




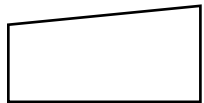
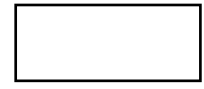
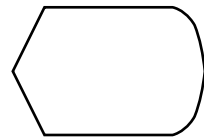
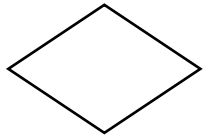
# Fluxograma

Analisaremos agora o método de representação de algoritmos denominado fluxograma.

Conceitualmente um fluxograma é um tipo de diagrama, e pode ser entendido como uma representação esquemática de um processo, constitui uma representação gráfica que ilustra de forma descomplicada a seqüência de execução dos elementos que o compõem. Podemos entendê-lo, na prática, como a documentação dos passos necessários para a execução de um processo qualquer.

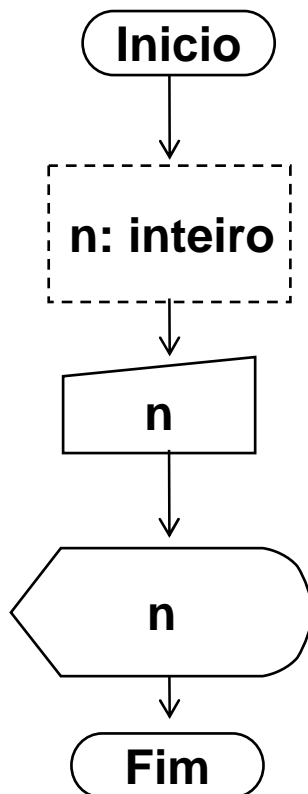
Veremos agora alguns símbolos empregados na construção de fluxogramas.

# Fluxograma

Símbolo	Nome	Descrição
	<b>Terminador</b>	Indica o início e o fim do fluxo do algoritmo.
	<b>Seta de fluxo</b>	Indica o sentido do fluxo de execução do algoritmo. É através dela que os símbolos do fluxograma são conectados.
	<b>Declaração</b>	Delimita a seção de declaração de variáveis.
	<b>Entrada de dados</b>	Corresponde à instrução de entrada de dados através do teclado.
	<b>Atribuição</b>	Símbolo utilizado para indicar cálculos e atribuição de valores.
	<b>Saída de dados</b>	Corresponde à instrução de saída de dados. Os dados serão exibidos na tela do computador.
	<b>Desvio condicional</b>	Divide o fluxo do programa em dois caminhos, dependendo do teste lógico que fica dentro do losango.

# Exemplo de Fluxograma

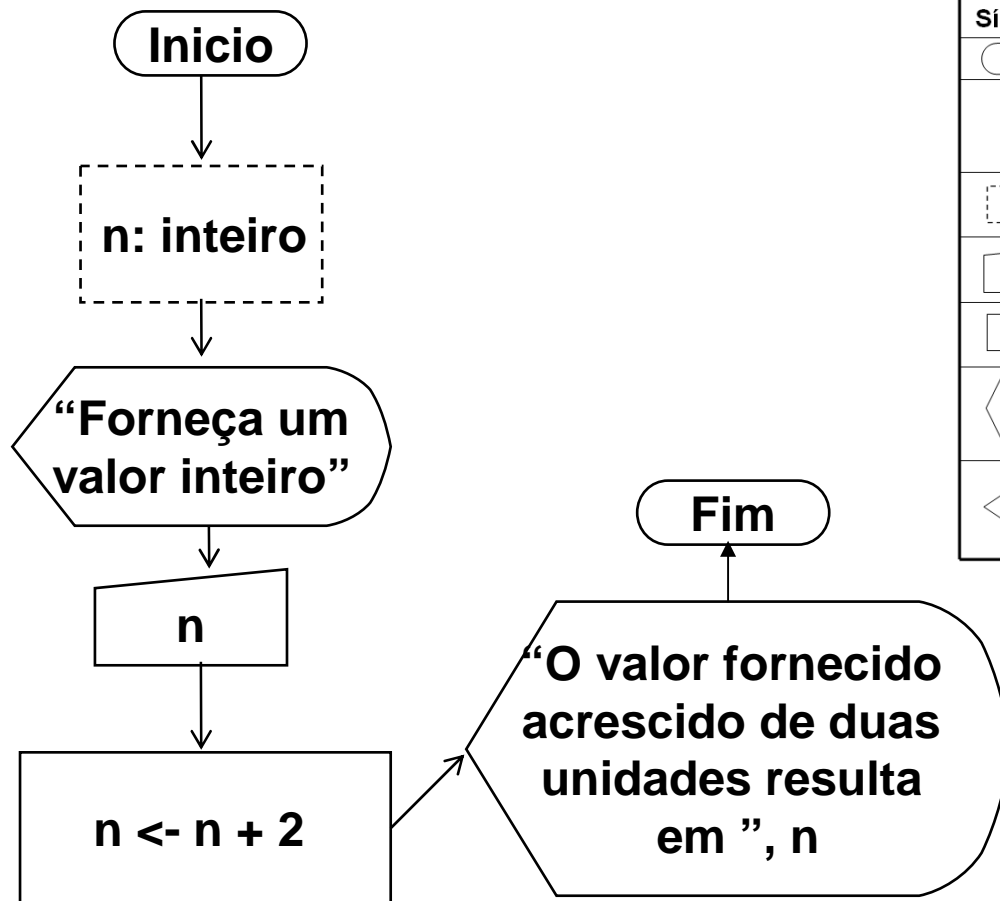
De forma similar à análise feita com pseudocódigo, iniciaremos nossa análise por um fluxograma que efetua a leitura, através do teclado, de um valor inteiro e o retorna no monitor.



