Estruturas de Controle de Fluxo em C#

Introdução e Engajamento

Imagine que seu programa é uma estrada. **Estruturas de controle de fluxo** são as placas, semáforos e bifurcações que dizem por onde seguir, quando parar, quando repetir um caminho ou seguir outro. Elas dão vida, inteligência e adaptabilidade ao seu código — sem elas, o programa executaria instrução por instrução, sempre na mesma ordem, sem reagir ao que acontece!

Provocações para Engajamento

- **Você já tomou uma decisão do tipo "Se chover, pego guarda-chuva; senão, vou de bicicleta"?** Escreva em uma frase o último "se... então..." que você tomou na vida real.
- **Consegue imaginar situações em que você repete uma ação várias vezes até atingir um objetivo?** Pense em exemplos do cotidiano (ex: repetir exercícios até acabar a série, ou contar moedas até somar certo valor). Anote um desses exemplos.
- **Desafio mental:** Se tivesse que ensinar um robô a escovar os dentes, como faria para ele repetir os movimentos corretos até terminar?

O que são Estruturas de Controle de Fluxo?

Estruturas de controle de fluxo são comandos especiais de uma linguagem de programação que **determinam o caminho** que seu código vai seguir com base em decisões (condições) e repetições (loops).

Decisão

if, else, else if, switch

Repetição

for, while, do-while

Em C#, estes comandos permitem que seu programa tome decisões e execute tarefas repetitivas de forma eficiente.

Estruturas de Decisão

if e **else**

Permitem executar trechos de código apenas se uma condição for verdadeira (ou falsa).

Sintaxe básica:

```
if (condição) {
    // Código executado se condição for
    // verdadeira
} else {
    // Código executado se condição for
    // falsa
}
```

Exemplo:

```
int idade = 18;
if (idade >= 18) {
    Console.WriteLine("Maior de idade");
} else {
    Console.WriteLine("Menor de idade");
}
```

Dica Profissional:

- Você pode usar apenas if (sem else) quando só se importa com a condição verdadeira.
- Evite "if"s encadeados muito complexos — deixe o código legível.

Estrutura else if

else if

Permite testar várias condições diferentes, uma após a outra.

Sintaxe:

```
if (condição1) {
    // Código 1
} else if (condição2) {
    // Código 2
} else {
    // Código 3
}
```

Exemplo:

```
int nota = 7;
if (nota >= 9) {
    Console.WriteLine("Excelente");
} else if (nota >= 7) {
    Console.WriteLine("Bom");
} else {
    Console.WriteLine("Regular");
}
```

Condição 1	Condição 2	else
Verifica a primeira condição	Se a primeira falhar, verifica a segunda	Executa se todas as condições falharem

Dica Profissional:

- Use else if para deixar a lógica clara e evitar múltiplos if independentes.
- Sempre coloque um else ao final para tratar situações "fora do previsto".

Estrutura switch

switch

Seleciona entre múltiplos caminhos de execução com base no valor de uma variável. Muito útil quando há vários casos possíveis para um mesmo valor.

Sintaxe:

```
switch (variavel) {
    case valor1:
        // Código para valor1
        break;
    case valor2:
        // Código para valor2
        break;
    // ...
    default:
        // Código se nenhum valor anterior for igual
        break;
}
```

Exemplo:

```
int diaSemana = 3;
switch (diaSemana) {
    case 1:
        Console.WriteLine("Domingo");
        break;
    case 2:
        Console.WriteLine("Segunda");
        break;
    // ...
    default:
        Console.WriteLine("Outro dia");
        break;
}
```

Dica Profissional:

- Sempre use break após cada caso para evitar o chamado "fall-through" (execução involuntária dos próximos casos).
- O default é uma boa prática para tratar valores inesperados.

Estruturas de Repetição (Loops)

for

Usado quando se sabe **quantas vezes** algo deve se repetir.

Sintaxe:

