

Instruções Condicionais



Teste de Condição

As **instruções condicionais** (ou desvios condicionais) permitem executar um conjunto de ações (instruções) ou outro conjunto de ações, alternativamente, sob o controlo de uma condição (teste de condição).

Tipos de instruções condicionais (em linguagem algorítmica):

- Seleção simples: `se`
- Seleção composta: `se...senao`
 `se...senao se`
- Seleção de múltipla escolha: `escolha...caso`

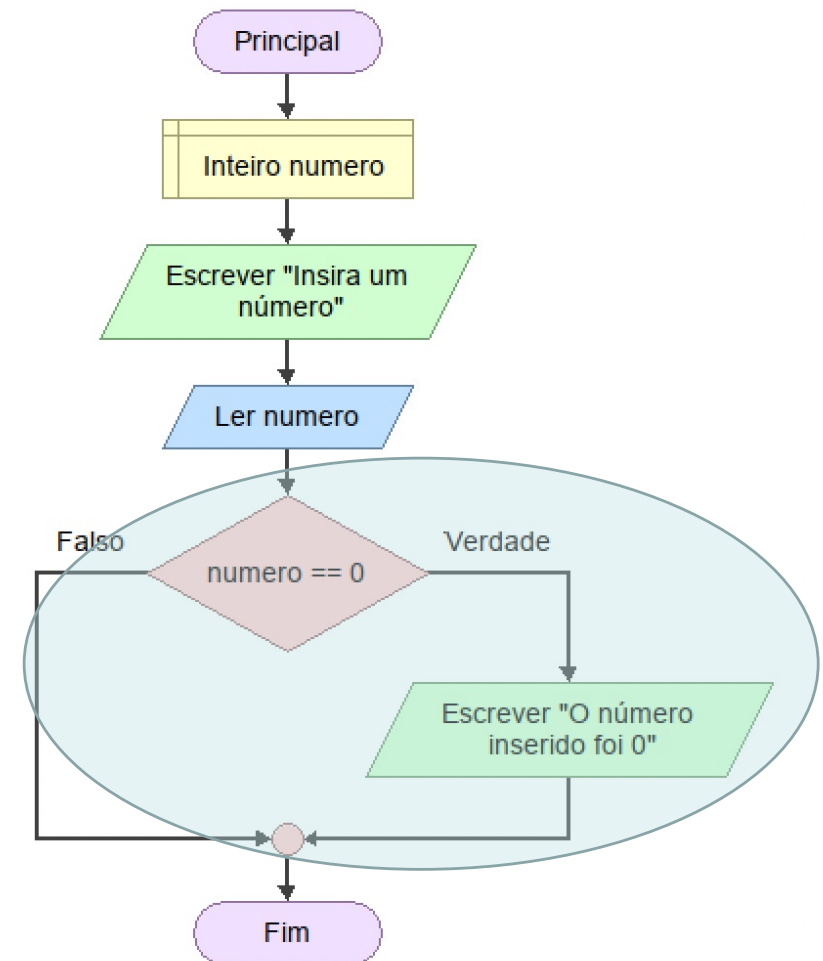
Seleção Simples

```
se( <expressão_lógica> )  
    <instrução>
```

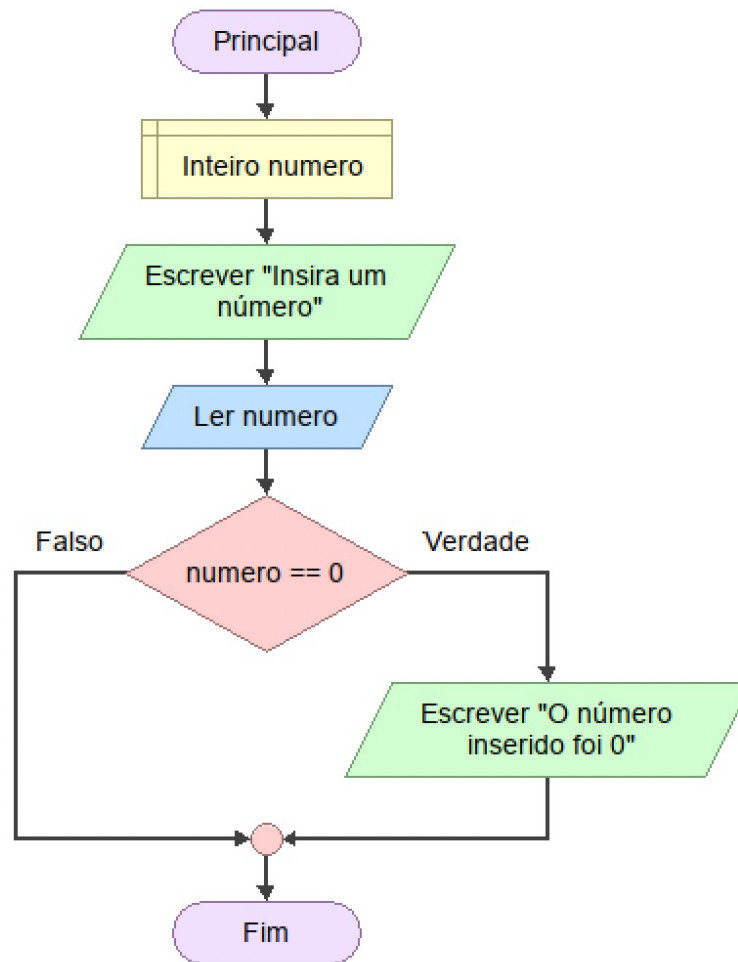
A instrução só é executada se a expressão lógica for verdadeira.

```
se(numero == 0)  
    escreva("o número inserido foi 0")
```

