2º Lista de Exercícios - Introdução à Programação]

**VETORES**

1) Faça um programa que leia 15 números inteiros e armazene em um vetor. Ao final imprima o vetor armazenado.

#include <stdio.h>

int main() {

int vetor[15];

printf("Digite 15 numeros inteiros:\n");

for (int i = 0; i < 15; ++i) {

printf("Numero %d: ", i + 1);

scanf("%d", &vetor[i]);

}

printf("\nVetor armazenado:\n");

for (int i = 0; i < 15; ++i) {

printf("%d ", vetor[i]);

}

printf("\n");

return 0;

}

2) Ler um vetor de 10 elementos reais. Crie um segundo vetor, com todos os elementos na ordem inversa, ou seja, o último elemento passará a ser o primeiro, o penúltimo será o segundo e assim por diante. Imprima os dois vetores.

#include <stdio.h>

int main() {

int tamanho = 10;

float vetorOriginal[10];

float vetorInverso[10];

printf("Digite 10 elementos reais:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%f", &vetorOriginal[i]);

}

for (int i = 0; i < tamanho; ++i) {

vetorInverso[i] = vetorOriginal[tamanho - 1 - i];

}

printf("\nVetor Original:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%.2f ", vetorOriginal[i]);

}

printf("\nVetor Inverso:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%.2f ", vetorInverso[i]);

}

printf("\n");

return 0;

}

3) Ler um vetor de 12 elementos inteiros e positivos. Criar um segundo vetor da seguinte forma: os elementos de índice par receberão os respectivos elementos divididos por 2; os elementos de índice ímpar receberão os respectivos elementos multiplicados por 3. Imprima os dois vetores.

#include <stdio.h>

int main() {

int tamanho = 12;

int vetorOriginal[12];

float vetorModificado[12];

printf("Digite 12 elementos inteiros e positivos:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%d", &vetorOriginal[i]);

while (vetorOriginal[i] <= 0) {

printf("Digite um numero inteiro e positivo: ");

scanf("%d", &vetorOriginal[i]);

}

}

for (int i = 0; i < tamanho; ++i) {

if (i % 2 == 0) {

vetorModificado[i] = vetorOriginal[i] / 2.0;

} else {

vetorModificado[i] = vetorOriginal[i] \* 3.0;

}

}

printf("\nVetor Original:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%d ", vetorOriginal[i]);

}

printf("\nVetor Modificado:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%.2f ", vetorModificado[i]);

}

printf("\n");

return 0;

}

4) Ler um vetor com 10 nomes de pessoas, após pedir que o usuário digite um nome qualquer de pessoa. Escrever a mensagem “ACHEI”, se o nome estiver armazenado no vetor C ou “NÃO ACHEI” caso contrário.

#include <stdio.h>

#include <string.h>

int main() {

int tamanho = 10;

char nomes[10][50];

char nomeBusca[50];

printf("Digite 10 nomes de pessoas:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Nome %d: ", i + 1);

scanf("%s", nomes[i]);

}

printf("Digite um nome para buscar: ");

scanf("%s", nomeBusca);

int encontrado = 0;

for (int i = 0; i < tamanho; ++i) {

if (strcmp(nomes[i], nomeBusca) == 0) {

encontrado = 1;

break;

}

}

if (encontrado) {

printf("ACHEI\n");

} else {

printf("NAO ACHEI\n");

}

return 0;

}

5) Faça um programa que leia e monte dois vetores de números inteiros com 20 números cada. Depois de montados gere um terceiro vetor formado pela diferença dos dois vetores lidos, um quarto vetor formado pela soma dos dois vetores lidos e por último um quinto vetor formado pela multiplicação dos dois vetores lidos.

#include <stdio.h>

int main() {

int tamanho = 20;

int vetor1[20], vetor2[20], vetorDiferenca[20], vetorSoma[20], vetorMultiplicacao[20];

printf("Digite os 20 elementos do primeiro vetor:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%d", &vetor1[i]);

}

printf("\nDigite os 20 elementos do segundo vetor:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%d", &vetor2[i]);

}

for (int i = 0; i < tamanho; ++i) {

vetorDiferenca[i] = vetor1[i] - vetor2[i];

}

for (int i = 0; i < tamanho; ++i) {

vetorSoma[i] = vetor1[i] + vetor2[i];

}

for (int i = 0; i < tamanho; ++i) {

vetorMultiplicacao[i] = vetor1[i] \* vetor2[i];

}

printf("\nVetor Diferenca:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%d ", vetorDiferenca[i]);

}

printf("\nVetor Soma:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%d ", vetorSoma[i]);

}

printf("\nVetor Multiplicacao:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%d ", vetorMultiplicacao[i]);

}

printf("\n");

return 0;

}

6) Utilizando vetores, crie um programa que organize uma quantidade qualquer de números inteiros fornecidos pelo usuário da seguinte forma: primeiro os números pares em ordem crescente e depois os números ímpares em ordem decrescente.

