### 11. Verificar Número Primo: Determine se um número é primo #include <stdio.h>

### #include <stdbool.h>

### bool ePrimo(int num) {

### if (num <= 1) return false;

### if (num <= 3) return true;

### if (num % 2 == 0 || num % 3 == 0) return false;

### for (int i = 5; i \* i <= num; i += 6) {

### if (num % i == 0 || num % (i + 2) == 0) return false;

### }

### return true;

### }

### int main() {

### int num;

### printf("Digite um número: ");

### scanf("%d", &num);

### if (ePrimo(num)) {

### printf("O número é primo.\n");

### } else {

### printf("O número não é primo.\n");

### }

### return 0;

### }

### 12. Soma dos Números Naturais: Calcule a soma dos primeiros N números naturais #include <stdio.h>

### int somaNaturais(int n) {

### return n \* (n + 1) / 2;

### }

### int main() {

### int n;

### printf("Digite o valor de N: ");

### scanf("%d", &n);

### if (n < 0) {

### printf("O valor de N deve ser um número natural (não negativo).\n");

### } else {

### printf("A soma dos primeiros %d números naturais é: %d\n", n, somaNaturais(n));

### }

### return 0;

### }

**13. Fibonacci: Exiba os primeiros N termos da sequência de Fibonacci**#include <stdio.h>

void fibonacci(int n) {

int a = 0, b = 1, c;

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

printf("%d ", a);

c = a + b;

a = b;

b = c;

}

printf("\n");

}

int main() {

int n;

printf("Digite o número de termos: ");

scanf("%d", &n);

fibonacci(n);

return 0;