Se a expressão for falsa, vai diretamente para o fim



Seleção Simples

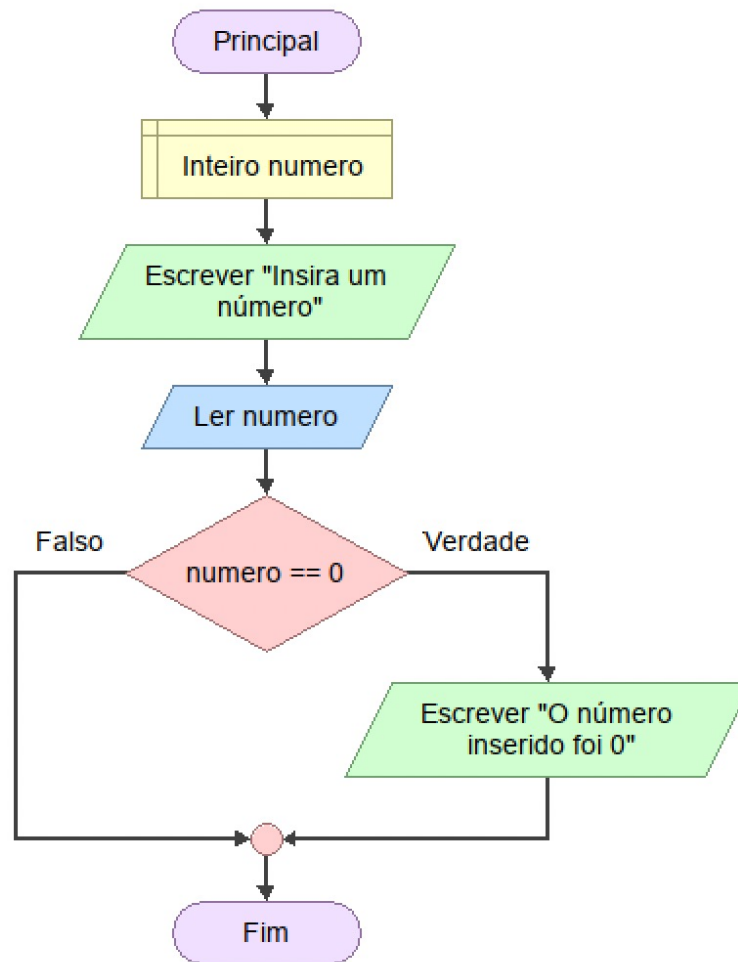


Portugol

```
se( numero == 0 )  
| escreva( "O número inserido foi 0"
```

Seleção Simples

Python



```
if numero == 0:  
    print("O número inserido foi 0")
```

Seleção Simples

Programas Completos em Portugol e Python

Portugol

```
programa {  
    funcao inicio() {  
        inteiro numero  
        escreva( "Insira um número ")  
        leia( numero )  
        se( numero == 0 )  
            escreva( "O número inserido foi 0")  
    }  
}
```