Símbolo	Nome	Descrição
	Terminador	Indica o início e o fim do fluxo do algoritmo.
	Seta de fluxo	Indica o sentido do fluxo de execução do algoritmo. É através dela que os símbolos do fluxograma são conectados.
	Declaração	Delimita a seção de declaração de variáveis.
	Entrada de dados	Corresponde à instrução de entrada de dados através do teclado.
	Atribuição	Símbolo utilizado para indicar cálculos e atribuição de valores.
	Saída de dados	Corresponde à instrução de saída de dados. Os dados serão exibidos na tela do computador.
	Desvio condicional	Divide o fluxo do programa em dois caminhos, dependendo do teste lógico que fica dentro do losango.

# Exemplo de Fluxograma

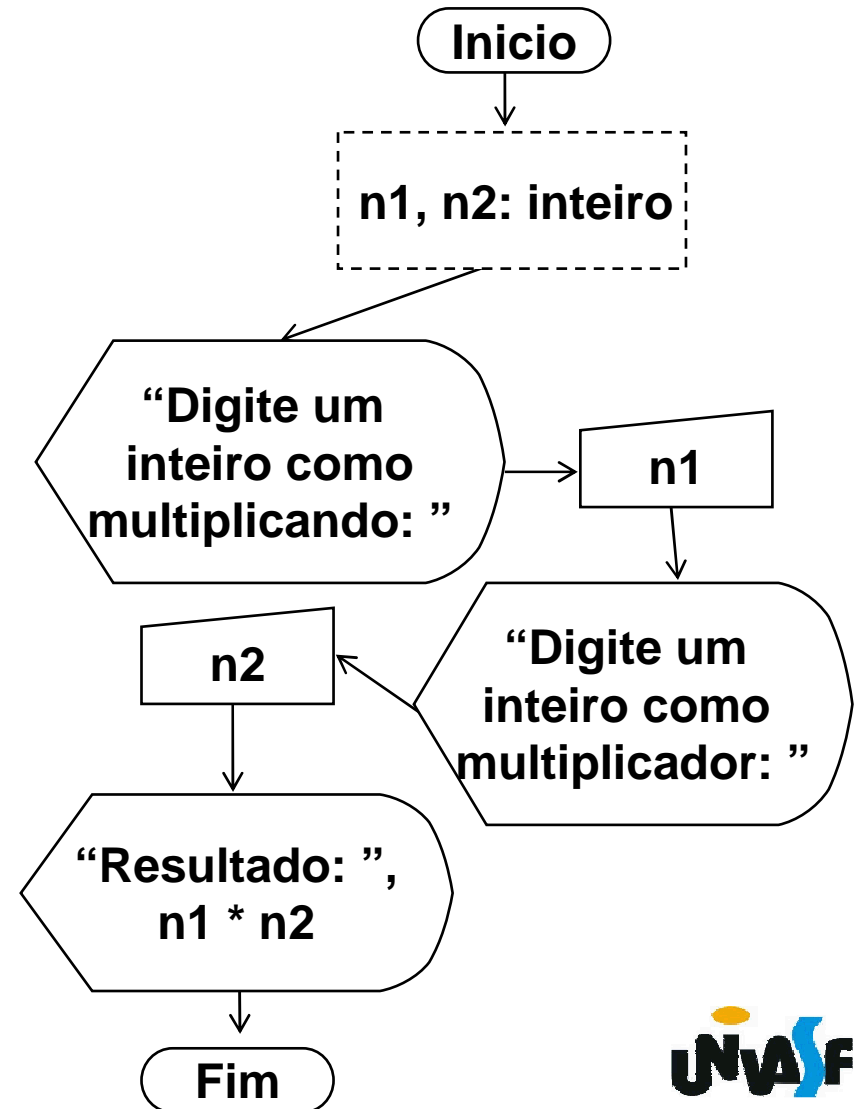
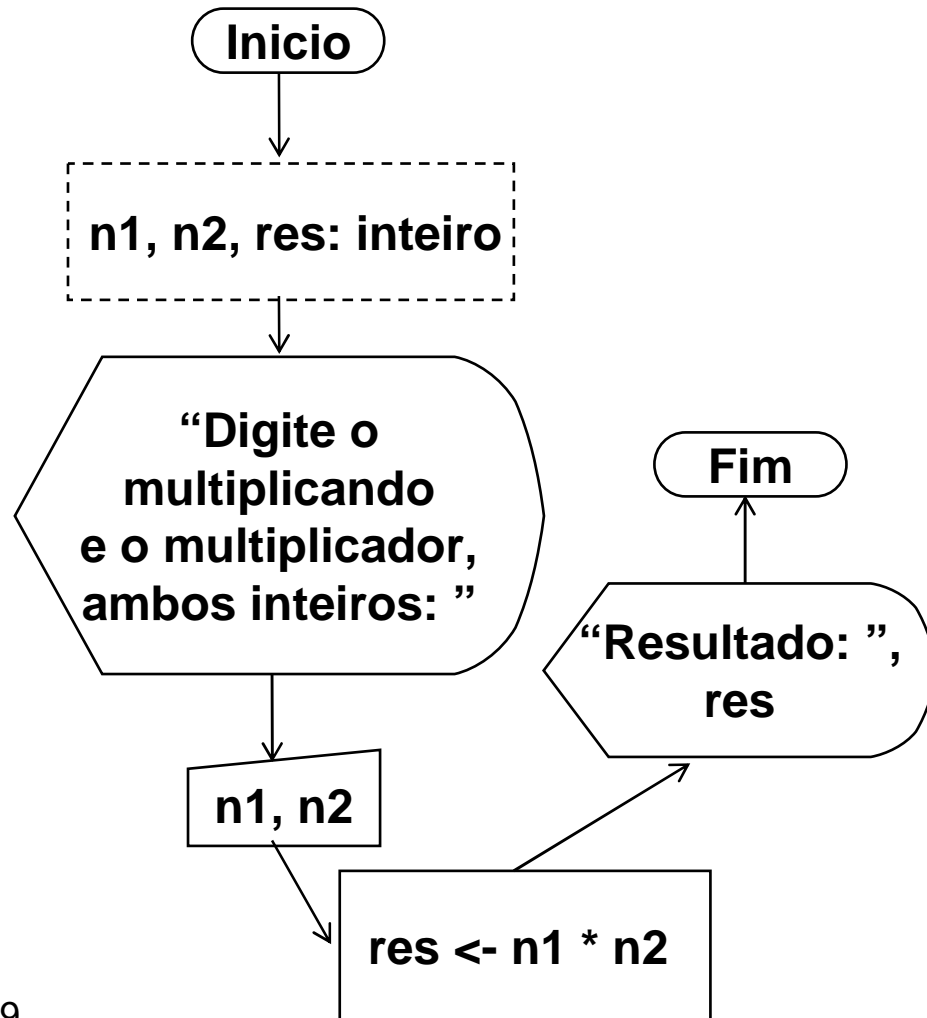
Observaremos agora um fluxograma que recebe um valor inteiro, através da entrada padrão, e acresce duas unidades a este exibindo o resultado na saída padrão.



