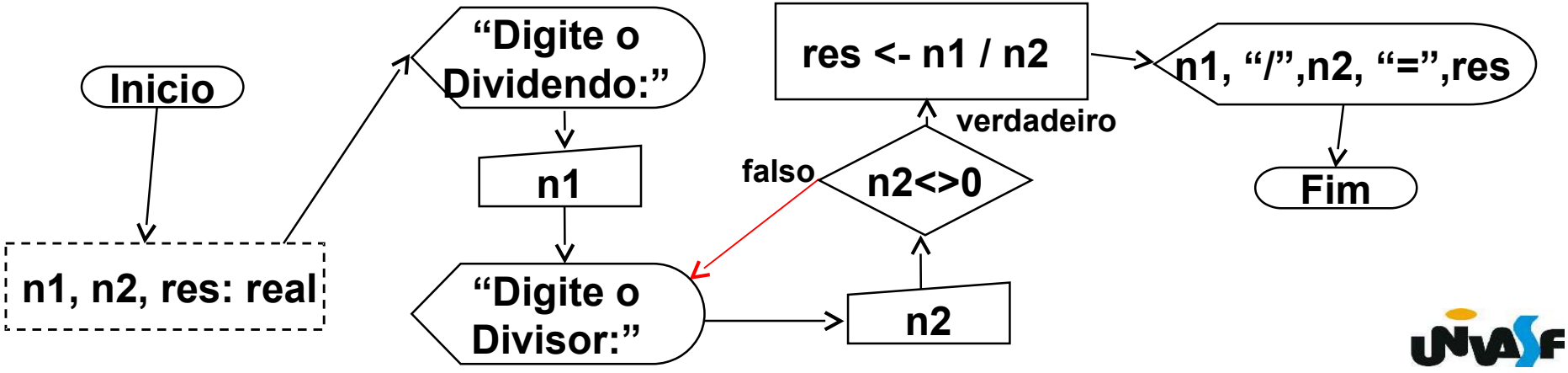
Estruturas de Controle de Fluxo

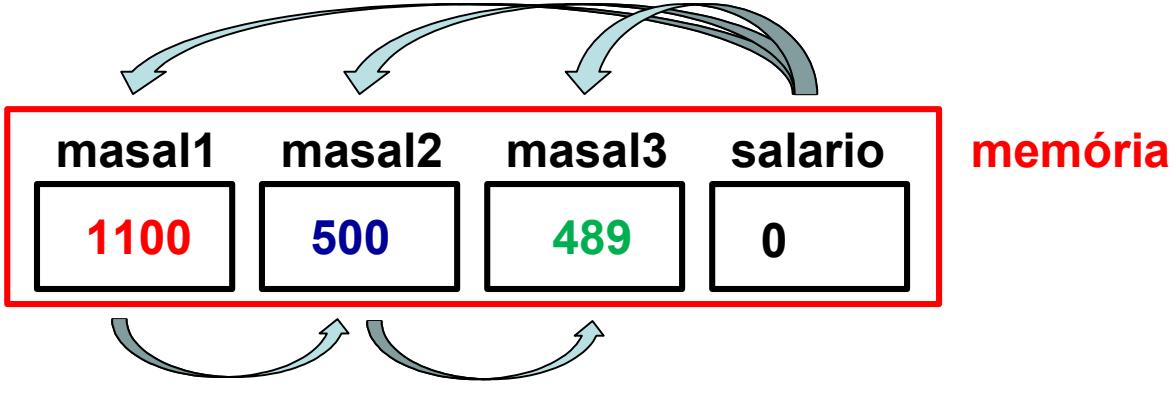
Fluxograma/Exercício 16 – Com base no que foi exposto, construa um fluxograma para obter o resultado da divisão entre dois números. **OBS.**: Caso um dos operandos não seja válido o mesmo deve ser novamente solicitado, até um valor válido ser fornecido, ou seja, as entradas devem ser validadas.