Python

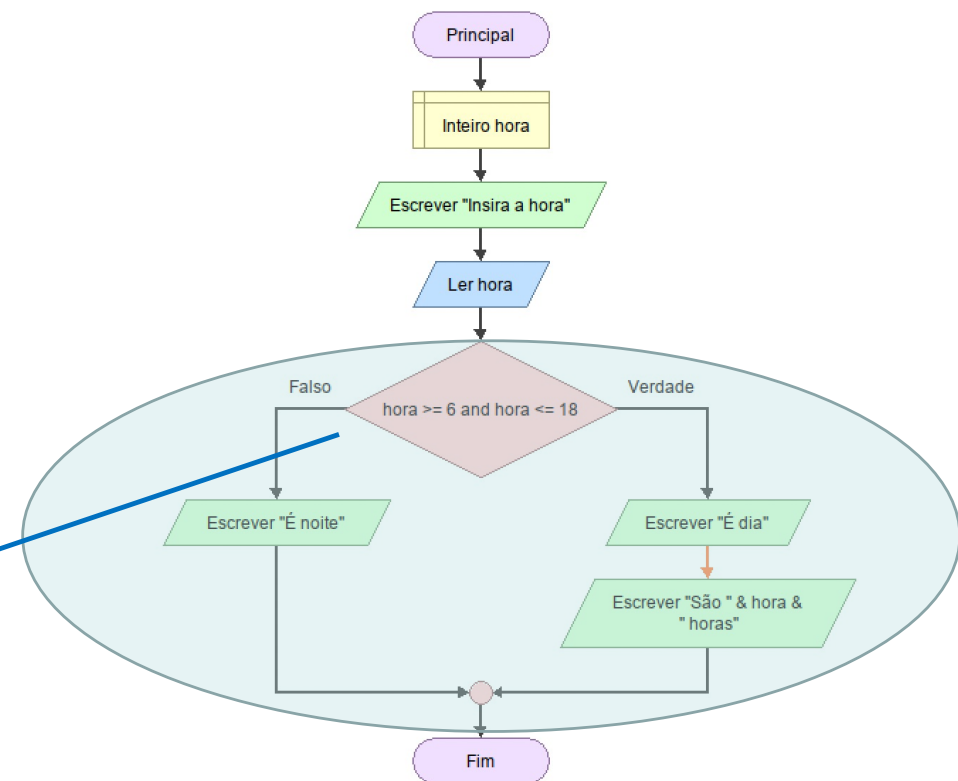
```
print("Insira um número")  
numero = int(input())  
if numero == 0:  
    print("O número inserido foi 0")
```

Seleção Composta (ou seleção de duas vias)

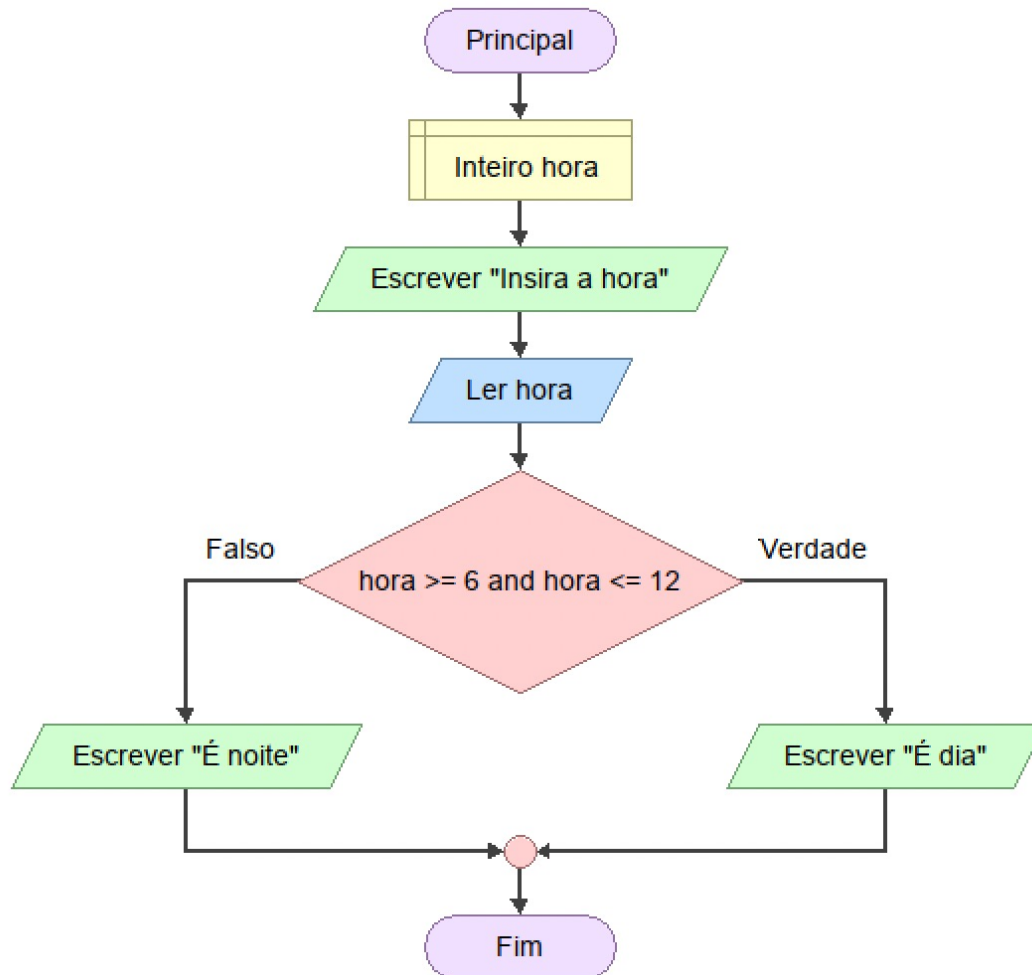
```
se( <expressão_lógica> )  
    <instrução_1>  
senao  
    <instrução_2>
```