}

### 14. Inversão de Números: Inverta a ordem dos dígitos de um número #include <stdio.h>

### int inverterNumero(int num) {

### int invertido = 0;

### while (num != 0) {

### invertido = invertido \* 10 + num % 10;

### num /= 10;

### }

### return invertido;

### }

### int main() {

### int num;

### printf("Digite um número: ");

### scanf("%d", &num);

### printf("Número invertido: %d\n", inverterNumero(num));

### return 0;

### }

### 15. Cálculo de Potência: Calcule o valor de um número elevado a uma potência #include <stdio.h>

### int potencia(int base, int expoente) {

### int resultado = 1;

### for (int i = 0; i < expoente; i++) {

### resultado \*= base;

### }

### return resultado;

### }

### int main() {

### int base, expoente;

### printf("Digite a base: ");

### scanf("%d", &base);

### printf("Digite o expoente: ");

### scanf("%d", &expoente);

### printf("Resultado: %d\n", potencia(base, expoente));

### return 0;

### }

### 16. Palíndromo: Verifique se um número ou palavra é palíndromo #include <stdio.h>

### #include <string.h>

### #include <ctype.h>

### int ePalindromo(char str[]) {

### int inicio = 0;

### int fim = strlen(str) - 1;

### while (inicio < fim) {

### if (tolower(str[inicio]) != tolower(str[fim])) {

### return 0;

### }

### inicio++;

### fim--;

### }

### return 1;

### }

### int main() {

### char str[100];

### printf("Digite uma palavra ou frase: ");

### fgets(str, 100, stdin);

### str[strcspn(str, "\n")] = '\0'; // Remove o newline

### if (ePalindromo(str)) {

### printf("É um palíndromo.\n");

### } else {

### printf("Não é um palíndromo.\n");

### }

### return 0;

### }

### 17. Cálculo do MDC: Calcule o Máximo Divisor Comum (MDC) entre dois números #include <stdio.h>

### int mdc(int a, int b) {

### while (b != 0) {

### int temp = b;

### b = a % b;

### a = temp;

### }

### return a;

### }

### int main() {

### int a, b;

### printf("Digite dois números: ");

### scanf("%d %d", &a, &b);

### printf("MDC: %d\n", mdc(a, b));

### return 0;

### }

### 18. Cálculo do MMC: Calcule o Mínimo Múltiplo Comum (MMC) entre dois números #include <stdio.h>

### int mdc(int a, int b) {

### while (b != 0) {

### int temp = b;

### b = a % b;

### a = temp;

### }

### return a;

### }

### int mmc(int a, int b) {

### return (a \* b) / mdc(a, b);

### }

### int main() {

### int a, b;

### printf("Digite dois números: ");

### scanf("%d %d", &a, &b);

### printf("MMC: %d\n", mmc(a, b));

### return 0;

### }

### 19. Números Perfeitos: Verifique se um número é perfeito #include <stdio.h>

### int eNumeroPerfeito(int num) {

### int soma = 0;

### for (int i = 1; i <= num / 2; i++) {

### if (num % i == 0) {

### soma += i;

### }

### }

### return soma == num;

### }

### int main() {

### int num;

### printf("Digite um número: ");

### scanf("%d", &num);

### if (eNumeroPerfeito(num)) {

### printf("O número é perfeito.\n");

### } else {

### printf("O número não é perfeito.\n");

### }

### return 0;

### }

### 20. Ordenação de Vetor: Ordene um vetor de números em ordem crescente #include <stdio.h>

### void ordenarVetor(int vet[], int tamanho) {

### int temp;

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

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

### if (vet[i] > vet[j]) {

### temp = vet[i];

### vet[i] = vet[j];

### vet[j] = temp;

### }

### }

### }

### }

### int main() {

### int vet[100], tamanho;

### printf("Digite o tamanho do vetor: ");

### scanf("%d", &tamanho);

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

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

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

### }

### ordenarVetor(vet, tamanho);

### printf("Vetor ordenado:\n");

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

### printf("%d ", vet[i]);

### }

### printf("\n");

### return 0;

### }

### 21. Busca Linear: Implemente uma busca linear em um vetor #include <stdio.h>

### int buscaLinear(int vet[], int tamanho, int chave) {

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

### if (vet[i] == chave) {

### return i;

### }

### }

### return -1;

### }

### int main() {

### int vet[100], tamanho, chave;

### printf("Digite o tamanho do vetor: ");

### scanf("%d", &tamanho);

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

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

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

### }

### printf("Digite o valor a ser buscado: ");

### scanf("%d", &chave);

### int resultado = buscaLinear(vet, tamanho, chave);

### if (resultado != -1) {

### printf("Valor encontrado na posição %d.\n", resultado);

### } else {

### printf("Valor não encontrado.\n");

### }

### return 0;

### }

### 22. Busca Binária: Implemente uma busca binária em um vetor ordenado #include <stdio.h>

### int buscaBinaria(int vet[], int tamanho, int chave) {

### int esquerda = 0, direita = tamanho - 1;

### while (esquerda <= direita) {

### int meio = esquerda + (direita - esquerda) / 2;

### if (vet[meio] == chave) {

### return meio;

### }

### if (vet[meio] < chave) {

### esquerda = meio + 1;

### } else {

### direita = meio - 1;

### }

### }

### return -1;

### }

### int main() {

### int vet[100], tamanho, chave;

### printf("Digite o tamanho do vetor: ");

### scanf("%d", &tamanho);

### printf("Digite os elementos do vetor em ordem crescente:\n");

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

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

### }

### printf("Digite o valor a ser buscado: ");

### scanf("%d", &chave);

### int resultado = buscaBinaria(vet, tamanho, chave);

### if (resultado != -1) {

### printf("Valor encontrado na posição %d.\n", resultado);

### } else {

### printf("Valor não encontrado.\n");

### }

### return 0;

### }

### 23. Jogo de Adivinhação: Implemente um jogo de adivinhação onde o usuário tenta adivinhar um número secreto #include <stdio.h>