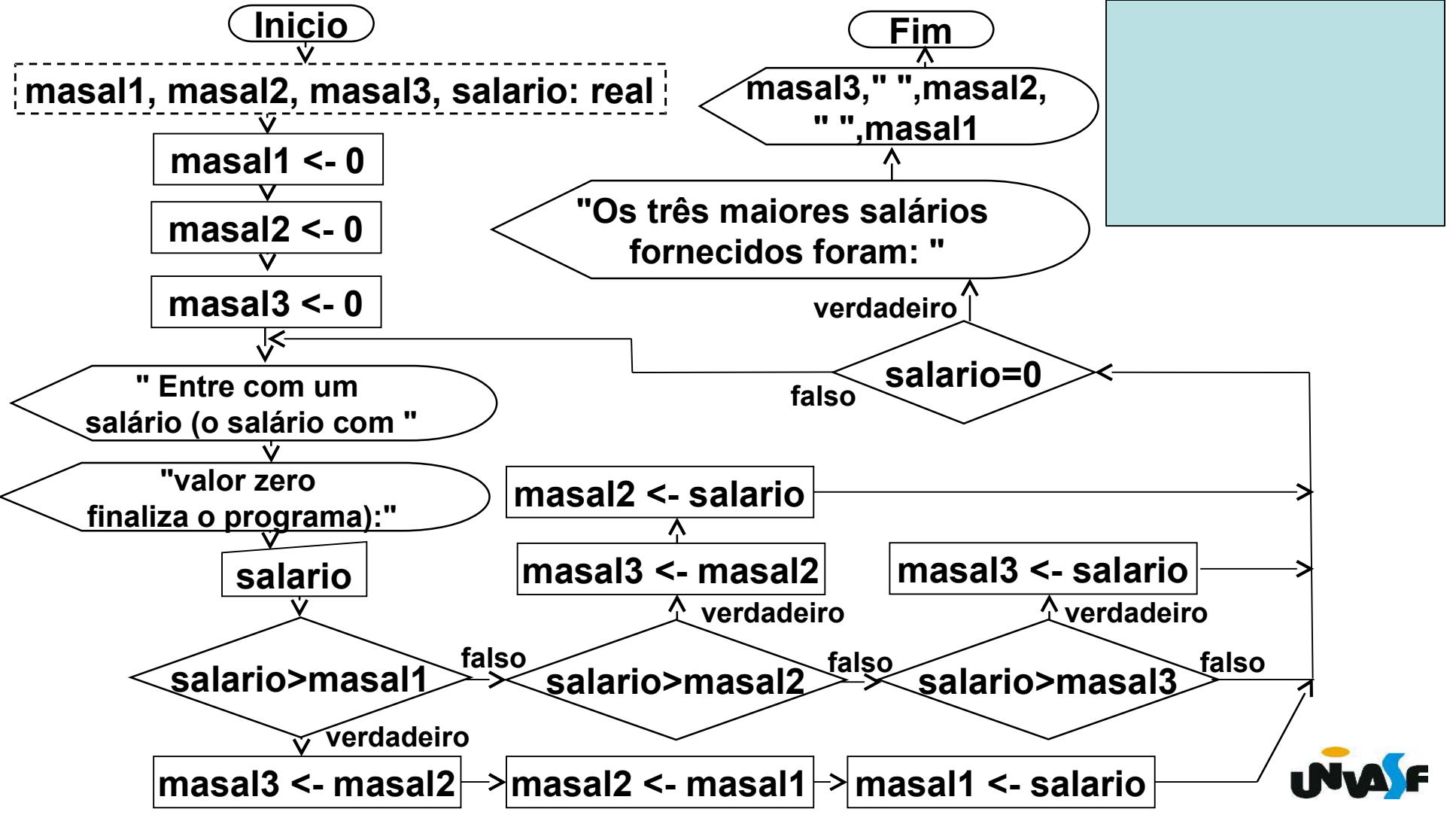
Estruturas de Controle de Fluxo

Exercício 17:

Elabore um algoritmo, representando-o através de um pseudocódigo e de um fluxograma, para ler uma sequência de salários, onde o indicador (***Flag***) de termino da sequência de salários é um salário igual a 0 (zero). O algoritmo deve escrever, em ordem crescente, os três maiores valores dos salários lidos.



```
algoritmo "exercício 17 laço de repetição"
var
   masal1, masal2, masal3, salario: real
inicio
   masal1 <- 0
   masal2 <- 0
   masal3 <- 0
   repita
      repita
         escreva ("Entre com um salário (o salário com ")
         escreva ("valor zero finaliza o programa): ")
         leia (salario)
      ate (salario>=0)
      se (salario>masal1) entao
         masal3 <- masal2
         masal2 <- masal1
         masal1 <- salario
      senao
         se (salario>masal2) entao
            masal3 <- masal2
            masal2 <- salario
         senao
            se (salario>masal3) entao
               masal3 <- salario
            fimse
         fimse
      fimse
   ate (salario = 0)
   escreva ("Os três maiores salários fornecidos foram:")
   escreva (masal3, " ", masal2, " ", masal1)
fimalgoritmo
```


Inicio
masal1, masal2, masal3, salario: real
masal1 <- 0
masal2 <- 0
masal3 <- 0
" Entre com um salário (o salário com "
"valor zero finaliza o programa):"
salario
salario>masal1
falso
verdadeiro
masal3 <- masal2
masal2 <- masal1
masal1 <- salario
salario>masal2
falso
verdadeiro
masal2 <- salario
masal3 <- masal2
salario>masal3
falso
verdadeiro
masal3 <- salario
salario=0
falso
verdadeiro
"Os três maiores salários fornecidos foram: "
masal3," ",masal2, " ",masal1
Fim

Estruturas de Controle de Fluxo Estruturas ou Laços de Repetição (enquanto e repita) - Exercícios

Parte 2

UNIVASF

Estruturas de Controle de Fluxo

Exercício 18:

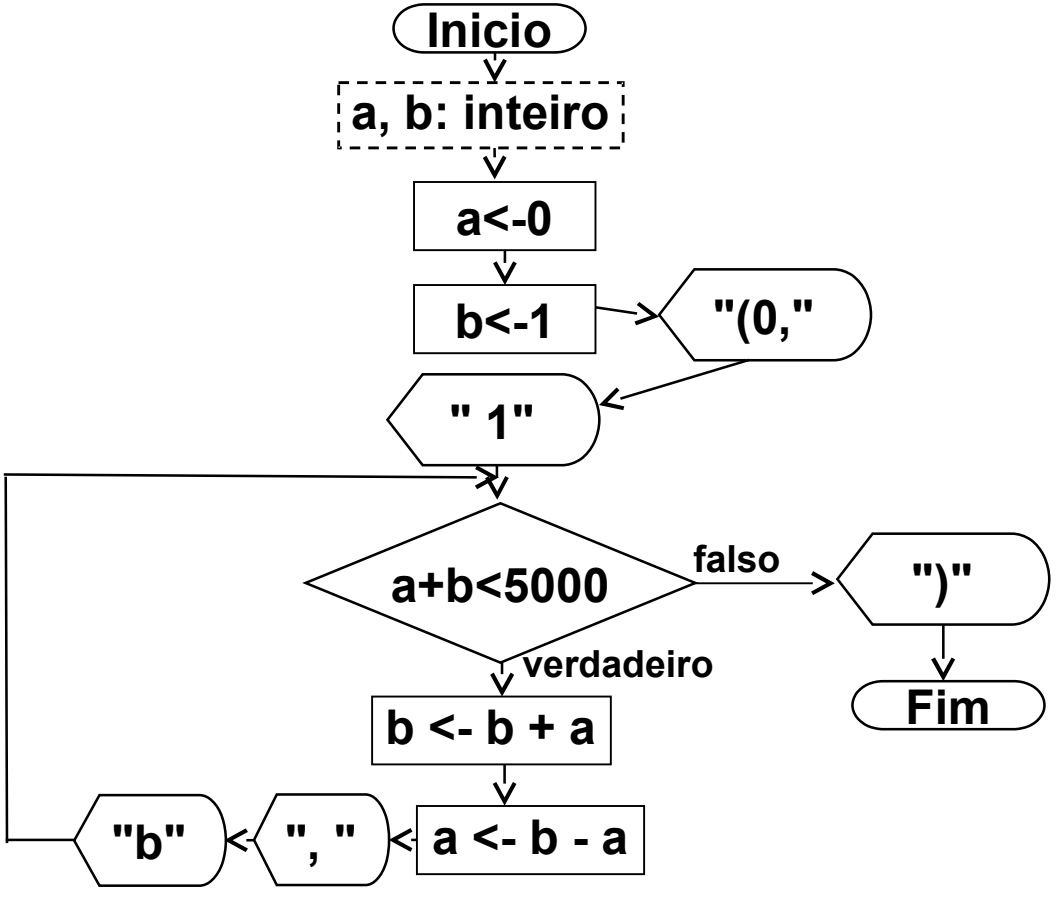
Faça um algoritmo, representando-o através de um pseudocódigo e de um fluxograma, para escrever a série de Fibonacci = (0, 1, 1, 2, 3, 5, 8, 13, 21, 34, …), enquanto o valor do termo a ser escrito for menor que 5000.