se a expressão_lógica for verdadeira, é executada a instrução_1;
se for falsa, é executada a instrução_2

```
se(hora >= 6 e hora <= 18)  
    escreva("É dia")  
senao  
    escreva("É noite")
```



Seleção Composta

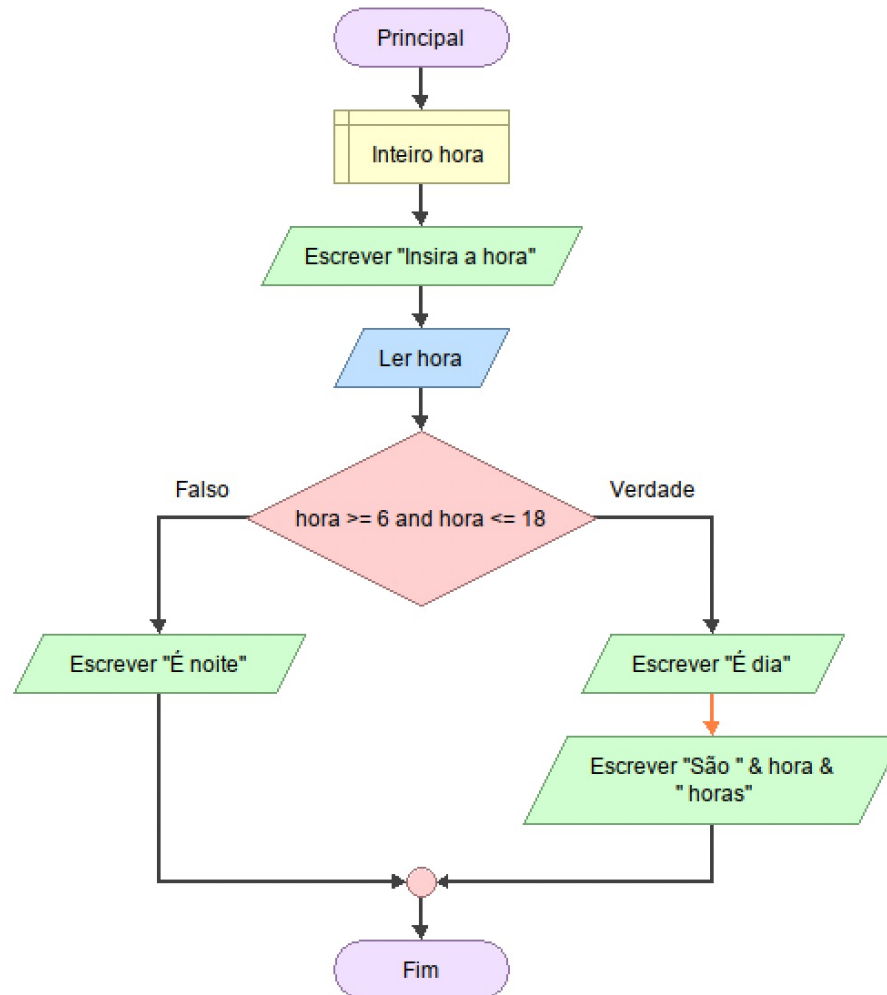


Portugol

```
se (hora >= 6 e hora <= 18)
{
    escreva ("É dia")
}
senao
{
    escreva ("É noite")
}
```


Seleção Composta

Python



```
print("Insira a hora")
hora = int(input())
if hora >= 6 and hora <= 18:
    print("É dia")
    print("São " + str(hora) + " horas")
else:
    print("É noite")
```