#include <stdio.h>

void organizarNumeros(int vetor[], int tamanho) {

int pares[tamanho];

int impares[tamanho];

int numPares = 0, numImpares = 0;

for (int i = 0; i < tamanho; ++i) {

if (vetor[i] % 2 == 0) {

pares[numPares++] = vetor[i];

} else {

impares[numImpares++] = vetor[i];

}

}

for (int i = 0; i < numPares - 1; ++i) {

for (int j = i + 1; j < numPares; ++j) {

if (pares[i] > pares[j]) {

int temp = pares[i];

pares[i] = pares[j];

pares[j] = temp;

}

}

}

for (int i = 0; i < numImpares - 1; ++i) {

for (int j = i + 1; j < numImpares; ++j) {

if (impares[i] < impares[j]) {

int temp = impares[i];

impares[i] = impares[j];

impares[j] = temp;

}

}

}

for (int i = 0; i < numPares; ++i) {

vetor[i] = pares[i];

}

for (int i = 0; i < numImpares; ++i) {

vetor[numPares + i] = impares[i];

}

}

int main() {

int tamanho;

printf("Digite a quantidade de numeros inteiros: ");

scanf("%d", &tamanho);

int vetor[tamanho];

printf("Digite os numeros inteiros:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Numero %d: ", i + 1);

scanf("%d", &vetor[i]);

}

organizarNumeros(vetor, tamanho);

printf("\nVetor Organizado:\n");

for (int i = 0; i < tamanho; ++i) {

printf("%d ", vetor[i]);

}

printf("\n");

return 0;

}

7) Dados dois vetores de tamanho N, faça uma função que diga se os mesmos possuem conteúdo igual.

#include <stdio.h>

int vetoresIguais(int vetor1[], int vetor2[], int tamanho) {

for (int i = 0; i < tamanho; ++i) {

if (vetor1[i] != vetor2[i]) {

return 0;

}

}

return 1;

}

int main() {

int tamanho;

printf("Digite o tamanho dos vetores: ");

scanf("%d", &tamanho);

int vetor1[tamanho];

int vetor2[tamanho];

printf("Digite os elementos do primeiro vetor:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%d", &vetor1[i]);

}

printf("\nDigite os elementos do segundo vetor:\n");

for (int i = 0; i < tamanho; ++i) {

printf("Elemento %d: ", i + 1);

scanf("%d", &vetor2[i]);

}

if (vetoresIguais(vetor1, vetor2, tamanho)) {

printf("\nOs vetores tem conteudo igual.\n");

} else {

printf("\nOs vetores tem conteudo diferente.\n");

}

return 0;

}

**MATRIZES**

1) Faça um algoritmo que construa uma matriz de nome MAT de 10 linhas e 15 colunas contendo números inteiros. Em seguida escreva a soma dos elementos de cada linha e se a soma dos elementos é par ou ímpar. Por fim, escreva a soma dos elementos de cada coluna e se a soma dos elementos é par ou ímpar.

#include <stdio.h>

int main() {

int linhas = 10;

int colunas = 15;

int MAT[linhas][colunas];

printf("Digite os elementos da matriz (%dx%d):\n", linhas, colunas);

for (int i = 0; i < linhas; ++i) {

for (int j = 0; j < colunas; ++j) {

printf("Elemento [%d][%d]: ", i + 1, j + 1);

scanf("%d", &MAT[i][j]);

}

}

printf("\nSoma dos elementos por linha:\n");

for (int i = 0; i < linhas; ++i) {

int somaLinha = 0;

for (int j = 0; j < colunas; ++j) {

somaLinha += MAT[i][j];

}

printf("Linha %d: Soma = %d, %s\n", i + 1, somaLinha, somaLinha % 2 == 0 ? "Par" : "impar");

}

printf("\nSoma dos elementos por coluna:\n");

for (int j = 0; j < colunas; ++j) {

int somaColuna = 0;

for (int i = 0; i < linhas; ++i) {

somaColuna += MAT[i][j];

}

printf("Coluna %d: Soma = %d, %s\n", j + 1, somaColuna, somaColuna % 2 == 0 ? "Par" : "Impar");

}

return 0;

}

2) Faça um algoritmo que construa uma matriz 50 por 50 de números reais e depois de construída, colocar o conteúdo de sua diagonal principal dentro de um vetor e depois do vetor montado, imprimir o vetor.

#include <stdio.h>

#define TAMANHO\_MATRIZ 50

int main() {

float matriz[TAMANHO\_MATRIZ][TAMANHO\_MATRIZ];

for (int i = 0; i < TAMANHO\_MATRIZ; i++) {

for (int j = 0; j < TAMANHO\_MATRIZ; j++) {

matriz[i][j] = (float)(i \* TAMANHO\_MATRIZ + j + 1);

}

}

float diagonalPrincipal[TAMANHO\_MATRIZ];

for (int i = 0; i < TAMANHO\_MATRIZ; i++) {

diagonalPrincipal[i] = matriz[i][i];

}

printf("Matriz 50x50:\n");

for (int i = 0; i < TAMANHO\_MATRIZ; i++) {

for (int j = 0; j < TAMANHO\_MATRIZ; j++) {

printf("%.2f\t", matriz[i][j]);

}

printf("\n");

}

printf("\nConteudo da diagonal principal:\n");

for (int i = 0; i < TAMANHO\_MATRIZ; i++) {

printf("%.2f ", diagonalPrincipal[i]);

}

return 0;

}

3) Faça um programa que gere uma matriz transposta (Matriz transposta é toda a matriz onde são trocadas as linhas pelas colunas, ou vice-versa).