UNIVASF

```
algoritmo "exercício 18 laço de repetição a"
var
   a, b: inteiro
inicio
   a <- 0
   b <- 1
   escreva ("(",a)
   repita
      escreva (", ")
      escreva (b)
      b <- b + a
      a <- b - a
   ate (b>=5000)
   escreva (")")
fimalgoritmo
```

UNIVASF

```
algoritmo "exercício 18 laço de repetição b"
var
   a, b: inteiro
Inicio
   a <- 0
   b <- 1
   escreval("(0,")
   escreva (" 1")
   enquanto (a+b<5000) faca
      b <- b + a
      a <- b - a
      escreva (", ")
      escreva (b)
   fimenquanto
   escreva (")")
fimalgoritmo
```


Inicio
a, b: inteiro
a<-0
b<-1
"(0,"
" 1"
a+b<5000
falso
")"
Fim
verdadeiro
b <- b + a
a <- b - a
", "
"b"

Exercício 19:

Faça um algoritmo para com base no nome, sexo ("M" =Masculino ou "F"=Feminino), três notas e o número de faltas dos alunos de uma turma, onde o ***Flag*** será um nome igual a "fim", escrever:

a. A situação final de cada aluno;
b. A média das notas dos homens e a média das notas das mulheres;
c. O percentual de homens e o percentual de mulheres reprovados por média;
d. O percentual de homens e o percentual de mulheres reprovados por falta;
e. O percentual geral de reprovação da turma.

Obs.: As situações possíveis são: Aprovado, Reprovado por Falta ou Reprovado por Média. A média mínima para obter aprovação é 7,00 e o limite de faltas é 15. A reprovação por falta sobrepõe a reprovação por Média. **As entradas devem ser validadas.**

```
algoritmo " exercício 19 laço de repetição "
var nome, sexo: caractere
    n1, n2, n3, media, med_h_soma, med_m_soma: real
    faltas,num_h_rf,num_m_rf,num_h_rm,num_m_rm: inteiro
    total_h,total_m : inteiro
inicio
    num_h_rf <- 0
    num_m_rf <- 0
    num_h_rm <- 0
    num_m_rm <- 0
    total_h <- 0
    total_m <- 0
    med_h_soma <- 0.0
    med_m_soma <- 0
    repita
        escreva ("Entre com o nome do aluno(a) " )
        escreva ("(para finalizar a execução digite a palavra 'fim'): ")
        leia (nome)
        se (nome<>"fim") entao
            repita
                escreva ("Entre com o sexo (M =Masculino e F=Feminino): ")
                leia (sexo)
            ate ((sexo="m") ou (sexo="f"))
```

```
repita
   escreva ("Entre com o número de faltas do aluno(a): ")
   leia (faltas)
ate (faltas>=0)
repita
   escreva ("Entre com a primeira nota obtida: ")
   leia (n1)
ate ((n1>=0) e (n1<=10))
repita
   escreva ("Entre com a segunda nota obtida: ")
   leia (n2)
ate ((n2>=0) e (n2<=10))
repita
   escreva ("Entre com a terceira nota obtida: ")
   leia (n3)
ate ((n3>=0) e (n3<=10))
media <- (n1+n2+n3)/3
se (sexo="M") entao
   med_h_soma <- med_h_soma+media
   total_h <- total_h + 1
```

```
    se (faltas>15) entao
       escreval ("O aluno ",nome," foi reprovado por falta.")
       num_h_rf <- num_h_rf +1
    senao
       se (media<7) entao
          escreval ("O aluno "+nome+" foi reprovado por média.")
          num_h_rm <- num_h_rm +1
       senao
          escreval ("O(A) aluno "+nome+" foi aprovado.")
       fimse //do senao que indica que o aluno foi aprovado
    fimse //do senao que indica que o aluno não foi reprovado
          //por falta
senao
    med_m_soma <- med_m_soma+media
    total_m <- total_m + 1
```

```
        se (faltas>15) entao
           escreval ("A aluna ",nome," foi reprovada por falta.")
           num_m_rf <- num_m_rf +1
        senao
           se (media<7) entao
              escreval ("A aluna "+nome+" foi reprovada por média.")
              num_m_rm <- num_m_rm +1
           senao
              escreval ("A aluna "+nome+" foi aprovada.")
           fimse //do senao que indica que o aluno foi aprovado
        fimse //do senao que indica que o aluno não foi reprovado
        //por falta
     fimse //do senao que indica se o aluno é do sexo feminino
  fimse //do se que verifica de o nome é diferente de "fim"
ate (nome="fim")
se (total_h>0) entao
  escreva ("A média das notas dos homens é: ")
  escreval (med_h_soma/total_h:5:2)
```

```
    escreva ("O percentual de homens reprovados por média é: ")
    escreval (100*num_h_rm/total_h:6:2,"%")
    escreva ("O percentual de homens reprovados por falta é: ")
    escreval (100*num_h_rf/total_h:6:2,"%")
senao
    escreva ("Não foi possível calcular a média das notas e os ")
    escreva ("percentuais de homens reprovados por média e por ")
    escreval ("falta em função da não existência de alunos.")
fimse
se (total_m>0) entao
    escreva ("A média das notas das mulheres é: ")
    escreval (med_m_soma/total_m:5:2)
    escreva ("O percentual de mulheres reprovadas por média é: ")
    escreval (100*num_m_rm/total_m:6:2,"%")
    escreva ("O percentual de mulheres reprovadas por falta é: ")
    escreval (100*num_m_rf/total_m:6:2,"%")
senao
```

```
    escreva ("Não foi possível calcular a média das notas e os ")
    escreva ("percentuais de mulheres reprovadas por média e po")
    escreval ("r falta em função da não existência de alunas.")
  fimse
  se ((total_h>0) ou (total_m>0)) entao
    escreva ("O percentual geral de reprovação da turma é: ")
    escreval(100*(num_m_rf+num_m_rm+num_h_rf+num_h_rm)/(total_h+total_m):6:2,"%")
  senao
    escreva ("Não foi possível calcular o percentual geral de ")
    escreva ("reprovação da turma em função da não ")
    escreval ("existência de alunos.")
  fimse
fimalgoritmo
```