Símbolo	Nome	Descrição
	Terminador	Indica o início e o fim do fluxo do algoritmo.
	Seta de fluxo	Indica o sentido do fluxo de execução do algoritmo. É através dela que os símbolos do fluxograma são conectados.
	Declaração	Delimita a seção de declaração de variáveis.
	Entrada de dados	Corresponde à instrução de entrada de dados através do teclado.
	Atribuição	Símbolo utilizado para indicar cálculos e atribuição de valores.
	Saída de dados	Corresponde à instrução de saída de dados. Os dados serão exibidos na tela do computador.
	Desvio condicional	Divide o fluxo do programa em dois caminhos, dependendo do teste lógico que fica dentro do losango.

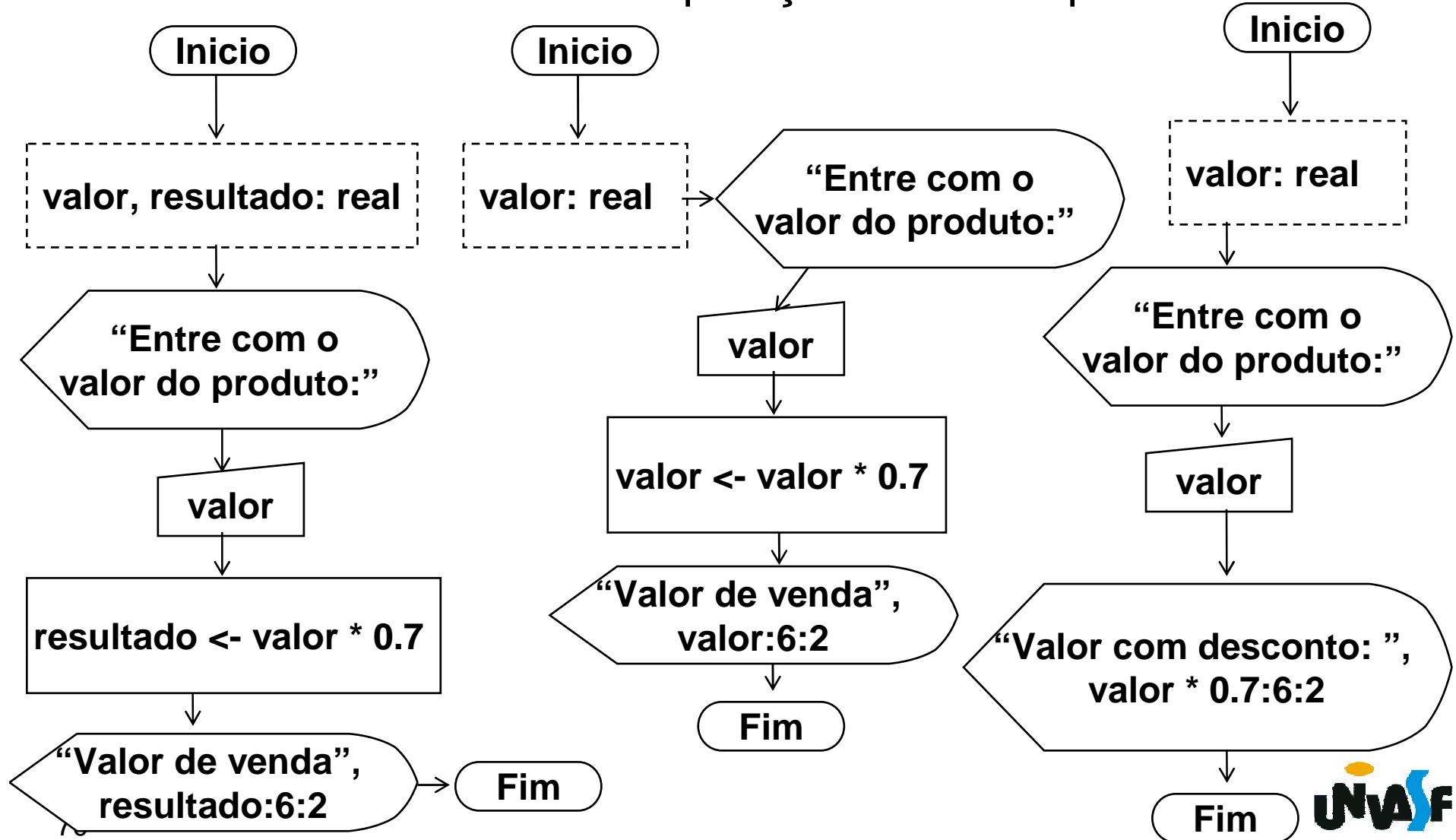
# Exercício de Fluxograma

Construa um fluxograma para obter o resultado da multiplicação de dois números inteiros quaisquer fornecidos pelo usuário.