#include <stdio.h>

#define LINHAS 3

#define COLUNAS 4

void gerarMatrizTransposta(int matriz[LINHAS][COLUNAS], int transposta[COLUNAS][LINHAS]) {

for (int i = 0; i < LINHAS; i++) {

for (int j = 0; j < COLUNAS; j++) {

transposta[j][i] = matriz[i][j];

}

}

}

void imprimirMatriz(int linhas, int colunas, int matriz[][colunas]) {

for (int i = 0; i < linhas; i++) {

for (int j = 0; j < colunas; j++) {

printf("%d\t", matriz[i][j]);

}

printf("\n");

}

}

int main() {

int matriz[LINHAS][COLUNAS] = {

{1, 2, 3, 4},

{5, 6, 7, 8},

{9, 10, 11, 12}

};

int transposta[COLUNAS][LINHAS];

printf("Matriz Original:\n");

imprimirMatriz(LINHAS, COLUNAS, matriz);

gerarMatrizTransposta(matriz, transposta);

printf("\nMatriz Transposta:\n");

imprimirMatriz(COLUNAS, LINHAS, transposta);

return 0;

}

4) Faça um programa que faça a multiplicação de duas matrizes.

#include <stdio.h>

#include <stdlib.h>

void multiplicarMatrizes(int \*\*matriz1, int linhas1, int colunas1,

int \*\*matriz2, int linhas2, int colunas2,

int \*\*resultado) {

if (colunas1 != linhas2) {

printf("As matrizes nao podem ser multiplicadas devido as dimensoes inconsistentes.\n");

return;

}

for (int i = 0; i < linhas1; i++) {

for (int j = 0; j < colunas2; j++) {

resultado[i][j] = 0;

for (int k = 0; k < colunas1; k++) {

resultado[i][j] += matriz1[i][k] \* matriz2[k][j];

}

}

}

}

void imprimirMatriz(int linhas, int colunas, int \*\*matriz) {

for (int i = 0; i < linhas; i++) {

for (int j = 0; j < colunas; j++) {

printf("%d\t", matriz[i][j]);

}

printf("\n");

}

}

void liberarMatriz(int linhas, int \*\*matriz) {

for (int i = 0; i < linhas; i++) {

free(matriz[i]);

}

free(matriz);

}

int main() {

int linhas1, colunas1, linhas2, colunas2;

printf("Informe o numero de linhas e colunas da primeira matriz (separadas por espaco): ");

scanf("%d %d", &linhas1, &colunas1);

printf("Informe o numero de linhas e colunas da segunda matriz (separadas por espaco): ");

scanf("%d %d", &linhas2, &colunas2);

if (colunas1 != linhas2) {

printf("As matrizes nao podem ser multiplicadas devido as dimensoes inconsistentes.\n");

return 1;

}

int \*\*matriz1 = (int \*\*)malloc(linhas1 \* sizeof(int \*));

for (int i = 0; i < linhas1; i++) {

matriz1[i] = (int \*)malloc(colunas1 \* sizeof(int));

}

int \*\*matriz2 = (int \*\*)malloc(linhas2 \* sizeof(int \*));

for (int i = 0; i < linhas2; i++) {

matriz2[i] = (int \*)malloc(colunas2 \* sizeof(int));

}

int \*\*resultado = (int \*\*)malloc(linhas1 \* sizeof(int \*));

for (int i = 0; i < linhas1; i++) {

resultado[i] = (int \*)malloc(colunas2 \* sizeof(int));

}

printf("Informe os elementos da primeira matriz:\n");

for (int i = 0; i < linhas1; i++) {

for (int j = 0; j < colunas1; j++) {

scanf("%d", &matriz1[i][j]);

}

}

printf("Informe os elementos da segunda matriz:\n");

for (int i = 0; i < linhas2; i++) {

for (int j = 0; j < colunas2; j++) {

scanf("%d", &matriz2[i][j]);

}

}

multiplicarMatrizes(matriz1, linhas1, colunas1, matriz2, linhas2, colunas2, resultado);

printf("\nMatriz Resultante:\n");

imprimirMatriz(linhas1, colunas2, resultado);

liberarMatriz(linhas1, matriz1);

liberarMatriz(linhas2, matriz2);

liberarMatriz(linhas1, resultado);

return 0;

}

5) Faça um programa que faça a soma de duas matrizes.

#include <stdio.h>

#include <stdlib.h>

void somarMatrizes(int \*\*matriz1, int linhas1, int colunas1,

int \*\*matriz2, int linhas2, int colunas2,

int \*\*resultado) {

if (linhas1 != linhas2 || colunas1 != colunas2) {

printf("As matrizes nao podem ser somadas devido as dimensoes inconsistentes.\n");

return;

}

for (int i = 0; i < linhas1; i++) {

for (int j = 0; j < colunas1; j++) {

resultado[i][j] = matriz1[i][j] + matriz2[i][j];

}

}

}

void imprimirMatriz(int linhas, int colunas, int \*\*matriz) {

for (int i = 0; i < linhas; i++) {

for (int j = 0; j < colunas; j++) {

printf("%d\t", matriz[i][j]);

}

printf("\n");

}

}

void liberarMatriz(int linhas, int \*\*matriz) {

for (int i = 0; i < linhas; i++) {

free(matriz[i]);

}

free(matriz);

}

int main() {

int linhas1, colunas1, linhas2, colunas2;

printf("Informe o numero de linhas e colunas da primeira matriz (separadas por espaco): ");

scanf("%d %d", &linhas1, &colunas1);

printf("Informe o numero de linhas e colunas da segunda matriz (separadas por espaco): ");

scanf("%d %d", &linhas2, &colunas2);

if (linhas1 != linhas2 || colunas1 != colunas2) {

printf("As matrizes não podem ser somadas devido as dimensoes inconsistentes.\n");

return 1;