Estruturas de Controle de Fluxo
Estruturas ou Laços de Repetição (para)

UNIVASF

Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição

Analisando os laços de repetição estudados...
Perceberemos que normalmente ocorre:

- uma inicialização de uma variável;
- dentro do laço ocorre uma atualização no valor da variável mencionada.

```
<variável> <- <valor-inicial>
enquanto (<variável> <= <valor-limite>) faca
  <sequência-de-comandos>
  <variável> <- <variável> + <incremento>
fimenquanto
```

UNIVASF

Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição

Sintaxe:

...

para <*variável*> de <*valor-inicial*> ate <*valor-limite*>
[passo <*incremento*>] faca

    <*sequência-de-comandos*>

fimpara

...

```
para <variável> de <valor-inicial> ate <valor-limite>
[passo <incremento>] faca
   <sequência-de-comandos>
fimpara
```

```
<variável> <- <valor-inicial>
enquanto (<variável> <= <valor-limite>) faca
   <sequência-de-comandos>
   <variável> <- <variável> + <incremento>
fimenquanto
```

Equivalência quando incremento é positivo

```
<variável> <- <valor-inicial>
enquanto (<variável> >= <valor-limite>) faca
   <sequência-de-comandos>
   <variável> <- <variável> + < -incremento>
fimenquanto
```

Equivalência quando o incremento é negativo

UNIVASF

Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição (continuação)

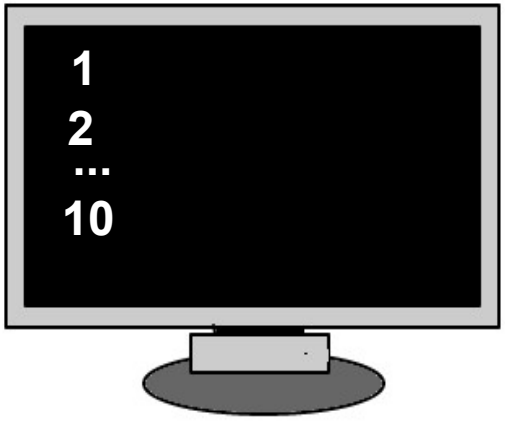
Exemplo 7:

O pseudocódigo e o fluxograma a seguir representam um algoritmo que escreve na saída padrão os números inteiros contidos no intervalo [1, 10].

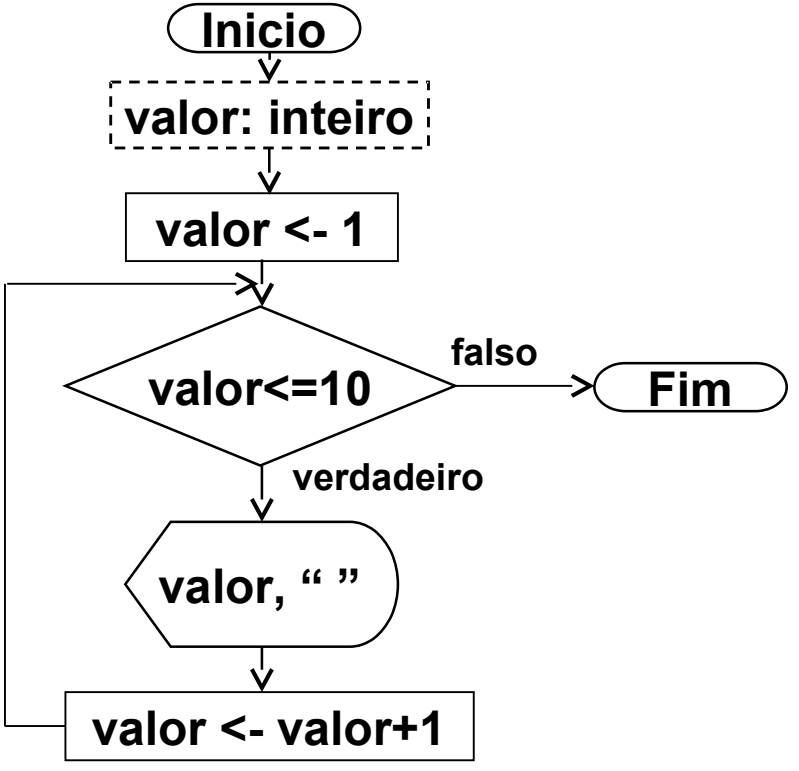
UNIVASF

Estruturas de Controle de Fluxo

```
algoritmo "exemplo 7"
var
    valor: inteiro
inicio
    para valor de 1 ate 10 faca
     escreval (valor)
    fimpara
fimalgoritmo
```



Estruturas de Controle de Fluxo



Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição (continuação)