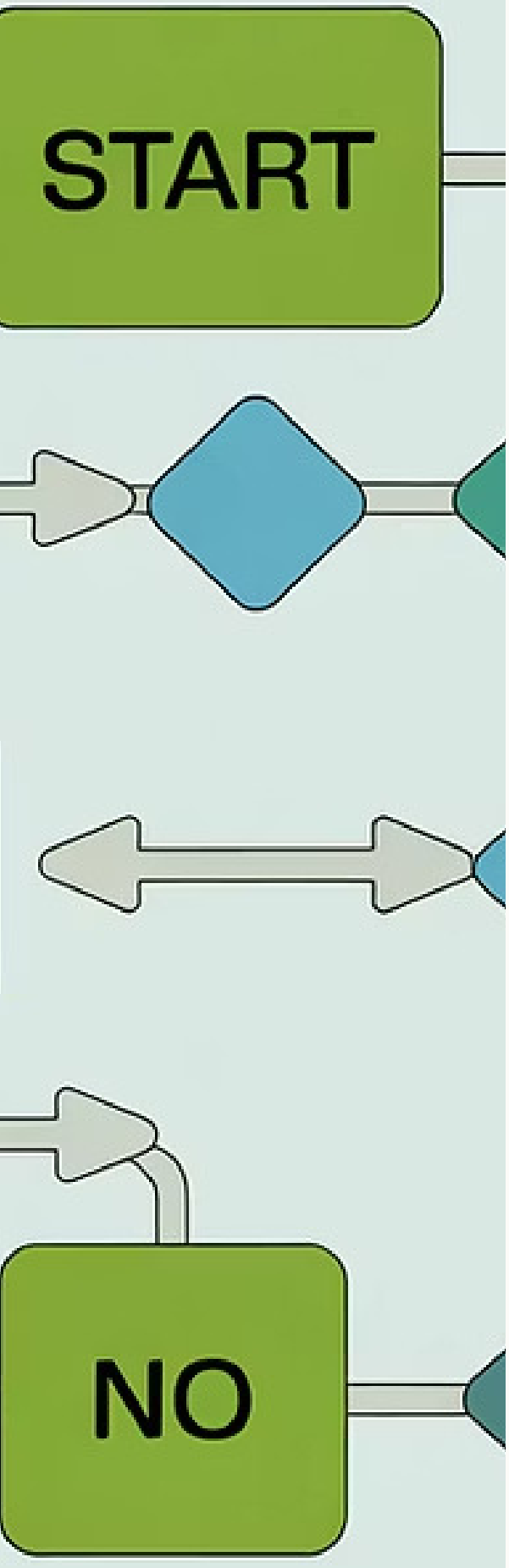
```
for (inicialização; condição; incremento) {
    // Código repetido enquanto a condição for verdadeira
}
```

Exemplo:

```
for (int i = 0; i < 10; i++) {
    Console.WriteLine("Iteração: " + i);
}
```

Dica Profissional:

Muito usado para percorrer listas, vetores e fazer contagem.



Estrutura while

while

Repete um bloco de código **enquanto** uma condição for verdadeira. Use quando não sabe ao certo quantas vezes precisará repetir.

Sintaxe:

```
while (condição) {
    // Código repetido
}
```

Exemplo:

```
int contador = 0;
while (contador < 5) {
    Console.WriteLine("Contador: " +
      contador);
    contador++;
}
```

Dica Profissional:

Cuidado para não esquecer de atualizar a variável de condição, evitando loops infinitos!

Estrutura do-while

do-while

Executa o bloco de código **ao menos uma vez**, depois repete enquanto a condição for verdadeira.

Sintaxe:

```
do {
    // Código repetido
} while (condição);
```

Exemplo:

```
int numero = 0;
do {
    Console.WriteLine("Número: " + numero);
    numero++;
} while (numero < 3);
```

Execução do Código

Primeiro, o bloco de código é executado uma vez

Verificação da Condição

Depois, a condição é verificada

Repetição

Se a condição for verdadeira, volta para o primeiro passo

Dica Profissional:

Útil quando você quer garantir que o código rode pelo menos uma vez (ex: pedir uma senha até estar correta).

Boas Práticas e Dicas Profissionais

Clareza e Simplicidade

Prefira estruturas claras e simples a "gambiarras" com muitos if ou loops aninhados.

Documentação

Sempre comente trechos de lógica complexa.

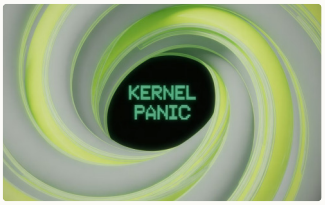
Nomenclatura

Use nomes de variáveis autoexplicativos.

Segurança

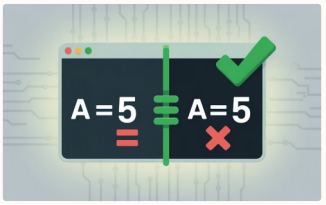
Lembre-se: loops infinitos travam seu programa! Sempre tenha certeza de que a condição vai mudar em algum momento.

Armadilhas e Erros Comuns



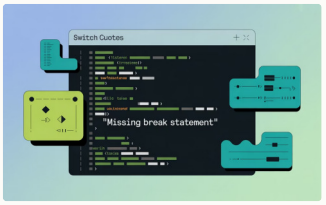
Loop Infinito

Esqueceu de atualizar a condição ou incrementou errado.



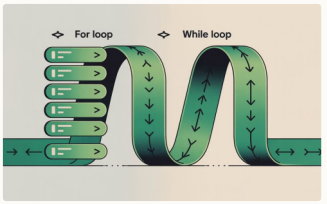
Condições Mal Formuladas

Usar = ao invés de == em condições (if (x = 10) faz atribuição!).



Falta de break em switch

Pode causar execução de casos não desejados.



Confundir for com while

Use o for para repetições contadas, while para repetições dependentes de condições dinâmicas.

Exercícios Práticos

1 Decisões do Dia a Dia

Reescreva uma decisão do seu dia a dia usando if/else em C#.