}

int \*\*matriz1 = (int \*\*)malloc(linhas1 \* sizeof(int \*));

for (int i = 0; i < linhas1; i++) {

matriz1[i] = (int \*)malloc(colunas1 \* sizeof(int));

}

int \*\*matriz2 = (int \*\*)malloc(linhas2 \* sizeof(int \*));

for (int i = 0; i < linhas2; i++) {

matriz2[i] = (int \*)malloc(colunas2 \* sizeof(int));

}

int \*\*resultado = (int \*\*)malloc(linhas1 \* sizeof(int \*));

for (int i = 0; i < linhas1; i++) {

resultado[i] = (int \*)malloc(colunas1 \* sizeof(int));

}

printf("Informe os elementos da primeira matriz:\n");

for (int i = 0; i < linhas1; i++) {

for (int j = 0; j < colunas1; j++) {

scanf("%d", &matriz1[i][j]);

}

}

printf("Informe os elementos da segunda matriz:\n");

for (int i = 0; i < linhas2; i++) {

for (int j = 0; j < colunas2; j++) {

scanf("%d", &matriz2[i][j]);

}

}

somarMatrizes(matriz1, linhas1, colunas1, matriz2, linhas2, colunas2, resultado);

printf("\nMatriz Resultante:\n");

imprimirMatriz(linhas1, colunas1, resultado);

liberarMatriz(linhas1, matriz1);

liberarMatriz(linhas2, matriz2);

liberarMatriz(linhas1, resultado);

return 0;

}

6) Faça um programa que deverá permitir que o usuário entre com os valores dos elementos de uma matriz quadrada de ordem 4 e possibilite o usuário realizar as seguintes funcionalidades: a) Imprimir todos os elementos da matriz;

b) Somar os quadrados de todos os elementos da primeira coluna;

c) Somar todos os elementos da terceira linha;

d) Somar os elementos da diagonal principal; e

e) Somar todos os elementos de índice par da segunda linha.

#include <stdio.h>

#define ORDEM 4

void imprimirMatriz(int matriz[ORDEM][ORDEM]) {

printf("Matriz:\n");

for (int i = 0; i < ORDEM; i++) {

for (int j = 0; j < ORDEM; j++) {

printf("%d\t", matriz[i][j]);

}

printf("\n");

}

}

int somarQuadradosPrimeiraColuna(int matriz[ORDEM][ORDEM]) {

int soma = 0;

for (int i = 0; i < ORDEM; i++) {

soma += matriz[i][0] \* matriz[i][0];

}

return soma;

}

int somarTerceiraLinha(int matriz[ORDEM][ORDEM]) {

int soma = 0;

for (int j = 0; j < ORDEM; j++) {

soma += matriz[2][j];

}

return soma;

}

int somarDiagonalPrincipal(int matriz[ORDEM][ORDEM]) {

int soma = 0;

for (int i = 0; i < ORDEM; i++) {

soma += matriz[i][i];

}

return soma;

}

int somarElementosIndiceParSegundaLinha(int matriz[ORDEM][ORDEM]) {

int soma = 0;

for (int j = 0; j < ORDEM; j += 2) {

soma += matriz[1][j];

}

return soma;

}

int main() {

int matriz[ORDEM][ORDEM];

printf("Informe os elementos da matriz %dx%d:\n", ORDEM, ORDEM);

for (int i = 0; i < ORDEM; i++) {

for (int j = 0; j < ORDEM; j++) {

scanf("%d", &matriz[i][j]);

}

}

imprimirMatriz(matriz);

printf("\nSoma dos quadrados da primeira coluna: %d\n", somarQuadradosPrimeiraColuna(matriz));

printf("Soma dos elementos da terceira linha: %d\n", somarTerceiraLinha(matriz));

printf("Soma dos elementos da diagonal principal: %d\n", somarDiagonalPrincipal(matriz));

printf("Soma dos elementos de indice par da segunda linha: %d\n", somarElementosIndiceParSegundaLinha(matriz));

return 0;

}

**STRUCT**

1)Utilizando struct, faça um programa que permita a entrada de nome, endereço e telefone de 10 pessoas e os imprima em ordem alfabética.

#include <stdio.h>

#include <string.h>

struct Pessoa {

char nome[100];

char endereco[100];

char telefone[15];

};

int compararPessoas(const void \*a, const void \*b) {

return strcmp(((struct Pessoa \*)a)->nome, ((struct Pessoa \*)b)->nome);

}

int main() {

struct Pessoa pessoas[10];

for (int i = 0; i < 10; ++i) {

printf("Digite o nome da pessoa %d: ", i + 1);

fgets(pessoas[i].nome, sizeof(pessoas[i].nome), stdin);

pessoas[i].nome[strcspn(pessoas[i].nome, "\n")] = '\0'; // Remover a quebra de linha

printf("Digite o endereco da pessoa %d: ", i + 1);

fgets(pessoas[i].endereco, sizeof(pessoas[i].endereco), stdin);

pessoas[i].endereco[strcspn(pessoas[i].endereco, "\n")] = '\0'; // Remover a quebra de linha

printf("Digite o telefone da pessoa %d: ", i + 1);

fgets(pessoas[i].telefone, sizeof(pessoas[i].telefone), stdin);

pessoas[i].telefone[strcspn(pessoas[i].telefone, "\n")] = '\0'; // Remover a quebra de linha

printf("\n");

}

qsort(pessoas, 10, sizeof(struct Pessoa), compararPessoas);

printf("Informacoes das Pessoas em Ordem Alfabetica:\n");

for (int i = 0; i < 10; ++i) {

printf("Nome: %s\n", pessoas[i].nome);

printf("Endereço: %s\n", pessoas[i].endereco);

printf("Telefone: %s\n", pessoas[i].telefone);

printf("\n");

}

return 0;

}

2) Faça um programa que realize a leitura dos seguintes dados relativos a um conjunto de alunos: Matricula, Nome, Código da Disciplina, Nota1 e Nota2. Considere uma turma de até 10 alunos. Após ler todos os dados digitados, exibir na tela a listagem final dos alunos com as suas respectivas médias finais (use uma média ponderada: Nota1 com peso=1.0 e Nota2 com peso=2.0).

