Para se identificar a posição correta de uma carta, deve-se compará-la com as cartas presentes na mão esquerda, no sentido da direita para a esquerda. Em todos os momentos, as cartas na mão esquerda estão ordenadas, tendo sido obtidas no topo da pilha da mesa (CORMEN, 2009).

Um programador implementou um algoritmo de ordenação semelhante à forma de ordenação de cartas descrita no texto. Ao realizar um teste de com um vetor de nove posições (vetor [1..0]), verificou que o algoritmo não funcionava corretamente.

```
01 para i <- 2 até 9 faça
02    valor <- vetor[i]
03    j <- i - 1
04    enquanto ((j >= 1) e (valor < vetor[j])) faça
05       vetor[i] <- vetor[j]
06          j <- j - 1
07          se (j = 0) então
08             interrompa
09          fim se
10    fim enquanto
11    vetor[j + 1] <- valor
12 fim para
```

Com base nessas informações, assinale a opção em que se apresentam a linha e o respectivo comando a ser substituído, para que o algoritmo ordene corretamente um vetor de inteiros de forma crescente.

a) Linha 01; `para i <- 1 até 9 faça`

b) Linha 03; `j <- i`

c) Linha 04; `enquanto ((j >= 1) ou (valor < vetor[j])) faça`

d) Linha 05; `vetor[j + 1] <- vetor[j]`

e) Linha 11; `vetor[j] <- valor`

Questão 03 – (Peso da questão: 0,5):

Considere o seguinte trecho de código escrito na linguagem de programação C:

```c
#define LINHAS 2
#define COLUNAS 3
int main() {
    int matriz[LINHAS][COLUNAS] = {{1, 2, 3},{4, 5, 6}};
    int soma = 0;
    for (int i = 0; i < LINHAS; i++) {
        for (int j = 0; j < COLUNAS; j++) {
           soma += matriz[i][j];
        }
    }
        printf("Soma dos elementos da matriz: %d\n", soma);
    return 0;
}
```

Assinale a alternativa correta para a saída correta do código.

a) Soma dos elementos da matriz: 21
b) Soma dos elementos da matriz: 36
c) Soma dos elementos da matriz: 15
d) Soma dos elementos da matriz: 18
e) Soma dos elementos da matriz: 24

Questão 04 – (Peso da questão: 1,0):

Uma empresa de cosméticos comercializa cinco diferentes tipos de produtos e os armazena em uma estante de 40 x 40 posições. Em cada posição da estante, pode ficar armazenada apenas uma caixa com um desses produtos. Para facilitar sua identificação, os produtos foram codificados da seguinte forma:

1: shampoo;
2: condicionador;
3: hidratante;
4: tintura;
5: demaquilante;
0: vazio.

Nessa situação e considerando o desenvolvimento de um sistema para gerenciar a organização dos produtos na estante, estabeleceu-se a declaração de variáveis a seguir:

```
Var
     Estante: matriz [1..40][1..40] de inteiro
     Produtos: vetor [0..5] de texto = ("vazio","xampu","condicionador",
                          "hidratante", "tintura", "demaquilante")
     Contador: vetor [0..5] de inteiro = (0,0,0,0,0,0)
     i, j: inteiro
```

Com base nessa declaração e considerando a codificação dos produtos exposta, faça o que se pede nos itens a seguir:

a) Escreva um trecho de código para ler os códigos de produtos e armazená-los na matriz Estante.
b) Escreva um trecho de código para contar e imprimir a quantidade de caixas de cada tipo de produto na estante.

Questão 05 – (Peso da questão: 0,5):

Considere a declaração de um vetor em C:

```c
int numeros[] = {2, 4, 6, 8, 10};
```

Assinale a alternativa correta para acessar o terceiro elemento deste vetor:

a) numeros[2]
b) numeros(3)
c) numeros.3
d) numeros->3
e) números{3}

Questão 06 – (Peso da questão: 0,5):

Dado o código em C abaixo:

```
#include <stdio.h>
void trocarValores(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int main() {
    int x = 5, y = 10;
    trocarValores(&x, &y);
    printf("x: %d, y: %d\n", x, y);
    return 0;
}
```

Assinale a alternativa correta para a saída após a execução deste programa:

a) x: 5, y: 10
b) x: 10, y: 5
c) x: 15, y: 5
d) x: 5, y: 15
e) x: 10, y: 10

Questão 07 – (Peso da questão: 1,0):

Você foi designado para desenvolver um sistema de cadastro de alunos para uma instituição de ensino. Cada aluno possui informações como nome, matrícula, notas em três disciplinas e média final. Utilize structs em C para criar uma solução que permita realizar as seguintes operações:

- Cadastro de Aluno: Permita ao usuário cadastrar um novo aluno, informando nome, matrícula e as notas nas disciplinas de Matemática, Português e Ciências.
- Cálculo de Média: Implemente uma função que, dado um aluno, calcule a média final considerando as notas nas três disciplinas. A média deve ser armazenada na struct do aluno.
- Consulta de Aluno: Crie uma função que, dado o número de matrícula de um aluno, exiba na tela todas as informações sobre o aluno, incluindo nome, matrícula, notas e média final.
- Listagem de Alunos Aprovados e Reprovados: Desenvolva funções que listem os alunos aprovados e reprovados. Considere a média mínima para aprovação como 6.0.

Certifique-se de modularizar o código, criando funções separadas para cada uma das operações mencionadas acima. Além disso, forneça um menu interativo para que o usuário possa escolher qual operação deseja realizar. Ao final, seu programa deve ser capaz de cadastrar alunos, calcular suas médias e fornecer informações sobre o desempenho acadêmico, identificando quem foi aprovado e quem foi reprovado.