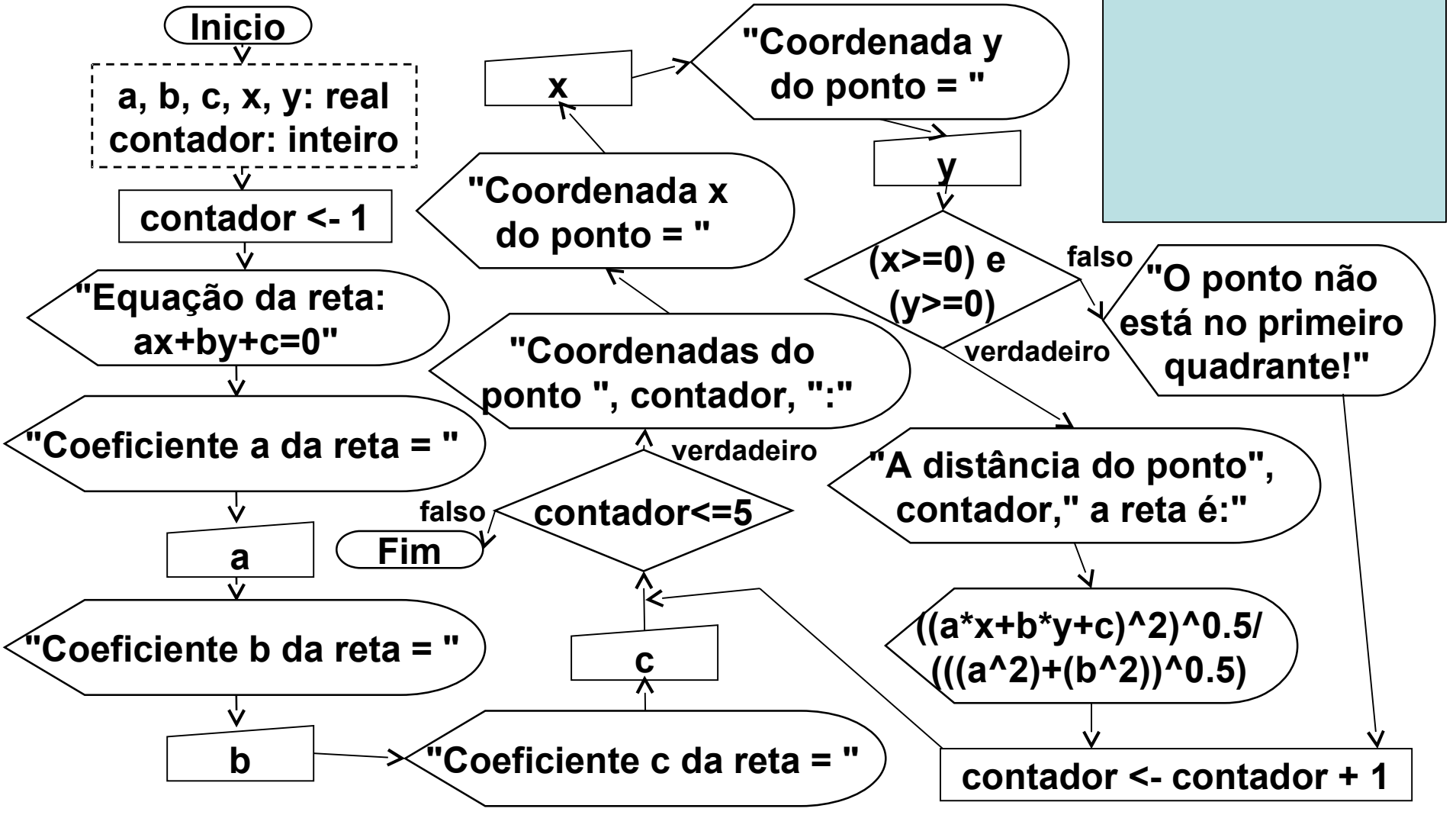
Exemplo 8:

Dada uma reta ax+by+c=0 e cinco pontos, faça um algoritmo para calcular, para cada ponto, o seguinte: se o ponto estiver no primeiro quadrante calcule e informe a distância do ponto a reta, caso contrário, escreva uma mensagem informando que o ponto não pertence ao primeiro quadrante.

UNIVASF

```
var   a,b,c,x,y: real
      contador: inteiro
inicio
   escreval ("Equação da reta: ax+by+c=0")
   escreva ("Coeficiente a da reta = ")
   leia (a)
   escreva ("Coeficiente b da reta = ")
   leia (b)
   escreva ("Coeficiente c da reta = ")
   leia (c)
   para contador de 1 ate 5 passo 1 faca
      escreval ("Coordenadas do ponto ",contador," :")
      escreva ("Coordenada x do ponto = ")
      leia (x)
      escreva ("Coordenada y do ponto = ")
      leia (y)
      se ((x>=0) e (y>=0)) entao
         escreva ("A distância do ponto ",contador," a reta é: ")
         escreval (((a*x+b*y+c)^2)^0.5/(((a^2)+(b^2))^0.5))
      senao

      fimse
   fimpara
fimalgoritmo
```


Inicio
a, b, c, x, y: real
contador: inteiro
contador <- 1
"Equação da reta: ax+by+c=0"
"Coeficiente a da reta = "
a
"Coeficiente b da reta = "
b
"Coeficiente c da reta = "
c
"Coordenadas do ponto ", contador, ":"
"Coordenada x do ponto = "
x
"Coordenada y do ponto = "
y
(x>=0) e (y>=0)
falso
"O ponto não está no primeiro quadrante!"
verdadeiro
"A distância do ponto", contador," a reta é:"
((a*x+b*y+c)^2)^0.5/ (((a^2)+(b^2))^0.5)
contador <- contador + 1
contador<=5
verdadeiro
falso
Fim

Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição (continuação)

Exercício 20:

Construa um algoritmo, representando-o através de um pseudocódigo e de um fluxograma, que exiba na saída padrão uma contagem decrescente do valor 30 até o valor 1.

