Comandos de Repetição

ECT3201 - Linguagem de Programação

Prof. Éverton Santi

Motivação

Nem todo problema pode ser resolvido com uma única decisão.

Em muitos casos, precisamos:

- repetir cálculos
- ler vários valores
- validar entradas
- executar ações até que uma condição mude

Quando Repetir?

Há dois cenários comuns:

1. **Não sabemos quantas repetições serão necessárias**
2. **Já sabemos quantas repetições devem ocorrer**

Essa diferença ajuda a escolher a estrutura mais adequada.

Dois Tipos de Repetição

Repetição não contada

- a continuidade depende de uma condição
- exemplos: `while` e `do-while`

Repetição contada

- já sabemos quantas iterações fazer
- exemplo: `for`

Estruturas de Repetição

Nesta aula veremos:

- `while`
- `do-while`
- `for`
- `break`
- `continue`

Comando `while`

Use `while` quando:

- a repetição depende de uma condição
- não sabemos quantas iterações serão necessárias

Leitura natural:

`enquanto (condição) faça`

Sintaxe do `while`

```
while (condicao) {
    // comandos
}
```

- a condição é testada antes da execução do bloco

Exemplo com `while`

- ler números inteiros até que o usuário digite `0`
- ao final, mostrar a soma dos valores digitados

Outro Exemplo com `while`

- ler uma nota até que ela esteja no intervalo de `0` a `10`
- mostrar uma mensagem quando a nota for válida

Cuidados com `while`

- a condição precisa poder se tornar falsa
- alguma variável importante deve ser atualizada
- leituras e atualizações fora de ordem podem gerar erro

Sem cuidado, o programa pode entrar em **laço infinito**.

Comando `do-while`

Use `do-while` quando:

- a repetição depende de uma condição
- o bloco deve executar **pelo menos uma vez**

Também é um caso de **repetição não contada**.

Sintaxe do `do-while`

```
do {
    // comandos
} while (condicao);
```

- a condição é testada depois da execução do bloco

Exemplo com `do-while`

- exibir um menu de opções
- repetir até que o usuário escolha `0`

`while` ou `do-while`?

Estrutura	Característica principal
`while`	testa a condição antes
`do-while`	executa o bloco e testa depois

Comando `break`

O comando `break` encerra imediatamente o laço.

Ele é útil quando:

- encontramos uma condição especial de parada
- queremos interromper a repetição antes do fim natural

Exemplo com `break`

- ler números inteiros
- parar imediatamente ao encontrar o primeiro valor negativo
- mostrar quantos valores positivos foram lidos antes disso

Comando `continue`

O comando `continue` interrompe apenas a iteração atual.

Depois disso:

- o laço segue para a próxima repetição

Exemplo com `continue`

- ler números inteiros
- somar apenas os valores positivos
- ignorar os valores negativos
- para quando usuário informar `0` (zero)

Comando `for`

Use `for` quando:

- já sabemos quantas vezes repetir
- queremos percorrer uma faixa de valores
- existe uma variável de controle bem definida

Esse é um caso de **repetição contada**.

Sintaxe do `for`

```
for (inicializacao; condicao; atualizacao) {
    // comandos
}
```

Partes principais:

- inicialização
- condição
- atualização

Exemplo com `for`

- mostrar os números de `1` até `10`

Outro Exemplo com `for`

- gerar a tabuada de um número `n` qualquer

Repetição Aninhada

Em alguns problemas, um laço pode aparecer dentro de outro.

Isso acontece quando precisamos:

- repetir blocos de repetições
- organizar saídas em linhas e colunas
- gerar tabelas ou padrões

Exemplo com Repetição Aninhada

- mostrar um retângulo de `4` linhas por `6` colunas usando o caractere `*`, como no exemplo abaixo:

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

Comparando as Estruturas

Estrutura	Tipo de repetição	Uso mais comum
`while`	não contada	condição testada antes
`do-while`	não contada	bloco executa ao menos uma vez
`for`	contada	número de iterações já conhecido

Exercício 1

Classifique cada problema como:

- **repetição não contada**
- **repetição contada**

1. Ler senhas até acertar
2. Mostrar os números de `1` até `100`
3. Exibir um menu até escolher sair
4. Calcular a soma dos `20` primeiros números

Exercício 2

Faça um programa que:

- leia idades inteiras até que o usuário digite `-1`
- ao final, mostre quantas idades correspondem a maiores de idade
- mostre também a média das idades válidas lidas

Considere que pelo menos uma idade válida será informada.

Exercício 3

Faça um programa que:

- leia um número inteiro positivo `n`
- mostre todos os números pares de `2` até `n`
- ao final, mostre quantos números pares foram exibidos

Exercício 4

Faça um programa que:

- leia um número inteiro `n`
- calcule o fatorial de `n`

$$n! = 1 \times 2 \times 3 \times \cdots \times n$$

- mostre o resultado final

Exercício 5

Faça um programa que:

- leia um número inteiro positivo `n`
- verifique quantos divisores inteiros positivos esse número possui
- mostre ao final se ele é um número primo ou não

Exercício 6

Faça um programa que:

- mostre um quadrado de tamanho `n` usando o caractere `#`
- o valor de `n` deve ser lido pelo programa
- cada linha do quadrado deve conter exatamente `n` caracteres