IEC037
Introdução à Programação de Computadores
Aula 06 – Estruturas Condicionais

Turmas: Física

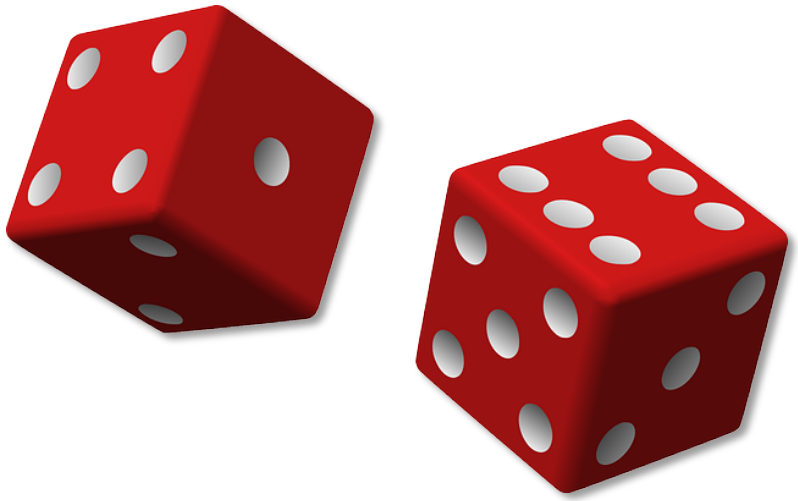
Professora: André Luiz da Costa Carvalho

E-mail: andre@icomp.ufam.edu.br

Site: http://iccufam.wordpress.com

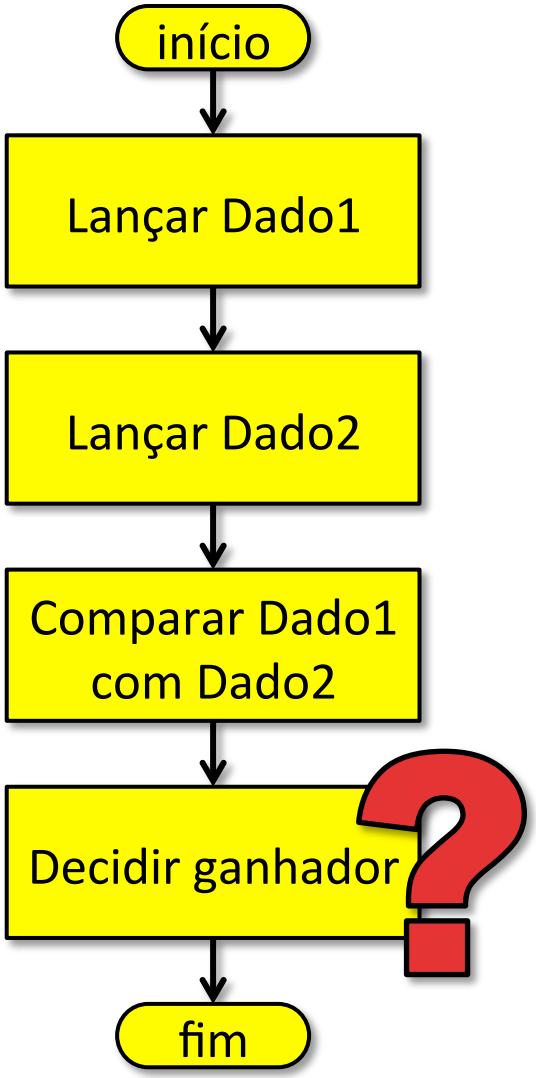
Problema Inicial

- Dois jogadores lançam dados. Ganha aquele que tirar a face com maior número.
- Como determinar quem ganhou? Ou se houve empate?



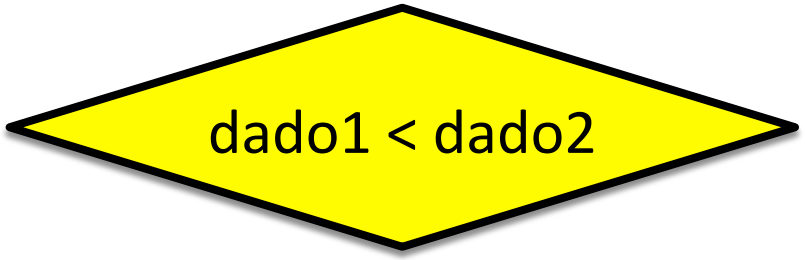
Tentativa de Solução :: Estrutura Sequencial

- Este problema requer uma alteração no fluxo de execução do programa.
- É necessário incluir alguma forma de ramificação, com um teste de condição para decidir qual ramo seguir durante a execução do algoritmo.