UNIVASF

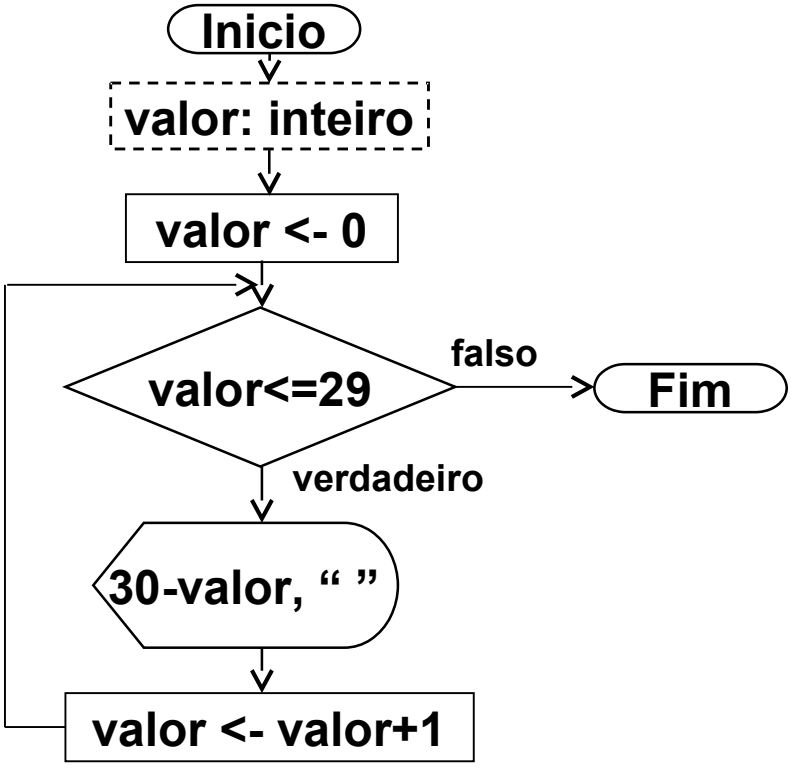
Estruturas de Controle de Fluxo

```
algoritmo "exercício 20"
var
    valor: inteiro
inicio
    para valor de 30 ate 1 passo -1 faca
     escreval (valor)
    fimpara
fimalgoritmo
```

Estruturas de Controle de Fluxo

```
algoritmo "exercício 20 S.A."
var
    valor: inteiro
inicio
    para valor de 0 ate 29 faca
     escreval (30-valor)
    fimpara
fimalgoritmo
```

Estruturas de Controle de Fluxo



Estruturas de Controle de Fluxo

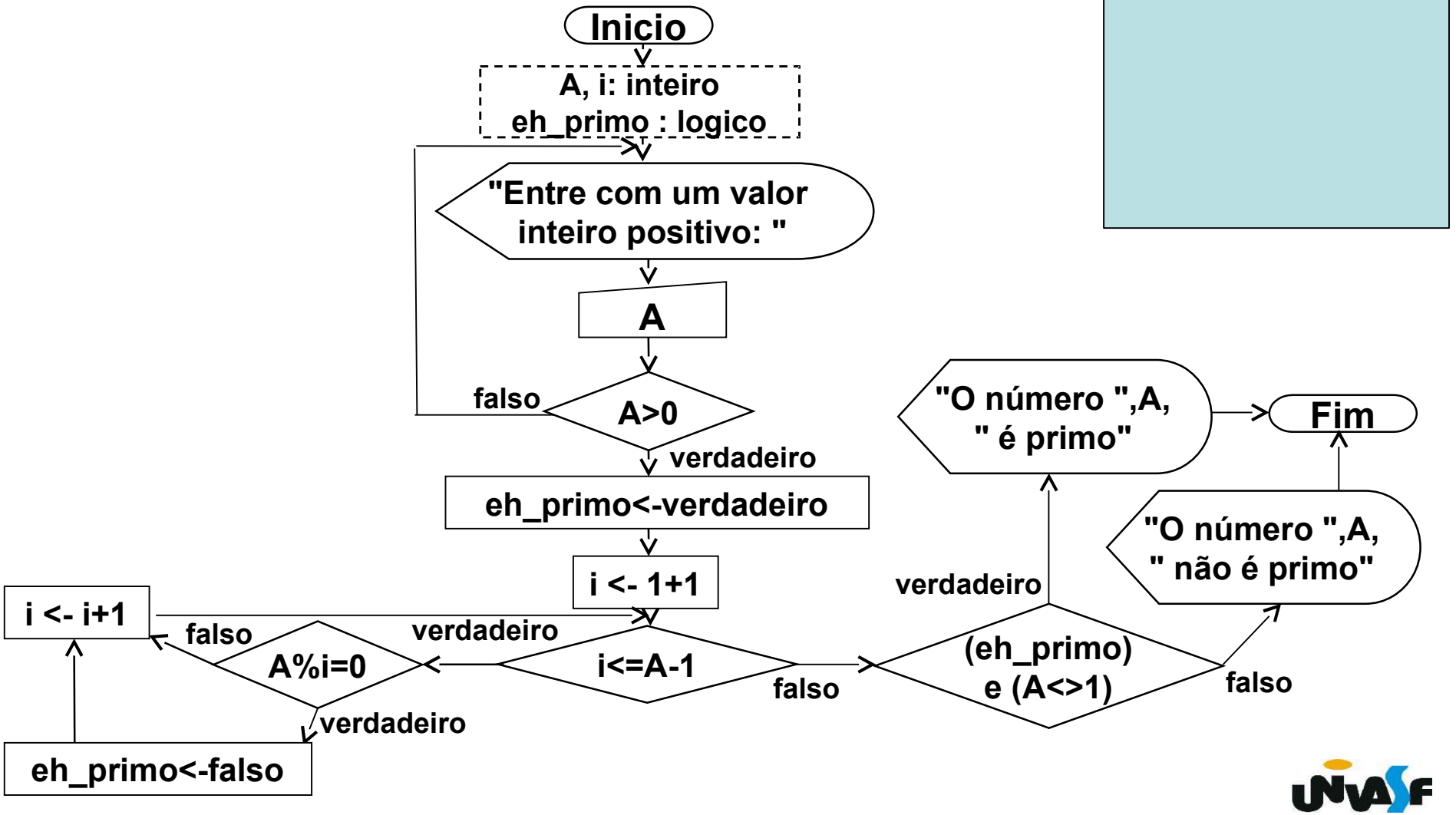
6. Estrutura ou laço de repetição (continuação)

Exercício 21:

Construa um algoritmo, representando-o através de um pseudocódigo e de um fluxograma, que leia da entrada padrão um número inteiro positivo e retorne na saída padrão uma mensagem indicando se o número lido é ou não um número primo. As entradas devem ser validadas.

