**Vetores**

**1. (Básico) - Exibir 10 números**

- **Descrição:** Crie um programa que declare um vetor para armazenar 10 números inteiros. Use um laço de repetição (for) para ler os 10 números do usuário e, em seguida, use outro laço (for) para exibi-los na tela.
- **Dica:** O primeiro for irá preencher o vetor, enquanto o segundo for irá iterar sobre os elementos já armazenados para mostrá-los.

**2. (Básico) - Soma de 5 números**

- **Descrição:** Declare um vetor para 5 números inteiros. Use um laço for para ler os 5 números. Durante a leitura, use uma variável separada para acumular a soma de todos os números e, ao final, exiba essa soma.
- **Dica:** Inicialize a variável de soma com 0 antes de começar o laço.

**3. (Básico) - Média de 8 números**

- **Descrição:** Declare um vetor para 8 números de ponto flutuante (float ou double). Use um laço for para ler os números e somá-los. Depois do laço, divida a soma total pela quantidade de números (8) e exiba a média.
- **Dica:** A variável para a soma deve ser do tipo float ou double para evitar a perda de precisão.

**4. (Intermediário) - Apenas números pares**

- **Descrição:** Leia 10 números inteiros para um vetor. Use um laço for para percorrer o vetor e, dentro dele, use uma estrutura condicional (if) para verificar se cada número é par. Se for, exiba o número.
- **Dica:** Use o operador de módulo (%) para verificar a paridade. Se numero % 2 == 0, o número é par.

**5. (Intermediário) - Maior e menor valor**

- **Descrição:** Leia 10 números inteiros para um vetor. Inicialize duas variáveis, uma para o maior valor e outra para o menor, com o primeiro elemento do vetor. Em seguida, use um laço for para percorrer o restante do vetor e atualizar as variáveis sempre que encontrar um valor maior ou menor, respectivamente. Ao final, exiba o maior e o menor.
- **Dica:** Inicialize maior = vetor[0] e menor = vetor[0] antes do laço para garantir que o primeiro elemento seja considerado.

**6. (Básico) - Exibir 5 nomes em ordem inversa**

- **Descrição:** Declare um vetor de strings para armazenar 5 nomes. Leia os 5 nomes do usuário. Use um laço for que comece do final do vetor e vá até o início (i = 4; i >= 0; i--) para exibir os nomes em ordem inversa.
- **Dica:** A sintaxe para um vetor de strings pode ser char nomes[5][50]; (5 nomes com até 50 caracteres cada).

**7. (Básico) - Contar números negativos**

- **Descrição:** Leia 6 números inteiros para um vetor. Crie uma variável contador_negativos inicializada com 0. Percorra o vetor com um laço for e, dentro dele, use uma instrução if para verificar se o número atual é menor que 0. Se for, incremente o contador. Ao final, exiba o total de números negativos.

**8. (Intermediário) - Buscar um número**

- **Descrição:** Leia 10 números para um vetor. Em seguida, peça ao usuário para informar um número a ser buscado. Percorra o vetor com um laço for e, em cada iteração, verifique se o número atual do vetor é igual ao número a ser buscado. Se encontrar, exiba uma mensagem de sucesso e você pode usar a função break para sair do laço. Se o laço terminar sem encontrar o número, exiba uma mensagem de falha.
- **Dica:** Use uma variável booleana (bool encontrado = false;) que muda para true se o número for achado. Depois do laço, verifique o valor dessa variável para exibir a mensagem correta.

**9. (Intermediário) - Vetores de pares e ímpares**

- **Descrição:** Leia 10 números inteiros. Declare dois novos vetores, um para armazenar os números pares e outro para os ímpares. Percorra os 10 números e, usando a verificação de paridade, armazene cada número em seu respectivo vetor. Ao final, exiba o conteúdo de ambos os vetores.
- **Dica:** Você precisará de dois contadores separados, um para o vetor de pares e outro para o de ímpares, para saber a próxima posição livre em cada vetor.

**10. (Intermediário) - Alunos aprovados**

- **Descrição:** Leia 10 notas de alunos para um vetor. Crie uma variável para contar os aprovados, inicializada com 0. Percorra o vetor de notas com um laço for. Use uma instrução if para verificar se cada nota é maior ou igual a 7. Se for, incremente o contador. Exiba o total de alunos aprovados.