#include <stdio.h>

struct Aluno {

int matricula;

char nome[100];

int codigoDisciplina;

float nota1;

float nota2;

};

int main() {

struct Aluno alunos[10];

int numeroAlunos;

do {

printf("Digite o numero de alunos (ate 10): ");

scanf("%d", &numeroAlunos);

} while (numeroAlunos <= 0 || numeroAlunos > 10);

printf("Digite os dados dos alunos:\n");

for (int i = 0; i < numeroAlunos; ++i) {

printf("Aluno %d:\n", i + 1);

printf("Matricula: ");

scanf("%d", &alunos[i].matricula);

printf("Nome: ");

scanf("%s", alunos[i].nome);

printf("Codigo da Disciplina: ");

scanf("%d", &alunos[i].codigoDisciplina);

printf("Nota1: ");

scanf("%f", &alunos[i].nota1);

printf("Nota2: ");

scanf("%f", &alunos[i].nota2);

printf("\n");

}

printf("Listagem Final dos Alunos:\n");

for (int i = 0; i < numeroAlunos; ++i) {

float mediaFinal = (alunos[i].nota1 \* 1.0 + alunos[i].nota2 \* 2.0) / 3.0;

printf("Matricula: %d\n", alunos[i].matricula);

printf("Nome: %s\n", alunos[i].nome);

printf("Codigo da Disciplina: %d\n", alunos[i].codigoDisciplina);

printf("Nota1: %.2f\n", alunos[i].nota1);

printf("Nota2: %.2f\n", alunos[i].nota2);

printf("Media Final: %.2f\n", mediaFinal);

printf("\n");

}

return 0;

}

3) Escreva um trecho de código para fazer a criação dos novos tipos de dados conforme solicitado abaixo: • Horario: composto de hora, minutos e segundos. • Data: composto de dia, mês e ano. • Compromisso: composto de uma data, horário e texto que descreve o compromisso.

#include <stdio.h>

struct Horario {

int hora;

int minutos;

int segundos;

};

struct Data {

int dia;

int mes;

int ano;

};

struct Compromisso {

struct Data dataCompromisso;

struct Horario horarioCompromisso;

char texto[100];

};

int main() {

int numCompromissos;

printf("Digite o numero de compromissos: ");

scanf("%d", &numCompromissos);

if (numCompromissos > 10) {

printf("O numero de compromissos nao pode ser maior que 10.\n");

return 1;

}

struct Compromisso compromissos[numCompromissos];

for (int i = 0; i < numCompromissos; ++i) {

printf("Compromisso %d:\n", i + 1);

printf("Digite a data (dia mes ano): ");

scanf("%d %d %d", &compromissos[i].dataCompromisso.dia, &compromissos[i].dataCompromisso.mes, &compromissos[i].dataCompromisso.ano);

printf("Digite o horario (hora minutos segundos): ");

scanf("%d %d %d", &compromissos[i].horarioCompromisso.hora, &compromissos[i].horarioCompromisso.minutos, &compromissos[i].horarioCompromisso.segundos);

printf("Digite o texto do compromisso: ");

scanf(" %[^\n]s", compromissos[i].texto);

printf("\n");

}

printf("Listagem dos Compromissos:\n");

for (int i = 0; i < numCompromissos; ++i) {

printf("Compromisso %d:\n", i + 1);

printf("Data: %02d/%02d/%04d\n", compromissos[i].dataCompromisso.dia, compromissos[i].dataCompromisso.mes, compromissos[i].dataCompromisso.ano);

printf("Horario: %02d:%02d:%02d\n", compromissos[i].horarioCompromisso.hora, compromissos[i].horarioCompromisso.minutos, compromissos[i].horarioCompromisso.segundos);

printf("Texto: %s\n", compromissos[i].texto);

printf("\n");

}

return 0;

}

4) Implemente um programa que leia o nome, a idade e o enderec¸o de uma pessoa e Armazene os dados em uma estrutura.

#include <stdio.h>

#include <string.h>

struct Pessoa {

char nome[100];

int idade;

char endereco[100];

};

int main() {

struct Pessoa pessoa;

printf("Digite o nome da pessoa: ");

scanf("%s", pessoa.nome);

printf("Digite a idade da pessoa: ");

scanf("%d", &pessoa.idade);

while (getchar() != '\n');

printf("Digite o endereco da pessoa: ");

fgets(pessoa.endereco, sizeof(pessoa.endereco), stdin);

size\_t enderecoLength = strlen(pessoa.endereco);

if (enderecoLength > 0 && pessoa.endereco[enderecoLength - 1] == '\n') {

pessoa.endereco[enderecoLength - 1] = '\0';

}

printf("\nDados da Pessoa:\n");

printf("Nome: %s\n", pessoa.nome);

printf("Idade: %d\n", pessoa.idade);

printf("Endereco: %s\n", pessoa.endereco);

return 0;

}

5) Construa uma estrutura aluno com nome, número de matrícula e curso. Leia do usuário a informação de 5 alunos, armazene em vetor dessa estrutura e imprima os dados na tela.