Portugol

```
programa {  
    funcao inicio() {  
        inteiro hora  
  
        escreva ("Digite a hora: ")  
        leia (hora)  
  
        se (hora >= 6 e hora <= 18)  
        {  
            escreva ("É dia")  
        }  
        senao  
        {  
            escreva ("É noite")  
        }  
    }  
}
```

Seleção Composta

Programas Completos em Portugol e Python

Python

```
print("Insira a hora")  
hora = int(input())  
if hora >= 6 and hora <= 12:  
    print("É dia")  
else:  
    print("É noite")
```

As instruções podem ser **simples** ou **compostas**:

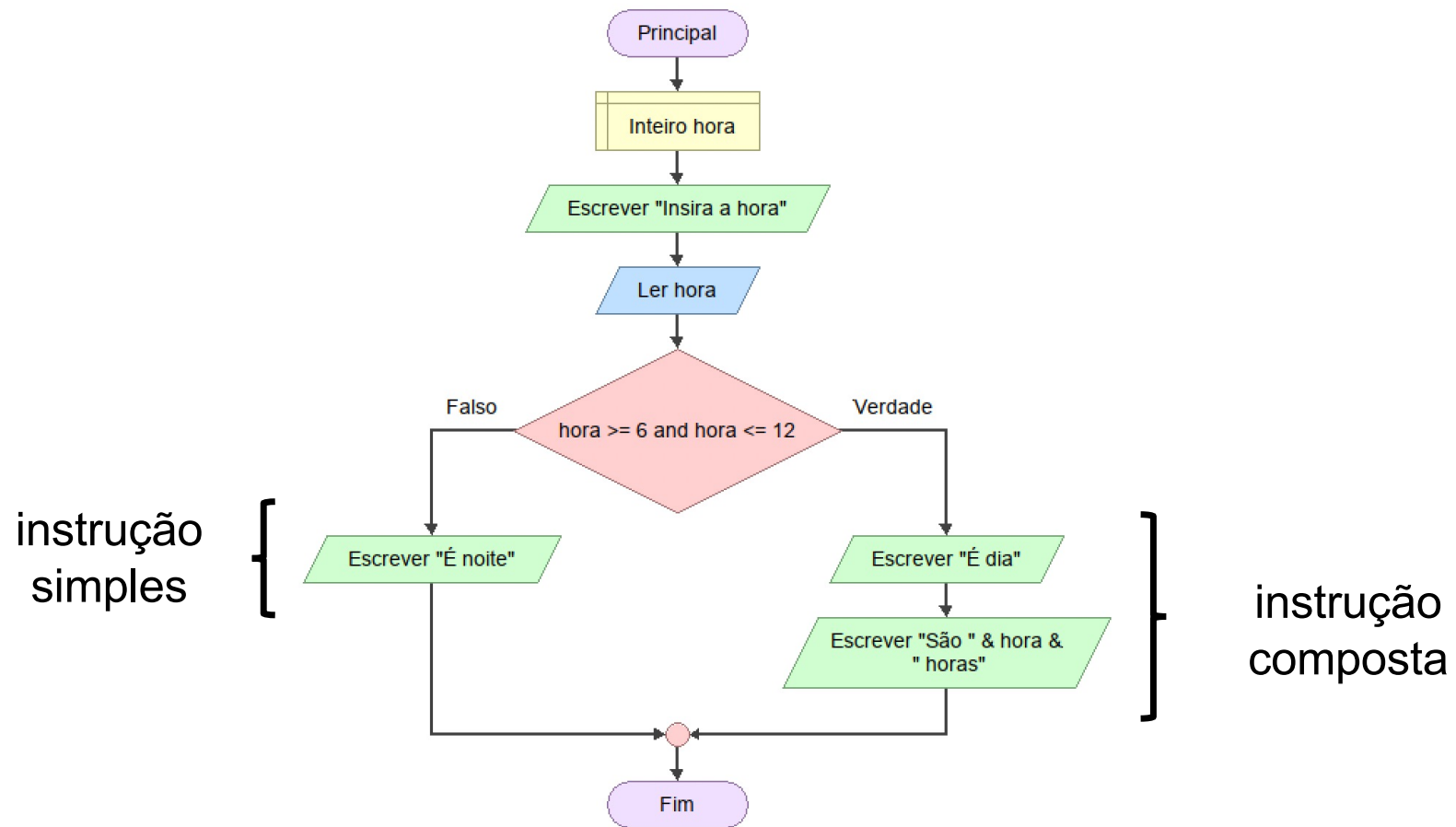
- **Instrução simples:** uma única instrução, normalmente ocupando uma só linha.

```
escreva("É dia")
```

