Exercício Loop for: Soma de Valores Acumulados

Este exercício envolve a criação de um programa em Java que calcula a soma dos números de 1 a 10 usando um loop for.

Instruções

1. **Inicialização:** Inicie uma variável chamada `soma` com o valor 0.
2. **Loop:** Execute um loop enquanto `i` for menor ou igual a 10.
3. **Acumulação:** A cada iteração, some o valor de `i` à variável `soma`.
4. **Incremento:** Incremente `i` em 1 a cada iteração.

Código do Exercício

```
public class SomaNumeros {
    public static void main(String[] args) {
        int soma = 0; // Inicializa a variável soma com 0
        for (int i = 1; i <= 10; i++) {
            soma += i; // Soma o valor de i à variável soma
        }
        System.out.println("A soma dos números de 1 a 10 é: " +
soma);
    }
}
```

Explicação do Código

- Iniciamos a variável `soma` com o valor 0.
- O loop verifica se `i` é menor ou igual a 10.
- A cada iteração, somamos o valor de `i` à variável `soma`.
- Após cada iteração, `i` é incrementado em 1.
- O loop continua até que `i` seja maior que 10, calculando a soma dos números de 1 a 10.

Exercício Loop while: Contagem de Doces

O objetivo deste exercício é criar um programa em Java que conte quantos doces o filho pode comer enquanto a condição estabelecida pela mãe for verdadeira (ou seja, no máximo 3 doces).

Instruções:

- Crie um programa em Java que conte quantos doces o filho pode comer enquanto a condição estabelecida pela mãe for verdadeira (ou seja, no máximo 3 doces).
- Passos
  - Inicialização da variável: Declare uma variável inteira chamada contadorDoces e inicialize-a com o valor 1.
  - Laço while: Utilize um laço while para repetir o seguinte bloco de código enquanto contadorDoces for menor ou igual a 3:

Código do Exercício:

```
public class ContagemDeDoces {
    public static void main(String[] args) {
        int contadorDoces = 1;

        while (contadorDoces <= 3) {
            System.out.println("Número de doces: " + contadorDoces);
            contadorDoces++;
        }

        System.out.println("Não pode comer mais doces.");
    }
}
```

Explicação:

1. Inicializamos a variável `contadorDoces` com o valor 1.
2. O laço `while` é executado enquanto `contadorDoces` for menor ou igual a 3.
3. A cada iteração, exibimos o número de doces consumidos.
4. Quando `contadorDoces` atinge 3, exibimos a mensagem "Não pode comer mais doces."

Exercício Complementar Loop do-while: Soma de Números Inteiros Positivos

Instruções

1. **Objetivo**: Escrever um programa Java que calcule a soma de todos os números inteiros positivos menores ou iguais a esse número.
2. **Passos**:
   - Defina uma variável com número inteiro positivo (valor 10, por exemplo).
   - Utilize um laço `do-while` para iterar e acumular a soma dos números inteiros positivos.
   - Exiba o resultado da soma.

Código do Exercício

```
public class SomaNumerosInteiros {
    public static void main(String[] args) {
        int numero = 10; // Exemplo de número inteiro positivo
        int soma = 0;

        // Calcula a soma dos números inteiros positivos
        do {
            soma += numero;
            numero--; // Decrementa o número para a próxima iteração
        } while (numero > 0);

        // Exibe o resultado da soma
        System.out.println("A soma dos números inteiros positivos é:
" + soma);
    }
}
```

Explicação do Código

1. Inicializamos a variável `numero` com o valor 10 (exemplo de número inteiro positivo).
2. Utilizamos um laço `do-while` para iterar enquanto o valor de `numero` for maior que zero.
3. Na iteração, acumulamos a soma dos números inteiros positivos na variável `soma`.

4. Decrementamos o valor de `numero` a cada iteração.
5. O laço continua até que `numero` seja menor ou igual a zero.
6. Por fim, exibimos o resultado da soma.

Exercício Complementar: Aplicando `continue` e `break`

Objetivo

Demonstrar a aplicação prática dos comandos `continue` e `break` em uma estrutura de repetição em Java, usando uma lista de códigos de carteirinha de convênio médico. O objetivo é entender como essas palavras-chave podem controlar o fluxo de execução de um laço `for` e como elas podem ser usadas para pular iterações ou interromper o loop com base em condições específicas.