#include <stdio.h>

#include <string.h>

struct Aluno {

char nome[100];

int matricula;

char curso[50];

};

int main() {

struct Aluno alunos[5];

printf("Digite as informacoes dos alunos:\n");

for (int i = 0; i < 5; ++i) {

printf("Aluno %d:\n", i + 1);

printf("Nome: ");

scanf("%s", alunos[i].nome);

printf("Numero de Matricula: ");

scanf("%d", &alunos[i].matricula);

while (getchar() != '\n');

printf("Curso: ");

fgets(alunos[i].curso, sizeof(alunos[i].curso), stdin);

char \*newline = strchr(alunos[i].curso, '\n');

if (newline != NULL) {

\*newline = '\0';

}

printf("\n");

}

printf("Listagem dos Alunos:\n");

for (int i = 0; i < 5; ++i) {

printf("Aluno %d:\n", i + 1);

printf("Nome: %s\n", alunos[i].nome);

printf("Numero de Matricula: %d\n", alunos[i].matricula);

printf("Curso: %s\n", alunos[i].curso);

printf("\n");

}

return 0;

}

6) Crie uma estrutura representando os alunos de um determinado curso. A estrutura deve conter a matrícula do aluno, nome, nota da primeira prova, nota da segunda prova e nota da terceira prova. (a) Permita ao usuário entrar com os dados de 5 alunos. (b) Encontre o aluno com maior nota da primeira prova. (c) Encontre o aluno com maior média geral. (d) Encontre o aluno com menor média geral (e) Para cada aluno diga se ele foi aprovado ou reprovado, considerando o valor 6 para Aprovação.

#include <stdio.h>

struct Aluno {

int matricula;

char nome[100];

float notaProva1;

float notaProva2;

float notaProva3;

float mediaGeral;

};

int main() {

struct Aluno alunos[5];

printf("Digite as informacoes dos alunos:\n");

for (int i = 0; i < 5; ++i) {

printf("Aluno %d:\n", i + 1);

printf("Matricula: ");

scanf("%d", &alunos[i].matricula);

printf("Nome: ");

scanf("%s", alunos[i].nome);

printf("Nota da Prova 1: ");

scanf("%f", &alunos[i].notaProva1);

printf("Nota da Prova 2: ");

scanf("%f", &alunos[i].notaProva2);

printf("Nota da Prova 3: ");

scanf("%f", &alunos[i].notaProva3);

l

alunos[i].mediaGeral = (alunos[i].notaProva1 + alunos[i].notaProva2 + alunos[i].notaProva3) / 3.0;

printf("\n");

}

float maiorNotaProva1 = alunos[0].notaProva1;

int indiceMaiorNotaProva1 = 0;

for (int i = 1; i < 5; ++i) {

if (alunos[i].notaProva1 > maiorNotaProva1) {

maiorNotaProva1 = alunos[i].notaProva1;

indiceMaiorNotaProva1 = i;

}

}

printf("Aluno com maior nota na Prova 1:\n");

printf("Nome: %s\n", alunos[indiceMaiorNotaProva1].nome);

printf("Nota da Prova 1: %.2f\n", alunos[indiceMaiorNotaProva1].notaProva1);

printf("\n");

float maiorMediaGeral = alunos[0].mediaGeral;

int indiceMaiorMediaGeral = 0;

for (int i = 1; i < 5; ++i) {

if (alunos[i].mediaGeral > maiorMediaGeral) {

maiorMediaGeral = alunos[i].mediaGeral;

indiceMaiorMediaGeral = i;

}

}

printf("Aluno com maior media geral:\n");

printf("Nome: %s\n", alunos[indiceMaiorMediaGeral].nome);

printf("Media Geral: %.2f\n", alunos[indiceMaiorMediaGeral].mediaGeral);

printf("\n");

float menorMediaGeral = alunos[0].mediaGeral;

int indiceMenorMediaGeral = 0;

for (int i = 1; i < 5; ++i) {

if (alunos[i].mediaGeral < menorMediaGeral) {

menorMediaGeral = alunos[i].mediaGeral;

indiceMenorMediaGeral = i;

}

}

printf("Aluno com menor media geral:\n");

printf("Nome: %s\n", alunos[indiceMenorMediaGeral].nome);

printf("Media Geral: %.2f\n", alunos[indiceMenorMediaGeral].mediaGeral);

printf("\n");

for (int i = 0; i < 5; ++i) {

printf("Situacao do Aluno %s: ", alunos[i].nome);

if (alunos[i].mediaGeral >= 6.0) {

printf("Aprovado\n");

} else {

printf("Reprovado\n");

}

}

return 0;

}

**RECURSIVIDADE**

1) Faça um programa que calcule o fatorial de um número informado pelo usuário utilizando função recursiva.