- **Instrução composta:** um conjunto de instruções simples, agrupadas por um par de chavetas { ... }.

```
{  
    escreva("É dia")  
    escreva("São ", horas, " horas")  
}
```

Instruções simples e compostas



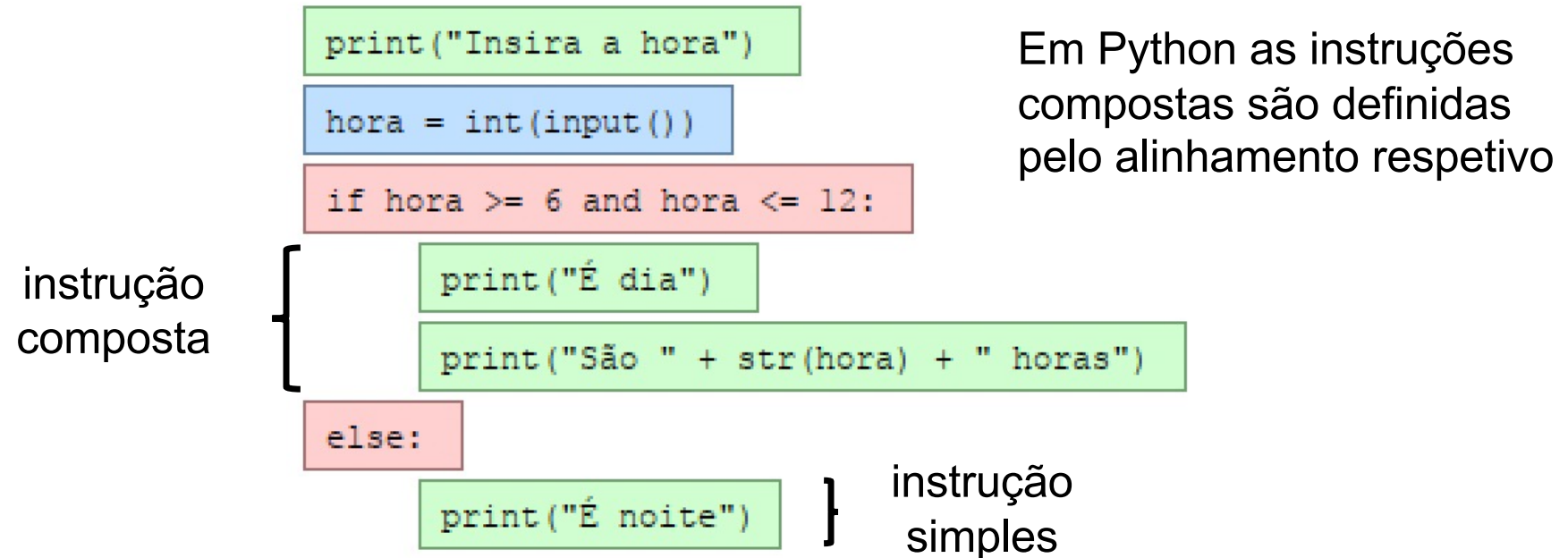
Instruções simples e compostas

```
programa
{
    funcao inicio()
    {
        inteiro hora
        escreva("Informe a hora: ")
        leia(hora)
        se(hora >= 6 e hora <= 18) {
            escreva("É dia\n")
            escreva("São ", hora, " horas")
        }
        senao
            escreva("É noite")
    }
}
```

instrução composta

instrução simples

Instruções simples e compostas



Seleção Composta – se ... senao se

```
se( <expressão_lógica_1> )  
    <instrução_1>  
senao  
    se( <expressão_lógica_2> )  
        <instrução_2>  
    senao  
        <instrução_3>
```

