

Essa lista de exercícios tem como objetivo principal desenvolver algoritmos a partir dos conteúdos abordados em sala de aula. Todos os exercícios também devem ser implementados em linguagem C. Nos programas que pedem para implementar apenas funções desenvolva também o programa principal (main) para testá-los.

1. Desenvolva uma função que receba dois pontos no plano cartesiano (x_1, y_1) , (x_2, y_2) , calcule e retorne a distância entre esses pontos.
2. Desenvolva uma função que receba um número n , calcule e retorne o somatório de 1 até n .
3. Desenvolva uma função que calcule se um número n é primo. Caso o número seja primo retorne 1 e 0, caso contrário.
4. Desenvolva uma função que acha as raízes de uma equação de segundo grau. Implemente uma função para calcular o valor do delta. Além disso, emita uma mensagem caso a equação não seja de segundo grau ou não tenha raízes reais.
5. Desenvolva uma função que calcule e retorne o fatorial de um número n .
6. Desenvolva uma função que calcule e retorne o n ésimo termo de fibonacci.
7. Desenvolva duas funções que recebam 3 números inteiros e retornem o Máximo Divisor Comum (MDC) e o Mínimo Múltiplo Comum (MMC) desses números. Crie um programa que leia 3 números inteiros e mostre o MDC e o MMC desses números.
8. Fazer uma função para calcular a raiz quadrada de um número positivo, baseado no método de aproximações sucessivas de Newton:
 - Seja $Y > 0$ o número e $N > 0$ a quantidade de aproximações.
 - A primeira aproximação para a raiz de Y é $X_1 = \frac{Y}{2}$.
 - As demais aproximações serão $X_{n+1} = \frac{X_n^2 + Y}{2X_n}$
9. Numa fábrica trabalham homens e mulheres divididos em três classes:
 - A os que fazem até 30 peças por mês.
 - B os que fazem de 31 a 35 peças por mês.
 - C os que fazem mais que 35 peças por mês.

A classe A recebe salário-mínimo. A classe B recebe salário-mínimo e mais 3% do salário-mínimo por peça acima das 30 iniciais. A classe C recebe salário-mínimo e mais 5% do salário-mínimo por peça acima das 30 iniciais. Desenvolva um programa com os seguintes requisitos:

- (a) Uma função que cadastre o nome, número de peças fabricadas por mês e o sexo de n funcionários. A quantidade de funcionários é definida pelo usuário.
- (b) Uma função que armazene em uma estrutura de dados o salário de todos os funcionários.
- (c) Uma função que calcule o valor total da folha de pagamento da fábrica.
- (d) Uma função que calcule a média de salários dos homens.
- (e) Uma função que calcule a média de peças fabricadas pelas mulheres em cada classe.
- (f) Uma função que exibe o nome, sexo e quantidade de peças fabricadas pelo funcionário de maior salário (suponha que não existe empate).

10. Utilizando funções, desenvolva um programa que exiba um menu com as seguintes opções:

1. $C = A + B$ (soma de matrizes).
2. A^t (matriz transposta).
3. $u = A * v$ (produto matriz por vetor).
4. $C = A * B$ (produto de matrizes).

O programa deve atender aos seguintes requisitos:

- (a) As matrizes e vetores são sempre gerados aleatoriamente dentro de um intervalo $[\text{min}, \text{max}]$ definido pelo usuário. Os valores gerados devem pertencer ao domínio dos números reais.
- (b) As matrizes e vetores não podem ser gerados antecipadamente, ou seja, eles devem ser gerados a partir da opção escolhida pelo usuário.
- (c) Para cada opção selecionado, as matrizes e vetores gerados devem ser impressos. Da mesma forma, os vetores e matrizes resultantes também devem ser impressos.
- (d) A consistência da operação deve ser avaliada, ou seja, o usuário deve ser obrigado a digitar as dimensões para vetores e matrizes que permitam a execução da operação. O programa deve tratar tal situação.

11. Desenvolva um algoritmo que verifica se uma data composta por 3 números inteiros (dia, mês e ano) é válida ou não. Para isso, implemente as seguintes funções:

- (a) Verificar se um ano é válido (aceitar anos entre 0 e 9999).
- (b) Verificar se um mês é válido (aceitar mês entre 1 e 12).
- (c) Verificar se um ano é bissexto.
- (d) Retornar quantos dias tem um mês (`dias_do_mês`) em um determinado ano.
- (e) Verificar se um dia é válido (aceitar dia entre 1 e `dias_do_mês`).

12. Modularize o programa do jogo da velha definido na lista 6. Para isto crie as seguintes funções:
- (a) Função para escrever o tabuleiro: void escreverTabuleiro(char matriz[][TAM]).
 - (b) Função para preencher o tabuleiro: void preencherTabuleiro(char matriz[][TAM], int movLinha, int movColuna).
 - (c) Função para verificar o estado do jogo (Xganhou, Oganhou, empate): void verificaJogo(char matriz[][TAM]).