#include <stdio.h>

int calcularFatorial(int n) {

if (n == 0) {

return 1;

}

else {

return n \* calcularFatorial(n - 1);

}

}

int main() {

int numero;

printf("Digite um numero para calcular o fatorial: ");

scanf("%d", &numero);

if (numero < 0) {

printf("O fatorial nao esta definido para numeros negativos.\n");

} else {

int resultado = calcularFatorial(numero);

printf("O fatorial de %d e: %d\n", numero, resultado);

}

return 0;

}

2) Faça um programa em C para calcular a soma, multiplicação, subtração e divisão de números dados pelo usuário na entrada utilizando funções recursivas.

#include <stdio.h>

double soma(double a, double b) {

return a + b;

}

double multiplicacao(double a, double b) {

if (b == 0) {

return 0;

}

return a + multiplicacao(a, b - 1);

}

double subtracao(double a, double b) {

if (b == 0) {

return a;

}

return subtracao(a - 1, b - 1);

}

double divisao(double a, double b) {

if (b == 0) {

printf("Erro: Divisao por zero nao e permitida.\n");

return 0;

}

return a / b;

}

int main() {

double num1, num2;

printf("Informe dois numeros:\n");

printf("Numero 1: ");

scanf("%lf", &num1);

printf("Numero 2: ");

scanf("%lf", &num2);

printf("\nResultados:\n");

printf("Soma: %.2f\n", soma(num1, num2));

printf("Multiplicacao: %.2f\n", multiplicacao(num1, num2));

printf("Subtracao: %.2f\n", subtracao(num1, num2));

printf("Divisao: %.2f\n", divisao(num1, num2));

return 0;

}

3) Crie um programa em Linguagem C que conte os dígitos de um determinado número usando recursão, por exemplo, se o usuário inserir o número 177, a saída esperada deve ser: O número digitado possui 3 dígitos.

#include <stdio.h>

int contarDigitos(int numero) {

if (numero == 0) {

return 0;

}

return 1 + contarDigitos(numero / 10);

}

int main() {

int numero;

printf("Informe um numero: ");

scanf("%d", &numero);

if (numero < 0) {

numero = -numero;

}

printf("O numero digitado possui %d digitos.\n", contarDigitos(numero));

return 0;

}

4) Crie um programa em C para inverter uma string usando recursão.

#include <stdio.h>

#include <string.h>

void inverterString(char \*str, int inicio, int fim) {

if (inicio < fim) {

char temp = str[inicio];

str[inicio] = str[fim];

str[fim] = temp;

inverterString(str, inicio + 1, fim - 1);

}

}

int main() {

char string[100];

printf("Informe uma string: ");

fgets(string, sizeof(string), stdin);

string[strcspn(string, "\n")] = '\0';

inverterString(string, 0, strlen(string) - 1);

printf("String invertida: %s\n", string);

return 0;

}

5) Escrever uma função recursiva que calcule o valor de a elevado a b.

#include <stdio.h>

double potencia(double a, int b) {

if (b == 0) {

return 1.0;

}

return a \* potencia(a, b - 1);

}

int main() {

double a;

int b;

printf("Informe a base (a): ");

scanf("%lf", &a);

printf("Informe o expoente (b): ");

scanf("%d", &b);

double resultado = potencia(a, b);

printf("%lf elevado a %d e %lf\n", a, b, resultado);

return 0;

}

**CONDICIONAIS LÓGICOS**

**1)** ( Switch Case) Suponha que você esteja desenvolvendo o sistema de um hotel que deva exibir mensagens na recepção para diferentes ações dos hóspedes, são elas: ● Fazer Check-in ● Chamar serviço de quarto ● Fazer pedido Dentro da opção check-in o usuário deverá informar seu nome, cpf, telefone, cidade e estado. Dentro da opção chamar o serviço de quarto o usuário deverá informar o seu nome e o número do quarto. Dentro da opção fazer pedido o usuário deverá receber um novo menu com opções de comida, bebida e finalizar pedido.

#include <stdio.h>

void fazerCheckIn() {

char nome[50], cpf[15], telefone[15], cidade[50], estado[3];

printf("Informe seu nome: ");

scanf(" %[^\n]", nome);

printf("Informe seu CPF: ");

scanf("%s", cpf);

printf("Informe seu telefone: ");

scanf("%s", telefone);

printf("Informe sua cidade: ");

scanf(" %[^\n]", cidade);

printf("Informe seu estado: ");

scanf("%s", estado);

printf("\nCheck-in realizado com sucesso!\n");

}

void chamarServicoQuarto() {

char nome[50];

int numeroQuarto;

printf("Informe seu nome: ");

scanf(" %[^\n]", nome);

printf("Informe o numero do quarto: ");

scanf("%d", &numeroQuarto);

printf("\nServico de quarto chamado para o quarto %d!\n", numeroQuarto);

}

void fazerPedido() {

int opcao;

while (1) {

printf("\n--- Menu de Pedidos ---\n");

printf("1. Comida\n");

printf("2. Bebida\n");

printf("3. Finalizar Pedido\n");

printf("Escolha a opcao desejada: ");

scanf("%d", &opcao);

switch (opcao) {

case 1:

printf("Voce escolheu Comida.\n");

break;

case 2:

printf("Você escolheu Bebida.\n");

break;

case 3:

printf("Pedido finalizado.\n");

return;

default:

printf("Opcao invalida. Tente novamente.\n");

}

}

}

int main() {

int opcao;

while (1) {

printf("\n--- Menu Principal ---\n");

printf("1. Fazer Check-in\n");

printf("2. Chamar Servico de Quarto\n");

printf("3. Fazer Pedido\n");

printf("4. Sair\n");

printf("Escolha a opcao desejada: ");

scanf("%d", &opcao);

switch (opcao) {

case 1:

fazerCheckIn();

break;

case 2:

chamarServicoQuarto();

break;

case 3:

fazerPedido();

break;

case 4:

printf("Sistema encerrado. Obrigado!\n");

return 0;

default:

printf("Opcao invalida. Tente novamente.\n");

}

}

return 0;

}

2) (IF) Escreva um programa para ler o número de lados de um polígono regular e a medida do lado (em cm). Calcular e imprimir o seguinte: ● Se o número de lados for igual a 3 escrever TRIÂNGULO e o valor da área. ● Se o número de lados for igual a 4 escrever QUADRADO e o valor da sua área. ● Se o número de lados for igual a 5 escrever PENTÁGONO.

