Lista de Exercícios de Algoritmos e Programação em C

Foco: Comandos de Repetição e Decisão

Exercício 1: Calculadora com Menu Persistente

Crie um programa que funcione como uma calculadora. Ele deve apresentar um menu ao usuário e repetir a operação até que o usuário escolha a opção "Sair" (por exemplo, opção 0).

- Use um loop `do...while` para garantir que o menu seja exibido pelo menos uma vez e continue repetindo.
- Use um `switch` para tratar as opções do menu (Ex: 1. Somar, 2. Subtrair, 3. Multiplicar, 4. Dividir, 0. Sair).
- Dentro da opção de divisão (no `switch`), use um `if` para verificar se o divisor é zero. Se for, informe o usuário que a divisão por zero é impossível.
- Use o comando `break` para sair do loop `do...while` quando a opção 0 for escolhida.

Exercício 2 : Contagem de Dígitos Pares e Ímpares em um Inteiro

Peça ao usuário para digitar um número inteiro positivo (ex: 29384). O programa deve percorrer cada dígito desse número e contar quantos dígitos são pares e quantos são ímpares. (No exemplo 29384, seriam 3 pares e 2 ímpares).

- **Restrição:** Não use strings ou vetores.
- *Dica de "pensar":* Use um loop `while` que execute enquanto o número for maior que 0.
- Dentro do loop, use o operador módulo (`% 10`) para obter o último dígito.
- Use um `if` (ou um **if ternário** como desafio) para verificar se o dígito (`digito % 2`) é par ou ímpar e incrementar o contador correspondente.
- Use a divisão inteira (`/ 10`) para "remover" o último dígito do número antes da próxima iteração.

Exercício 3: Sequência de Fibonacci sem Vetor

Peça ao usuário um número `N` e imprima os `N` primeiros termos da sequência de Fibonacci (0, 1, 1, 2, 3, 5, 8...).

- **Restrição:** Você não pode usar vetores.
- *Dica de "pensar":* Você só precisa de três variáveis: `anterior`, `atual` e `proximo`.
- Use um loop `for` que rode `N` vezes.
- Dentro do loop, imprima o valor `anterior`. Calcule `proximo = anterior + atual`. Atualize as variáveis: `anterior = atual` e `atual = proximo`.
- Use `if` para tratar os casos base (os dois primeiros números).

Exercício 4: Somas Seletivas com `continue`

Peça ao usuário para digitar 10 números inteiros. O programa deve calcular e mostrar:

1. A soma de todos os números.
2. A soma apenas dos números pares.

3. A soma apenas dos números com valor superior a 100.

- Use um único loop `for` que rode 10 vezes.
- Dentro do loop, use um `if` para verificar se o número é ímpar (`numero % 2 != 0`). Se for, use `continue` para pular o resto do código *referente à soma dos pares* e ir para a próxima iteração.
- Use outro `if` para a soma dos maiores que 100.

Exercício 5 : Validação de Senha com Tentativas

Crie um programa que defina uma senha numérica (ex: `int senhaCorreta = 1234;`). O programa deve pedir ao usuário que digite a senha.

- O usuário tem no máximo **3 tentativas**.
- Use um loop `for` (de `i = 1` até 3) para controlar as tentativas.
- Dentro do loop, peça a senha. Use um `if` para verificar se a senha digitada é igual à `senhaCorreta`.
- Se a senha estiver correta, imprima "Acesso permitido" e use o comando `break` para sair do loop `for` imediatamente.
- Se a senha estiver incorreta, informe "Senha incorreta. Tentativas restantes: [X]".
- *Desafio:* Após o loop, use um `if` para verificar se o usuário *gastou* todas as tentativas (ex: se `i > 3`) e, nesse caso, imprima "Acesso bloqueado".

Exercício 6: Classificador de Faixa Etária (com Ternário)

Crie um programa que lê idades de um grupo de pessoas. O programa deve parar de ler quando uma idade negativa for inserida. Para cada idade lida, o programa deve classificar e contar:

- Crianças (0-12)
- Adolescentes (13-17)
- Adultos (18-59)
- Idosos (60+)
- Use um loop `while(1)` ou `while(true)`.
- Peça a idade. Use um `if` para verificar se a idade é negativa e, se for, use `break`.
- Use uma série de `if-else if-else` para incrementar os contadores corretos.
- **Desafio (Ternário):** Ao final (fora do loop), imprima o total de pessoas. Use um operador ternário para imprimir "pessoa" ou "pessoas". Ex: `printf("Total de %d %s.\n", total, (total == 1) ? "pessoa" : "pessoas");`

Exercício 7: Média de Notas da Turma (Flag de Parada)

Escreva um programa que lê as notas de um número indeterminado de alunos. O programa deve parar de ler notas quando o usuário digitar `-1`.

- Use um loop `while` (a condição pode ser `while (nota != -1)` ou `while(1)` com `break`).

- Você precisará de duas variáveis de acumulação: `somaDasNotas` e `contadorDeAlunos`.
- *Dica de "pensar":* O `scanf` deve vir *antes* da verificação do loop (se usar `while(nota != -1)`) ou *dentro* do loop (se usar `while(1)` com `break`).
- Ao final, fora do loop, use um `if` para verificar se `contadorDeAlunos` é maior que zero (para evitar divisão por zero) antes de calcular e imprimir a média.

Exercício 8 : Padrão de Asteriscos (Losango)

Peça ao usuário um número ímpar `N`. Use loops `for` aninhados para desenhar um losango de asteriscos com altura `N`.

Exemplo se `N=5`:

```
  *
 ***
*****
 ***
  *
```

- *Dica de "pensar":* Você precisará dividir o problema em duas partes (a parte de cima, incluindo o meio, e a parte de baixo), cada uma com seus loops aninhados.
- Para cada parte, você precisará de um loop `for` para os espaços em branco e outro loop `for` para os asteriscos.
- A lógica de quantos espaços e quantos asteriscos imprimir muda a cada linha (iteração do loop externo).

Exercício 9: Máximo Divisor Comum (MDC) com `while`

Peça ao usuário dois números inteiros positivos, `A` e `B`. Calcule o Máximo Divisor Comum (MDC) entre eles usando o algoritmo de Euclides (subtrações sucessivas ou módulo).

- *Método do Módulo (Recomendado):*
- Use um loop `while` que execute enquanto `B` for diferente de 0.
- Dentro do loop:
    1. Armazene `B` em uma variável temporária (ex: `temp = B`).
    2. Atualize `B` para ser o resto da divisão de `A` por `B` (`B = A % B`).
    3. Atualize `A` para ser o valor de `temp` (`A = temp`).
- Quando o loop terminar (quando `B` for 0), o valor de `A` será o MDC.

Exercício 10: Inversão de Número (Lógica de Repetição)

Peça ao usuário um número inteiro (ex: 4521). O programa deve inverter esse número e imprimi-lo (ex: 1254).

- **Restrição:** Sem strings ou vetores.
- *Dica de "pensar":* Você precisará usar o operador de módulo (`% 10`) para "arrancar" o último dígito e a divisão inteira (`/ 10`) para "remover" o último dígito.
- Use um loop `while` que continue enquanto o número original for maior que 0.

- Crie uma variável `numeroInvertido = 0`.
- Dentro do loop:
    1. `digito = numero % 10;`
    2. `numeroInvertido = (numeroInvertido * 10) + digito;`
    3. `numero = numero / 10;`
- Imprima `numeroInvertido` após o loop.