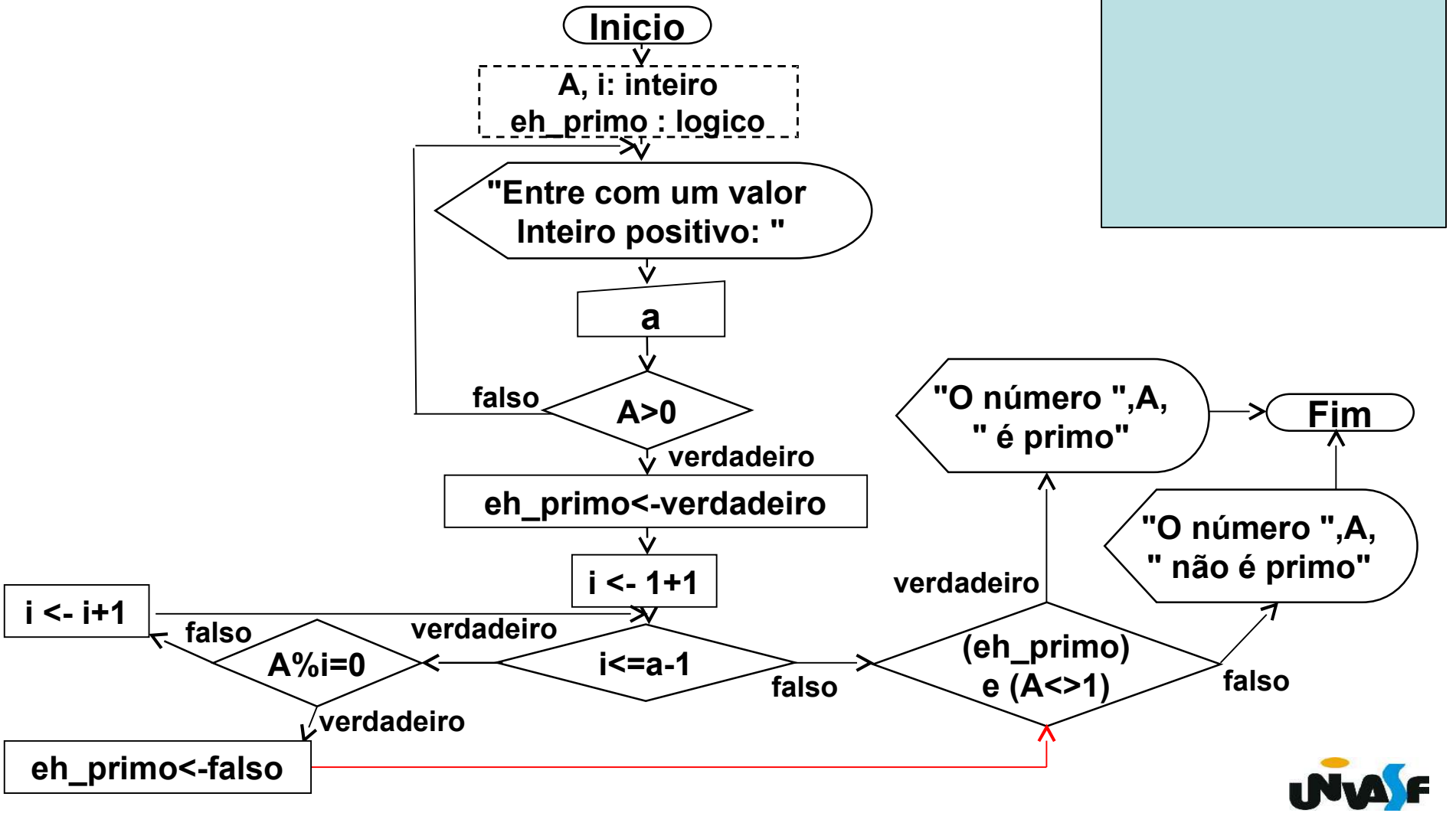
UNIVASF

```
algoritmo "exercício 21"
var    A, i: inteiro
       eh_primo: logico
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   para i de 1+1 ate A-1 faca
      se (A%i=0) entao
         eh_primo <- falso
      fimse
   fimpara
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```


Inicio
A, i: inteiro
eh_primo : logico
"Entre com um valor inteiro positivo: "
A
A>0
falso
verdadeiro
eh_primo<-verdadeiro
i <- 1+1
i<=A-1
verdadeiro
falso
A%i=0
falso
verdadeiro
i <- i+1
eh_primo<-falso
(eh_primo) e (A<>1)
verdadeiro
falso
"O número ",A, " é primo"
"O número ",A, " não é primo"
Fim

```
algoritmo "exercício 21 usando interrompa"
var    A, i: inteiro
       eh_primo: logico
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   para i de 1+1 ate A-1 faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa //causa uma saída imediata do laço
      fimse
   fimpara
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

UNIVASF


Inicio
A, i: inteiro
eh_primo : logico
"Entre com um valor
Inteiro positivo: "
a
falso
A>0
verdadeiro
eh_primo<-verdadeiro
i <- 1+1
i <- i+1
falso
A%i=0
verdadeiro
verdadeiro
i<=a-1
falso
verdadeiro
(eh_primo)
e (A<>1)
falso
eh_primo<-falso
"O número ",A,
" é primo"
Fim
"O número ",A,
" não é primo"

Estruturas de Controle de Fluxo

Analisando melhor o problema de verificar se um determinado número *n* é ou não um número primo observamos o seguinte:

supondo que o *n* seja 9:

9/1 = 9
9/2 = 4.50
9/3 = 3 — divisor<=quociente
9/4 = 2.25 — divisor>quociente
9/5 = 1.80
9/6 = 1.50
9/7 = 1.28
9/8 = 1.12
9/9 = 1
9^0.5=3

Não Primo

Testar apenas até $\lfloor \sqrt{n} \rfloor$

supondo que o *n* seja 11:

11/1 = 11
11/2 = 5.50
11/3 = 3.67 — divisor<=quociente
11/4 = 2.75 — divisor>quociente
11/5 = 2.20
11/6 = 1.83
11/7 = 1.57
11/8 = 1.37
11/9 = 1.22
11/10 = 1.1
11/11 = 1
11^0.5=3.32

Primo

Estruturas de Controle de Fluxo

6. Estrutura ou laço de repetição (continuação)

Exercício 22:

Com base no que foi discutido no slide anterior construa um algoritmo que leia da entrada padrão um número inteiro positivo e retorne na saída padrão uma mensagem indicando se o número lido é ou não um número primo. As entradas devem ser validadas.