---

**Matrizes**

**11. (Básico) - Exibir matriz 2x2**

- **Descrição:** Crie uma matriz 2x2. Use dois laços aninhados (for um dentro do outro) para ler os 4 elementos da matriz (2 linhas, 2 colunas). Depois, use outro par de laços aninhados para exibi-los na tela.

**12. (Básico) - Soma dos elementos da matriz**

- **Descrição:** Crie uma matriz 3x3. Leia os 9 elementos. Use dois laços aninhados para percorrer a matriz e somar todos os elementos em uma variável. Exiba a soma final.

**13. (Básico) - Média dos elementos da matriz**

- **Descrição:** Crie uma matriz 2x3. Leia os 6 elementos. Calcule a soma de todos os elementos e divida pelo total de elementos (6) para obter a média. Exiba a média.

**14. (Intermediário) - Diagonal principal**

- **Descrição:** Crie uma matriz 3x3. Leia os 9 elementos. Use um laço for para exibir apenas os elementos da diagonal principal.
- **Dica:** Os elementos da diagonal principal têm o mesmo índice de linha e coluna, ou seja, matriz[i][i].

**15. (Intermediário) - Soma da diagonal secundária**

- **Descrição:** Crie uma matriz 3x3. Leia os 9 elementos. Use um laço for para somar apenas os elementos da diagonal secundária.
- **Dica:** Os elementos da diagonal secundária seguem a regra matriz[i][j] onde i + j é igual ao número de linhas - 1. Para uma matriz 3x3, a soma é i + j = 2.

**16. (Intermediário) - Soma por linha**

- **Descrição:** Crie uma matriz 2x2. Leia os elementos. Use dois laços aninhados. O laço externo (for de linhas) deve iniciar a soma de cada linha com 0. O laço interno (for de colunas) deve somar os elementos da linha atual. Exiba a soma de cada linha separadamente.

**17. (Intermediário) - Maior elemento da matriz**

- **Descrição:** Crie uma matriz 3x3. Inicialize uma variável maior com o primeiro elemento (matriz[0][0]). Use dois laços aninhados para percorrer

todos os elementos da matriz, atualizando a variável maior sempre que um valor maior for encontrado. Exiba o maior valor ao final.

**18. (Intermediário) - Contar números pares**

- **Descrição:** Crie uma matriz 3x2. Use dois laços aninhados para ler os elementos. Dentro do laço, use uma instrução if com o operador de módulo (%) para contar quantos números são pares. Exiba a contagem final.

**19. (Intermediário) - Matriz com valores ao quadrado**

- **Descrição:** Crie duas matrizes 3x3. Leia os elementos para a primeira matriz. Use dois laços aninhados para percorrer a primeira matriz e, para cada elemento, calcule seu quadrado e armazene o resultado na posição correspondente da segunda matriz. Exiba a segunda matriz.

**20. (Intermediário) - Soma de duas matrizes**

- **Descrição:** Crie três matrizes 2x2. Leia os elementos para as duas primeiras matrizes. Use dois laços aninhados para somar os elementos correspondentes da primeira e da segunda matriz e armazenar o resultado na terceira matriz (matriz3[i][j] = matriz1[i][j] + matriz2[i][j]). Exiba a terceira matriz.

---

**Funções**

**21. (Básico) - Função que soma**

- **Descrição:** Crie uma função chamada somar. Ela deve receber dois parâmetros inteiros, somá-los e usar a instrução return para devolver o resultado. No programa principal (main), chame a função com dois números e exiba o valor retornado.

**22. (Intermediário) - Fatorial com função**

- **Descrição:** Crie uma função chamada fatorial que receba um número inteiro. Use um laço for dentro da função para calcular o fatorial e retorne o resultado. No programa principal, peça um número ao usuário e chame a função para calcular o fatorial, exibindo o resultado.

**23. (Básico) - Maior de dois números**

- **Descrição:** Crie uma função maior_numero que receba dois números inteiros. Use uma instrução if para comparar os dois números e retorne o maior deles.

**24. (Básico) - Média de três números**

- **Descrição:** Crie uma função calcular_media que receba três números de ponto flutuante (float). Dentro da função, some os três números e divida por 3.0 para retornar a média.

**25. (Básico) - Par ou ímpar**

- **Descrição:** Crie uma função eh_par que receba um número inteiro. Use o operador de módulo (%) para verificar se o número é par. A função deve retornar um valor booleano (true ou false) ou um inteiro (1 para par, 0 para ímpar) que indique o resultado.