Instruções

1. Declare a variável `codigoDeSaida` fora do laço `for` e defina seu valor como 9. Depois vá trocando os valores e analisando as saídas.
2. Inicie um laço `for` que comece em 1 e vá até 10, usando a variável `codigoCarteirinha`.
3. Verifique se o `codigoCarteirinha` é igual ao `codigoDeSaida` definido. Se verdadeiro, utilize `break` para interromper o loop e encerrar a execução do código.
4. Verifique se o `codigoCarteirinha` é aceito pelo hospital Santa Clara (códigos 3, 7 e 10). Se verdadeiro, imprima uma mensagem informando que o código é aceito e utilize `continue` para pular a iteração atual e continuar com a próxima.
5. Imprima uma mensagem genérica para os outros códigos, indicando que eles não são aceitos pelo hospital.

Código do Exercício em Java

```java
public class CarteirinhaConvenio {
    public static void main(String[] args) {
        // Variável código de saída que pode ser modificada pelo
aluno
        int codigoDeSaida = 9; // O aluno pode modificar este valor
conforme desejado

        // Laço que percorre códigos de carteirinha de 1 a 10
        for (int codigoCarteirinha = 1; codigoCarteirinha <= 10;
codigoCarteirinha++) {
            // Verifica se o codigoCarteirinha é igual ao
codigoDeSaida definido pelo aluno
            if (codigoCarteirinha == codigoDeSaida) {
                System.out.println("Código de saída encontrado,
interrompendo o loop.");
                break;
            }

            // Verifica se o codigoCarteirinha é aceito pelo hospital
Santa Clara
            if (codigoCarteirinha == 3 || codigoCarteirinha == 7 ||
codigoCarteirinha == 10) {
                System.out.println("O código da carteirinha " +
codigoCarteirinha + " é aceito pelo hospital Santa Clara.");
                continue;
            }

            // Mensagem genérica para outros códigos de carteirinha
            System.out.println("O código da carteirinha " +
codigoCarteirinha + " não é aceito pelo hospital Santa Clara.");
        }
    }
}
```

Saída Esperada

Quando o programa for executado com codigoDeSaida = 9, a saída será:

```
O código da carteirinha 1 não é aceito pelo hospital Santa Clara.
O código da carteirinha 2 não é aceito pelo hospital Santa Clara.
O código da carteirinha 3 é aceito pelo hospital Santa Clara.
O código da carteirinha 4 não é aceito pelo hospital Santa Clara.
O código da carteirinha 5 não é aceito pelo hospital Santa Clara.
O código da carteirinha 6 não é aceito pelo hospital Santa Clara.
O código da carteirinha 7 é aceito pelo hospital Santa Clara.
O código da carteirinha 8 não é aceito pelo hospital Santa Clara.
Código de saída encontrado, interrompendo o loop.
```

Explicação do Código

1. **Variável `codigoDeSaida`**: Declarada fora do laço `for` e definida como 9. O aluno pode modificar este valor para testar diferentes cenários.
2. **Laço for**: O laço `for` começa em 1 e vai até 10, usando a variável `codigoCarteirinha`.
3. **Verificação com if**: Se o `codigoCarteirinha` for igual ao `codigoDeSaida` (9), uma mensagem é impressa e o comando `break` é usado para interromper o laço.
4. **Verificação com if**: Se o `codigoCarteirinha` for 3, 7 ou 10, a mensagem de aceitação é impressa e o comando `continue` é usado para pular para a próxima iteração do laço.
5. **Mensagens Genéricas**: Para outros códigos de carteirinha, uma mensagem informando que não são aceitos é impressa.

Importante

Por que a carteirinha 10 não foi impressa? A carteirinha com o código 10 não foi impressa porque o loop foi interrompido ao encontrar a carteirinha com o código 9. A instrução `break` foi executada, terminando o loop antes de alcançar a iteração onde o código seria 10. Isso demonstra como o comando `break` pode interromper a execução do loop, impedindo que ele continue verificando os códigos subsequentes.