# Exercício de Fluxograma

Gere um fluxograma que aplique um desconto de 30% sobre o valor de um produto, recebido como entrada, e retorne o resultado da manipulação na saída padrão.



## Estruturas de Controle de Fluxo

Os algoritmos desenvolvidos até o momento constituem uma seqüência de ações que sempre são executadas em sua totalidade indiferente de qual(is) seja(m) o(s) valor(es) da(s) entrada(s).

Contudo para a resolução de determinados problemas ou para a execução de determinadas tarefas é necessária a realização de um conjunto distinto de ações e este conjunto é definido com base em uma análise da(s) entrada(s).

Um exemplo simples de uma destas situações é um algoritmo capaz de efetuar o cálculo do imposto de renda devido por um determinado contribuinte. Neste caso dependendo da quantidade de dependentes, do valor de sua renda e outras fatores o cálculo será feito de formas distintas.

# Estruturas de Controle de Fluxo

Em função do que foi mencionado foram criadas as estruturas de controle de fluxo, as quais são fundamentais para a construção de algoritmos complexos. Estas permitem que o programador especifique a seqüência de instruções que será executada.

## 1. Instrução condicional (simples)

Sintaxe: ...

se (*<expressão-lógica>*) entao

*<seqüência-de-comandos>*

fimse

...



## Estruturas de Controle de Fluxo

**Pseudocódigo/Exercício** – Construa o pseudocódigo de um algoritmo para obter o resultado da divisão de dois números inteiros quaisquer.

**algoritmo “exercício”**

**var** n1, n2: inteiro

**res:** real

**inicio**

**escreva** (“Digite o dividendo inteiro: ”)

**leia** (n1)

**escreva** (“Digite o divisor inteiro: ”)

**leia** (n2)

**se** (n2<>0) **entao**

**res** <- n1 / n2

**escreva** (“Resultado da divisão: ”, res)

**fimse**

**fimalgoritmo**

## Estruturas de Controle de Fluxo

**Pseudocódigo/Exercício** – Construa o pseudocódigo de um algoritmo para obter o resultado da divisão de dois números inteiros quaisquer.

**algoritmo “exercício b”**

**var n1, n2: inteiro**

**res: real**

**inicio**

**escreva (“Digite o dividendo inteiro: ”)**

**leia (n1)**

**escreva (“Digite o divisor inteiro: ”)**

**leia (n2)**

**se (n2<>0) entao**

**res <- n1 / n2**

**escreva (“Resultado da divisão: ”, res)**

**fimse**

**se (n2=0) entao**

**escreva (“Impossível dividir!”)**

**fimse**

**fimalgoritmo**

# Estruturas de Controle de Fluxo

## 1. Instrução condicional (composta)

Sintaxe:

```
...  
se (<expressão-lógica>) entao  
    <seqüência-de-comandos-1>  
senao  
    <seqüência-de-comandos-2>  
fimse  
...
```

//**Observação:** esta forma é denominada instrução condicional composta.

## Estruturas de Controle de Fluxo

**Pseudocódigo/Exercício** – Construa o pseudocódigo de um algoritmo para obter o resultado da divisão de dois números inteiros quaisquer.

**algoritmo “exercício c”**

**var n1, n2: inteiro**

**res: real**

**inicio**

**escreva (“Digite o dividendo inteiro: ”)**

**leia (n1)**

**escreva (“Digite o divisor inteiro: ”)**

**leia (n2)**

**se (n2=0) entao**

**escreva (“Impossível dividir!”)**

**senao**

**res <- n1 / n2**

**escreva (“Resultado da divisão: ”, res)**

**fimse**

**fimalgoritmo**

## Estruturas de Controle de Fluxo

**Pseudocódigo/Exercício** – Construa o pseudocódigo de um algoritmo para obter o resultado da divisão de dois números inteiros quaisquer.