**26. (Intermediário) - Soma de elementos de um vetor**

- **Descrição:** Crie uma função soma_vetor que receba um vetor de inteiros e seu tamanho como parâmetros. Use um laço for dentro da função para somar todos os elementos do vetor e retorne o resultado.

**27. (Intermediário) - Exibir matriz com função**

- **Descrição:** Crie uma função exibir_matriz que receba uma matriz 2x2 e exiba seus elementos. A função não precisa retornar nada (void).

**28. (Intermediário) - Maior valor em um vetor**

- **Descrição:** Crie uma função maior_valor_vetor que receba um vetor e seu tamanho. A função deve percorrer o vetor e retornar o maior valor encontrado.

**29. (Intermediário) - Contar caracteres de uma string**

- **Descrição:** Crie uma função contar_caracteres que receba uma string (char*). Use um laço de repetição (for ou while) para percorrer a string até encontrar o caractere nulo '\0' e conte a quantidade de caracteres. Retorne a contagem.

**30. (Básico) - Valor absoluto**

- **Descrição:** Crie uma função valor_absoluto que receba um número inteiro. Use uma instrução if para verificar se o número é negativo. Se for, multiplique-o por -1 e retorne o resultado. Se não, retorne o próprio número.

---

**Funções Recursivas**

**31. (Intermediário) - Fatorial recursivo**

- **Descrição:** Crie uma função fatorial_recursivo que receba um número n. A função deve ter um caso base: se n for 0 ou 1, ela retorna 1. No caso recursivo, ela retorna n * fatorial_recursivo(n - 1).

**32. (Intermediário) - Soma de 1 a n**

- **Descrição:** Crie uma função soma_recursiva que receba um número n. O caso base é se n for 1, a função retorna 1. O caso recursivo é retornar n + soma_recursiva(n - 1).

**33. (Intermediário) - Potência recursiva**

- **Descrição:** Crie uma função potencia_recursiva que receba uma base e um expoente. O caso base é se o expoente for 0, a função retorna 1. O caso recursivo é retornar base * potencia_recursiva(base, expoente - 1).

**34. (Básico) - Imprimir de 1 a n**

- **Descrição:** Crie uma função imprimir_crescente que receba um número n. O caso base é se n for 0. O caso recursivo é chamar imprimir_crescente(n - 1) e, em seguida, imprimir n.
- **Dica:** A ordem das instruções é importante para a impressão ser crescente.

**35. (Básico) - Imprimir de n até 1**

- **Descrição:** Crie uma função imprimir_decrescente que receba um número n. O caso base é se n for 0. O caso recursivo é imprimir n e, em seguida, chamar imprimir_decrescente(n - 1).
- **Dica:** A ordem das instruções é o oposto da questão 34.

**36. (Intermediário) - Fibonacci recursivo**

- **Descrição:** Crie uma função fibonacci que receba um número n para calcular o n-ésimo termo. O caso base é se n for 0 ou 1, a função retorna n. O caso recursivo é retornar fibonacci(n - 1) + fibonacci(n - 2).

**37. (Intermediário) - Soma de elementos do vetor recursivamente**

- **Descrição:** Crie uma função soma_vetor_recursiva que receba um vetor e seu tamanho n. O caso base é se n for 0, a função retorna 0. O caso recursivo é retornar vetor[n-1] + soma_vetor_recursiva(vetor, n - 1).

**38. (Intermediário) - Inverter vetor recursivamente**

- **Descrição:** Crie uma função inverter_vetor que receba um vetor e dois índices, inicio e fim. O caso base é quando inicio >= fim. O caso recursivo é

trocar os elementos nas posições inicio e fim e chamar a função novamente com inicio + 1 e fim - 1.

**39. (Intermediário) - Contar dígitos de um número recursivamente**

- **Descrição:** Crie uma função contar_digitos que receba um número inteiro. O caso base é se o número for menor que 10, a função retorna 1. O caso recursivo é retornar 1 + contar_digitos(numero / 10).

**40. (Avançado) - Encontrar o maior elemento em um vetor recursivamente**

- **Descrição:** Crie uma função maior_elemento_recursivo que receba um vetor e seu tamanho n. O caso base é se n for 1, a função retorna o único elemento. O caso recursivo é comparar o último elemento (vetor[n - 1]) com o resultado da chamada recursiva para o restante do vetor e retornar o maior dos dois.