### #include <stdlib.h>

### #include <time.h>

### int main() {

### int numeroSecreto, chute, tentativas = 0;

### srand(time(NULL));

### numeroSecreto = rand() % 100 + 1; // Número aleatório entre 1 e 100

### printf("Bem-vindo ao jogo de adivinhação!\n");

### do {

### printf("Digite seu chute (1 a 100): ");

### scanf("%d", &chute);

### tentativas++;

### if (chute < numeroSecreto) {

### printf("Muito baixo! Tente novamente.\n");

### } else if (chute > numeroSecreto) {

### printf("Muito alto! Tente novamente.\n");

### } else {

### printf("Parabéns! Você acertou em %d tentativas.\n", tentativas);

### }

### } while (chute != numeroSecreto);

### 

### return 0;

### }

### 24. Conversão de Binário para Decimal: Converta um número binário para decimal #include <stdio.h>

### #include <math.h>

### int binarioParaDecimal(int binario) {

### int decimal = 0, base = 1;

### while (binario > 0) {

### int ultimoDigito = binario % 10;

### decimal += ultimoDigito \* base;

### base \*= 2;

### binario /= 10;

### }

### return decimal;

### }

### int main() {

### int binario;

### printf("Digite um número binário: ");

### scanf("%d", &binario);

### printf("Decimal: %d\n", binarioParaDecimal(binario));

### return 0;

### }

### 25. Conversão de Decimal para Binário: Converta um número decimal para binário #include <stdio.h>

### void decimalParaBinario(int decimal) {

### if (decimal == 0) {

### printf("0");

### return;

### }

### int binario[32];

### int i = 0;

### while (decimal > 0) {

### binario[i] = decimal % 2;

### decimal /= 2;

### i++;

### }

### for (int j = i - 1; j >= 0; j--) {

### printf("%d", binario[j]);

### }

### printf("\n");

### }

### int main() {

### int decimal;

### printf("Digite um número decimal: ");

### scanf("%d", &decimal);

### printf("Binário: ");

### decimalParaBinario(decimal);

### return 0;

### }

### 26. Soma de Matrizes: Some duas matrizes #include <stdio.h>

### #define MAX 10

### void somaMatrizes(int a[MAX][MAX], int b[MAX][MAX], int resultado[MAX][MAX], int linhas, int colunas) {

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

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

### resultado[i][j] = a[i][j] + b[i][j];

### }

### }

### }

### int main() {

### int a[MAX][MAX], b[MAX][MAX], resultado[MAX][MAX];

### int linhas, colunas;

### printf("Digite o número de linhas e colunas: ");

### scanf("%d %d", &linhas, &colunas);

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

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

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

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

### }

### }

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

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

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

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

### }

### }

### somaMatrizes(a, b, resultado, linhas, colunas);

### printf("Resultado da soma das matrizes:\n");

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

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

### printf("%d ", resultado[i][j]);

### }

### printf("\n");

### }

### 

### return 0;

### }

### 27. Multiplicação de Matrizes: Multiplique duas matrizes #include <stdio.h>

### #define MAX 10

### void multiplicarMatrizes(int a[MAX][MAX], int b[MAX][MAX], int resultado[MAX][MAX], int l1, int c1, int l2, int c2) {

### for (int i = 0; i < l1; i++) {

### for (int j = 0; j < c2; j++) {

### resultado[i][j] = 0;

### for (int k = 0; k < c1; k++) {

### resultado[i][j] += a[i][k] \* b[k][j];

### }

### }

### }

### }

### int main() {

### int a[MAX][MAX], b[MAX][MAX], resultado[MAX][MAX];

### int l1, c1, l2, c2;

### printf("Digite o número de linhas e colunas da primeira matriz: ");

### scanf("%d %d", &l1, &c1);

### printf("Digite o número de linhas e colunas da segunda matriz: ");

### scanf("%d %d", &l2, &c2);

### if (c1 != l2) {

### printf("Número de colunas da primeira matriz deve ser igual ao número de linhas da segunda matriz.\n");

### return 1;

### }

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

### for (int i = 0; i < l1; i++) {

### for (int j = 0; j < c1; j++) {

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

### }

### }

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

### for (int i = 0; i < l2; i++) {

### for (int j = 0; j < c2; j++) {

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

### }

### }

### multiplicarMatrizes(a, b, resultado, l1, c1, l2, c2);

### printf("Resultado da multiplicação das matrizes:\n");

### for (int i = 0; i < l1; i++) {

### for (int j = 0; j < c2; j++) {

### printf("%d ", resultado[i][j]);

### }

### printf("\n");

### }

### 

### return 0;

### }

### 28. Determinante de uma Matriz 2x2: Calcule o determinante de uma matriz 2x2 #include <stdio.h>

### int determinanteMatriz2x2(int a, int b, int c, int d) {

### return a \* d - b \* c;

### }

### int main() {

### int a, b, c, d;

### printf("Digite os elementos da matriz 2x2:\n");

### scanf("%d %d %d %d", &a, &b, &c, &d);

### printf("Determinante: %d\n", determinanteMatriz2x2(a, b, c, d));

### return 0;

### }

### 29. Contagem de Vogais: Conte o número de vogais em uma string #include <stdio.h>

### #include <ctype.h>

### int contarVogais(char str[]) {

### int cont = 0;

### for (int i = 0; str[i] != '\0'; i++) {

### char c = tolower(str[i]);

### if (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') {

### cont++;

### }

### }

### return cont;

### }

### int main() {

### char str[100];

### printf("Digite uma string: ");

### fgets(str, 100, stdin);

### str[strcspn(str, "\n")] = '\0'; // Remove o newline

### printf("Número de vogais: %d\n", contarVogais(str));

### return 0;

### }

### 30. Conversão de String para Inteiro: Implemente uma função que converte uma string para um número inteiro (similar ao atoi) #include <stdio.h>

### #include <ctype.h>

### int stringParaInteiro(const char \*str) {

### int resultado = 0;

### while (\*str) {

### if (isdigit(\*str)) {

### resultado = resultado \* 10 + (\*str - '0');

### } else {

### printf("Caracter inválido encontrado.\n");

### return -1;

### }

### str++;

### }

### return resultado;

### }

### int main() {

### char str[100];

### printf("Digite uma string numérica: ");

### fgets(str, 100, stdin);

### str[strcspn(str, "\n")] = '\0'; // Remove o newline

### int numero = stringParaInteiro(str);

### if (numero != -1) {

### printf("Número inteiro: %d\n", numero);

### }

### return 0;

### }