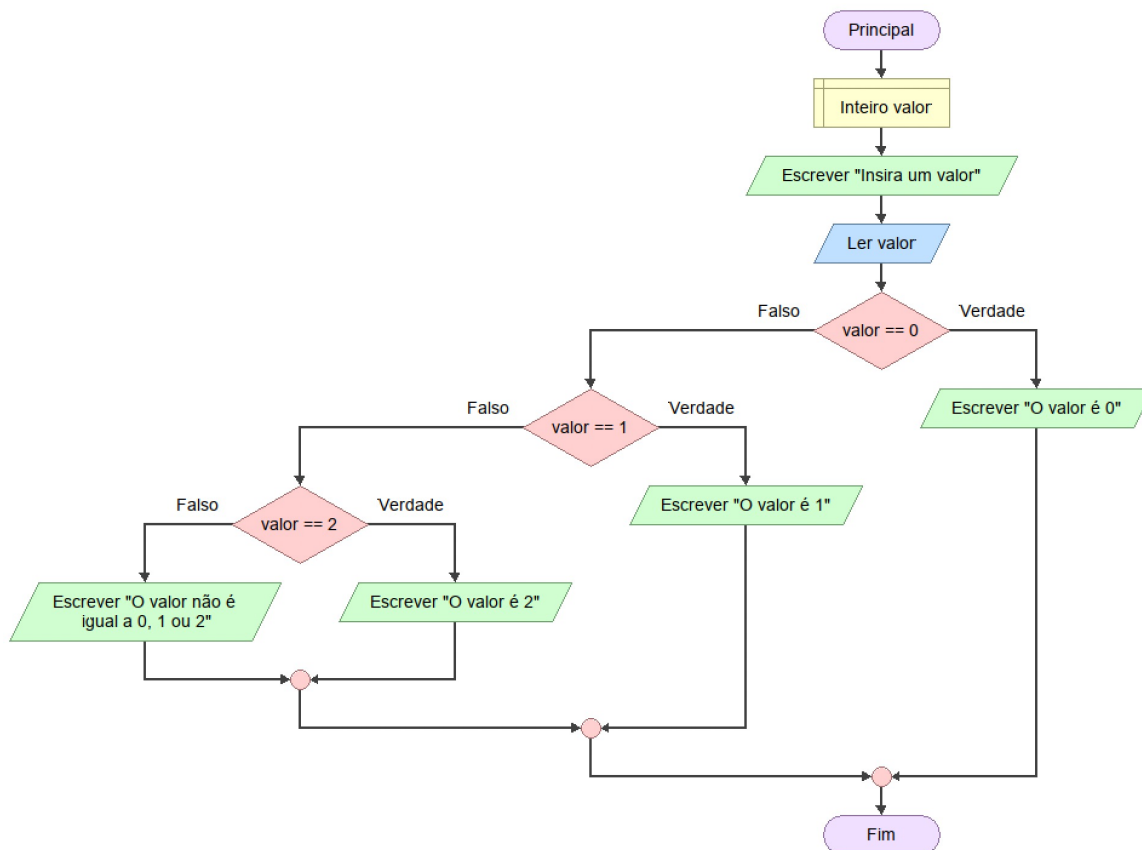
ou

```
se( <expressão_lógica_1> )  
    <instrução_1>  
senao se( <expressão_lógica_2> )  
    <instrução_2>  
senao  
    <instrução_3>
```

É possível encadear várias instruções se ... senao: um se, dentro de outro se ou dentro de um senao.

Não há limite para o número de instruções encadeadas.

Seleção de Múltipla Escolha (ou seleção de n vias)



Surge quando se pretende comparar o conteúdo de uma variável com uma **lista de valores** possíveis.

A variável `valor` tem como possibilidades os valores 0, 1 e 2.

Resulta em muitos testes de condição encadeados que podem tornar o fluxograma menos legível.

Seleção de Múltipla Escola - Portugal

```
escolha (<variável>)  
{  
    caso <valor_1>:  
        <instrução_1>  
    pare  
    ...  
    caso <valor_n>:  
        <instrução_n>  
    pare  
    [ caso contrario:  
        <instrução> ]  
}
```

**No Portugal, para evitar
muitas instruções “se”
encadeadas existe a
instrução alternativa
escolha ... caso**

Seleção de Múltipla Escolha - Python

```
print("Insira um valor")
valor = int(input())
if valor == 0:
    print("O valor é 0")
else:
    if valor == 1:
        print("O valor é 1")
    else:
        if valor == 2:
            print("O valor é 2")
        else:
            print("O valor não é igual a 0, 1 ou 2")
```