**algoritmo “exercício c”**

**var n1, n2: inteiro**

**res: real**

**inicio**

**escreva (“Digite o dividendo inteiro: ”)**

**leia (n1)**

**escreva (“Digite o divisor inteiro: ”)**

**leia (n2)**

**se (n2<>0) entao**

**res <- n1 / n2**

**escreva (“Resultado da divisão: ”, res)**

**senao**

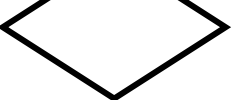
**escreva (“Impossível dividir!”)**

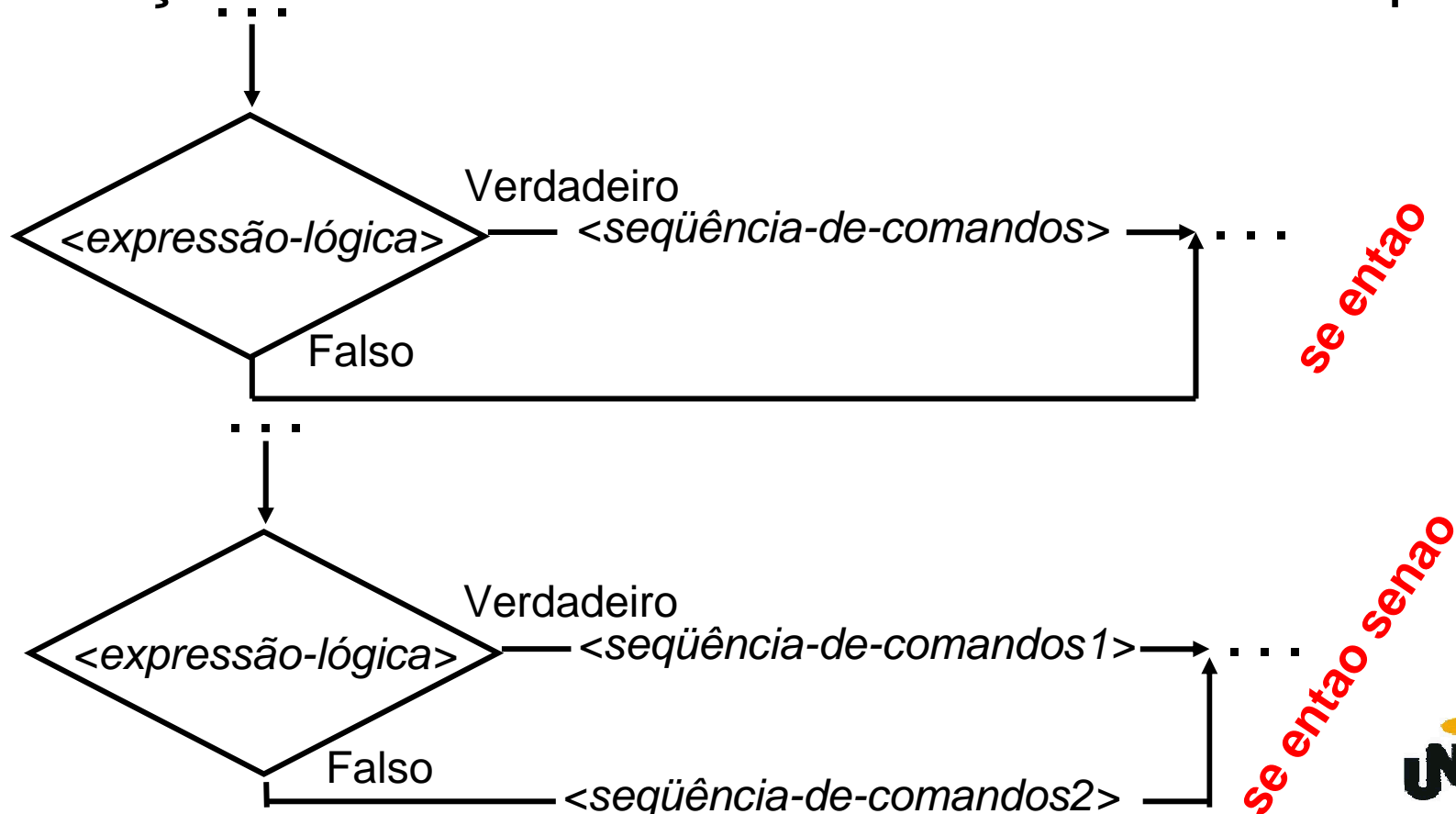
**fimse**

**fimalgoritmo**

# Estruturas de Controle de Fluxo

## Fluxograma

Vimos o símbolo  quando falamos sobre fluxograma. Este símbolo nos permite implementar a instrução “se entao” e “se entao senao”. Exemplo:



# Estruturas de Controle de Fluxo

**Fluxograma/Exercício** – Com base no que foi exposto construa um fluxograma para obter o resultado da divisão de dois números inteiros quaisquer.