#include <stdio.h>

#include <math.h>

int main() {

int numeroLados;

double medidaLado, area;

printf("Informe o numero de lados do poligono regular: ");

scanf("%d", &numeroLados);

printf("Informe a medida do lado (em cm): ");

scanf("%lf", &medidaLado);

if (numeroLados == 3) {

area = (sqrt(3) / 4) \* pow(medidaLado, 2);

printf("TRIANGULO\n");

printf("Area: %.2lf cm²\n", area);

} else if (numeroLados == 4) {

area = pow(medidaLado, 2);

printf("QUADRADO\n");

printf("Area: %.2lf cm²\n", area);

} else if (numeroLados == 5) {

printf("PENTAGONO\n");

} else {

printf("Poligono com numero de lados nao suportado.\n");

}

return 0;

}

**LAÇOS DE REPETIÇÃO**

1) (DO WHILE) Escreva um algoritmo que calcule a média dos números digitados pelo usuário, se eles forem pares. Termine a leitura se o usuário digitar zero (0).

#include <stdio.h>

int main() {

int numero, soma = 0, quantidade = 0;

do {

printf("Informe um numero (digite 0 para encerrar): ");

scanf("%d", &numero);

if (numero % 2 == 0 && numero != 0) {

soma += numero;

quantidade++;

}

} while (numero != 0);

if (quantidade > 0) {

double media = (double)soma / quantidade;

printf("Media dos numeros pares: %.2lf\n", media);

} else {

printf("Nenhum numero par foi digitado.\n");

}

return 0;

}

2) (DO WHILE) Construir um algoritmo que calcule a média aritmética de vários valores inteiros positivos. O final da leitura acontecerá quando for lido um valor negativo.

#include <stdio.h>

int main() {

int numero, soma = 0, quantidade = 0;

do {

printf("Informe um valor inteiro positivo (digite um valor negativo para encerrar): ");

scanf("%d", &numero);

if (numero >= 0) {

soma += numero;

quantidade++;

} else {

printf("Valor negativo inserido. Encerrando leitura.\n");

}

} while (numero >= 0);

if (quantidade > 0) {

double media = (double)soma / quantidade;

printf("Media aritmetica dos valores positivos: %.2lf\n", media);

} else {

printf("Nenhum valor positivo foi digitado.\n");

}

return 0;

}

3) (WHILE) Em uma eleição presidencial existem quatro candidatos. Os votos são informados através de códigos. Os dados utilizados para a contagem dos votos obedecem à seguinte codificação: ● 1,2,3,4 = voto para os respectivos candidatos; ● 5 = voto nulo; ● 6 = voto em branco; Elabore um algoritmo que leia o código do candidado em um voto. Calcule e escreva: ● total de votos para cada candidato; ● total de votos nulos; ● total de votos em branco; Como finalizador do conjunto de votos, tem-se o valor 0.

#include <stdio.h>

int main() {

int codigoVoto, totalCandidato1 = 0, totalCandidato2 = 0, totalCandidato3 = 0, totalCandidato4 = 0;

int totalNulos = 0, totalBrancos = 0;

printf("Informe o codigo do candidato (0 para encerrar): ");

while (1) {

scanf("%d", &codigoVoto);

if (codigoVoto == 0) {

break;

}

switch (codigoVoto) {

case 1:

totalCandidato1++;

break;

case 2:

totalCandidato2++;

break;

case 3:

totalCandidato3++;

break;

case 4:

totalCandidato4++;

break;

case 5:

totalNulos++;

break;

case 6:

totalBrancos++;

break;

default:

printf("Codigo de voto invalido. Voto nao contabilizado.\n");

}

printf("Informe o proximo codigo do candidato (0 para encerrar): ");

}

// Exibe os resultados

printf("\nResultados da eleicao:\n");

printf("Total de votos para Candidato 1: %d\n", totalCandidato1);

printf("Total de votos para Candidato 2: %d\n", totalCandidato2);

printf("Total de votos para Candidato 3: %d\n", totalCandidato3);

printf("Total de votos para Candidato 4: %d\n", totalCandidato4);

printf("Total de votos nulos: %d\n", totalNulos);

printf("Total de votos em branco: %d\n", totalBrancos);

return 0;

}

4) (FOR) Escrever um algoritmo que lê 5 valores para a, um de cada vez, e conta quantos destes valores são negativos, escrevendo esta informação.

#include <stdio.h>

int main() {

int valor;

int quantidadeNegativos = 0;

for (int i = 1; i <= 5; i++) {

printf("Informe o valor %d: ", i);

scanf("%d", &valor);

if (valor < 0) {

quantidadeNegativos++;

}

}

printf("Quantidade de valores negativos: %d\n", quantidadeNegativos);

return 0;

}