Num programa em Python surgem muitas instruções “if” encadeadas

Exemplo

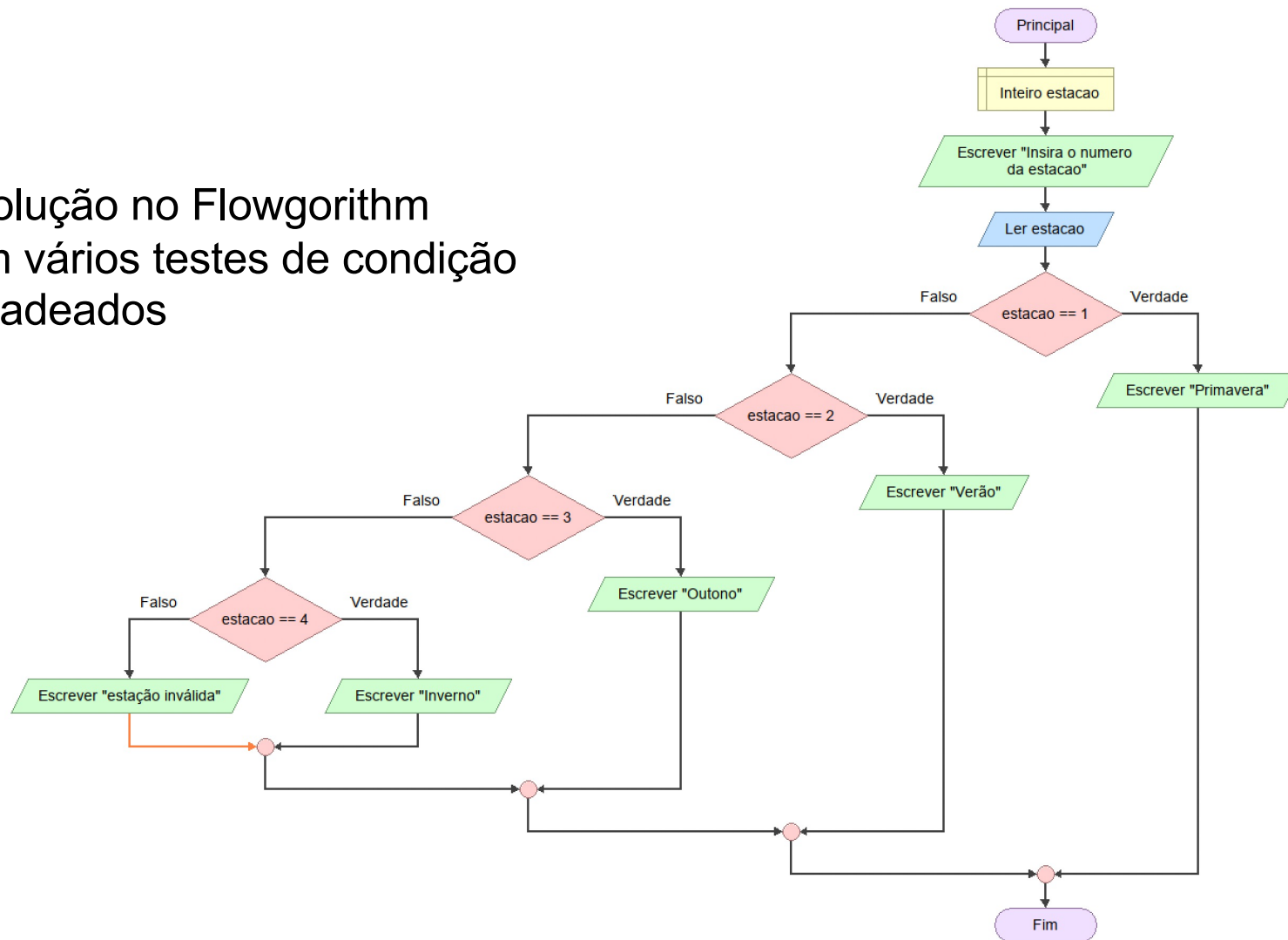
Ler um número entre 1 e 4 que representa a estação do ano e escrever o nome da estação do ano por extenso, sabendo que existe a correspondência seguinte:

- 1 – Primavera;
- 2 – Verão;
- 3 – Outono;
- 4 – Inverno.

Deverá ser apresentada uma mensagem de erro caso o número lido não esteja compreendido entre 1 e 4.

Exemplo

resolução no Flowgorithm
com vários testes de condição
encadeados



Exemplo

resolução em Python

```
print("Insira o numero da estacao")
estacao = int(input())
if estacao == 1:
    print("Primavera")
else:
    if estacao == 2:
        print("Verão")
    else:
        if estacao == 3:
            print("Outono")
        else:
            if estacao == 4:
                print("Inverno")
            else:
                print("estação inválida")
```

```
match estacao:
    case 1:
        print("primavera")
    case 2:
        print("verão")
    case 3:
        print("outono")
    case 4:
        print("inverno")
    case _:
        print("estação inválida")
```

Em Python, a instrução `match ... case` evita as instruções `if` encadeadas

Resumo

Fluxograma \longleftrightarrow Python