UNIVASF

```
algoritmo "exercício 22 - resposta"
var    A, i: inteiro
       eh_primo: logico
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   para i de 1+1 ate A^0.5 faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa
      fimse
   fimpara
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

```
algoritmo "exercício 22 - resposta 1"
var    A, i, aux: inteiro
       eh_primo: logico
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   aux<-1
   enquanto (aux*aux<=a)faca
      aux<-aux+1
   fimenquanto
   para i de 1+1 ate aux-1 faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa
      fimse
   fimpara
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

```
algoritmo "exercício 22 - resposta 2"
var    A, i, aux1: inteiro
       eh_primo: logico
       aux2:real
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   aux1<-1
   aux2<-a^0.5
   enquanto (aux1<=aux2) faca
      aux1<-aux1+1
   fimenquanto
   para i de 1+1 ate aux1-1 faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa
      fimse
   fimpara
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

```
algoritmo "exercício 22 - resposta 3"
var    A, i: inteiro
       eh_primo: logico
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   i <- 1+1
   enquanto (i<=A^0.5) faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa
      fimse
      i<-i+1
   fimenquanto
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

```
algoritmo "exercício 22 - resposta 4"
var    A, i: inteiro
       eh_primo: logico
       aux: real
inicio
   repita
      escreva ("Entre com um valor inteiro positivo: ")
      leia (A)
   ate (A>0)
   eh_primo <- verdadeiro
   i <- 1+1
   aux <- A^0.5
   enquanto (i<=aux) faca
      se (A%i=0) entao
         eh_primo <- falso
         interrompa
      fimse
      i<-i+1
   fimenquanto
   se ((eh_primo) e (A<>1)) entao
     escreva ("O número ",A," é primo")
   senao
     escreva ("O número ",A," não é primo")
   fimse
fimalgoritmo
```

Estruturas de Controle de Fluxo

Teorema de Böhm-Jacopini e Exercícios

UNIVASF

Estruturas de Controle de Fluxo

O teorema da programação estruturada, conhecido como **Teorema de Böhm-Jacopini**.

Enunciado em 1966 por Corrado Böhm e Giuseppe Jacopini sendo resultado da teoria das linguagens de programação.

O qual define que cada rotina computável pode ser descrita por um algoritmo que combine as instruções utilizando apenas três maneiras especificas:

1. Executar uma instrução, depois outra instrução (sequência);
2. Executar uma dentre duas sequências de instruções de acordo com um valor booleano (condição);
3. Executar uma sequência de instruções até que um valor booleano seja verdadeiro (iteração).

UNIVASF

Estruturas de Controle de Fluxo

6. Laços de repetição (continuação)

Exercício 23:

Com base nos conceitos estudados solucione o problema de receber um número natural e retornar o seu fatorial. Gerar três soluções, utilizando em cada uma, uma das estruturas de repetição vistas. As entradas devem ser validadas. **Obs.:** Os algoritmos gerados devem ser representados através de pseudocódigos.

UNIVASF

```
algoritmo "exercício 23 - repita"
var num, fat: inteiro
inicio
    repita
        escreva ("Digite um número natural: ")
        leia (num)
    ate (num>=0)
    se ((num=0) ou (num=1)) entao
        fat <- 1
    senao
        fat <- num
        repita
            num <- num - 1
            fat <- fat * num
        ate (num=1)
    fimse
    escreva ("O fatorial é ", fat)
fimalgoritmo
```

```
algoritmo " exercício 23 - enquanto"
var num, fat: inteiro
inicio
    num <- -1
    enquanto (num < 0) faca
        escreva ("Digite um número natural: ")
        leia (num)
    fimenquanto
    se (num=0) entao
        fat <- 1
    senao
        fat <- num
        enquanto (num>2) faca
            num <- num - 1
            fat <- fat * num
        fimenquanto
    fimse
    escreva ("O fatorial é ", fat)
fimalgoritmo
```

```
algoritmo "exercício 23 - para"
var
   i,num,fat: inteiro
inicio
   para i de 1 ate 1 faca
      escreva ("Digite um número natural: ")
      leia (num)
      se (num<0) entao
         i<-i-1
         escreval ("Não foi fornecido um valor válido!")
      fimse
   fimpara
   fat <- 1
   para i de 2 ate num faca
      fat <- fat * i
   fimpara
   escreva ("O fatorial de ", num, " é: ", fat)
fimalgoritmo
```

```
algoritmo "exercício 23 - para"
var
   i,num,fat: inteiro
inicio
   repita
      escreva ("Digite um número natural: ")
      leia (num)
   ate (num>=0)
   fat <- 1
   para i de 2 ate num faca
      fat <- fat * i
   fimpara
   escreva ("O fatorial de ", num, " é: ", fat)
fimalgoritmo
```

```
algoritmo "exercício 23 - para - resposta alternativa"
var
   num,aux: inteiro
inicio
   repita
      escreva ("Digite um número natural: ")
      leia (num)
   ate (num>=0)
   se (num=0) entao
      escreva ("O fatorial é: 1")
   senao
      para aux de num-1 ate 2 passo -1 faca
         num <- num * aux
      fimpara
      escreva ("O fatorial é: ", num)
   fimse
fimalgoritmo
```

Estruturas de Controle de Fluxo

6. Laços de repetição (continuação)

Exercício 24:

Escreva um algoritmo para calcular o valor da série, para 5 termos.

$$S = -\frac{1}{2!} + \frac{2}{4!} - \frac{3}{6!} + \ldots$$

Estruturas de Controle de Fluxo

6. Laços de repetição (continuação)

Exercício 25:

Escreva um algoritmo para calcular o valor da série, para N termos. Onde o valor de N será fornecido pelo usuário.

$$S = -\frac{1}{2!} + \frac{2}{4!} - \frac{3}{6!} + \ldots$$