2 Números Pares

Implemente um for que conte de 1 até 20, mostrando apenas os números pares.

3 Economizando

Crie um while que simule alguém economizando dinheiro até atingir um valor desejado.

4 Notas com switch

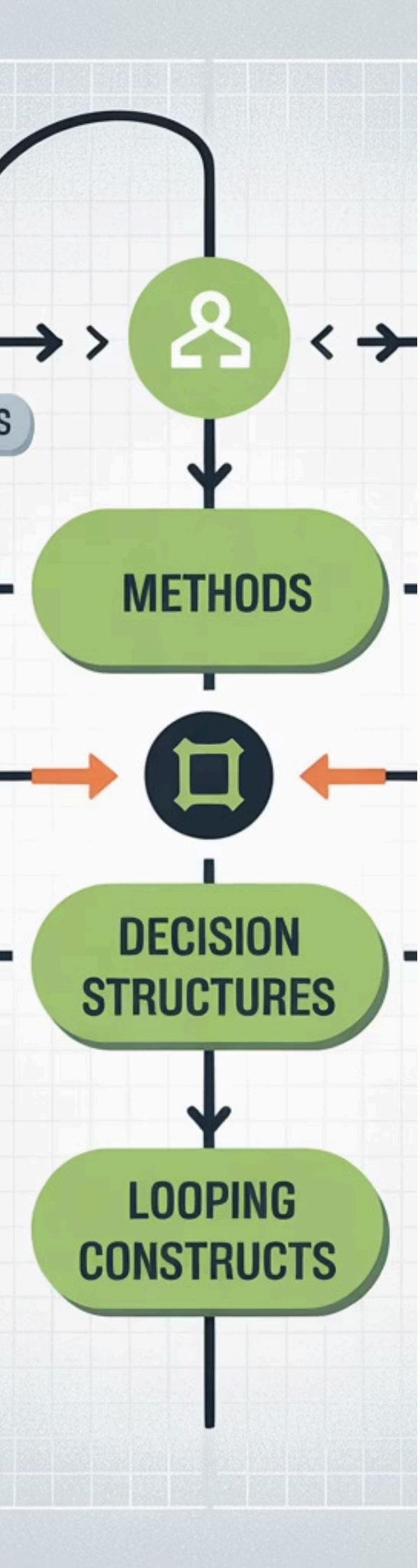
Utilize switch para mostrar uma mensagem diferente para cada nota (A, B, C, D, F) recebida em uma avaliação.

5 Senha Correta

Altere o exemplo do do-while para pedir repetidamente que o usuário digite a senha "1234" até acertar. (Use lógica mental ou pseudo-código se não for executar no computador)

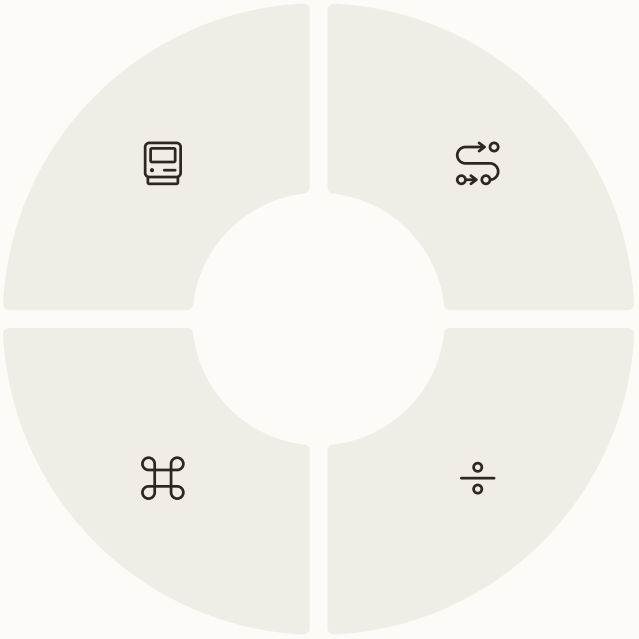
Resumo Visual (Quadro-Resumo)

Estrutura	Usado para...	Exemplo (simplificado)
if/else	Decisão binária ou múltipla	if (idade >= 18) { ... } else { ... }
else if	Múltiplas alternativas	if (...) else if (...) else { ... }
switch	Muitas opções de valor fixo	switch (opcao) { case 1: ... }
for	Repetição contada	for (int i = 0; i < 10; i++) { ... }
while	Repetição condicional (desconhecida)	while (condição) { ... }
do-while	Repetição, executando ao menos 1 vez	do { ... } while (condição);



Conexão com POO

Em programação orientada a objetos, **controle de fluxo é essencial dentro de métodos**: toda lógica de decisão e repetição estará presente em métodos de classes e objetos. Entender bem essas estruturas agora vai facilitar a escrita de métodos mais inteligentes e eficientes futuramente!



Classes

Definem a estrutura dos objetos

Métodos

Contêm estruturas de controle de fluxo

Decisões

Determinam comportamentos dos objetos

Repetições

Processam coleções de dados