Estruturas Condicionais

- Permitem alterar o fluxo de execução, de modo a selecionar qual parte do algoritmo deve ser executada.
- Essa decisão é tomada a partir de uma condição, que pode resultar apenas em:
  - Verdade, ou
  - Falsidade



condição

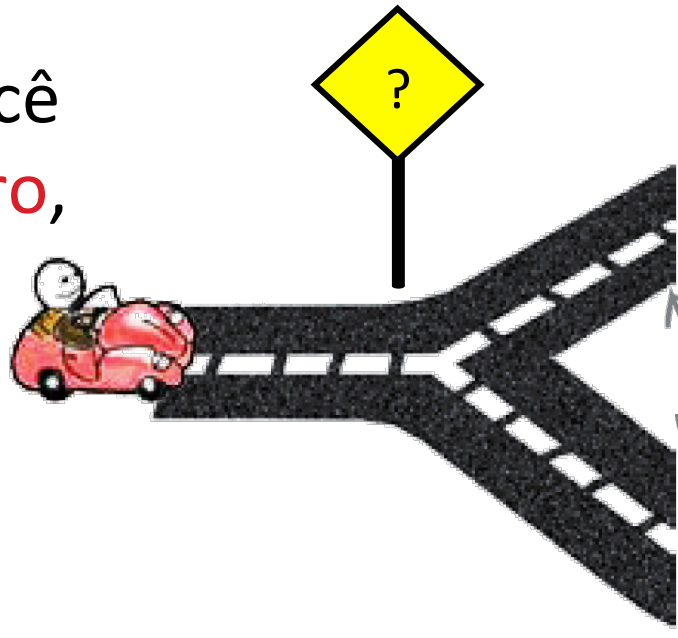
Estruturas Condicionais Simples

- Condição é verdadeira:
  - "bloco verdade" é executado.
- Condição é falsa:
  - "bloco verdade" **não** é executado.
  - Na condição simples, não há ação alternativa para a condição falsa.

```
If delta < 0:
    print "Não tem raiz real"
```

Estrutura Condicional Simples :: Analogia

- Pense a execução de um programa como a direção de um carro.
- Enquanto há retas, você só tem de seguir em frente. Somente nas bifurcações você terá de tomar decisões.
- Uma vez decidido o caminho, você somente percorre ou um ou outro, mas nunca ambos.



Estrutura Condicional Composta

- Condição é verdadeira:
  - "bloco verdade" é executado.
- Condição é falsa:
  - "bloco alternativo" é executado.
- Na condição composta, há uma ação alternativa caso a condição seja avaliada como falsa.

```
If média >= 0:
    print "Aprovado"
else:
    print "reprovado"
```

Uma condição possui apenas dois resultados possíveis

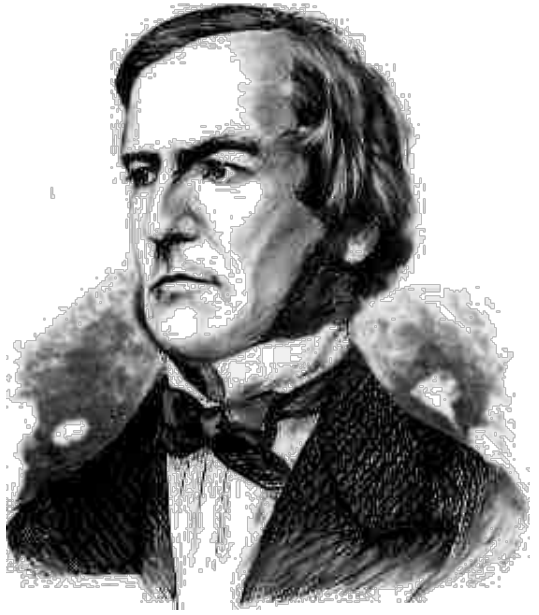
- O resultado de uma condição só pode ser:

Verdadeiro

Falso

Condições são expressões booleanas

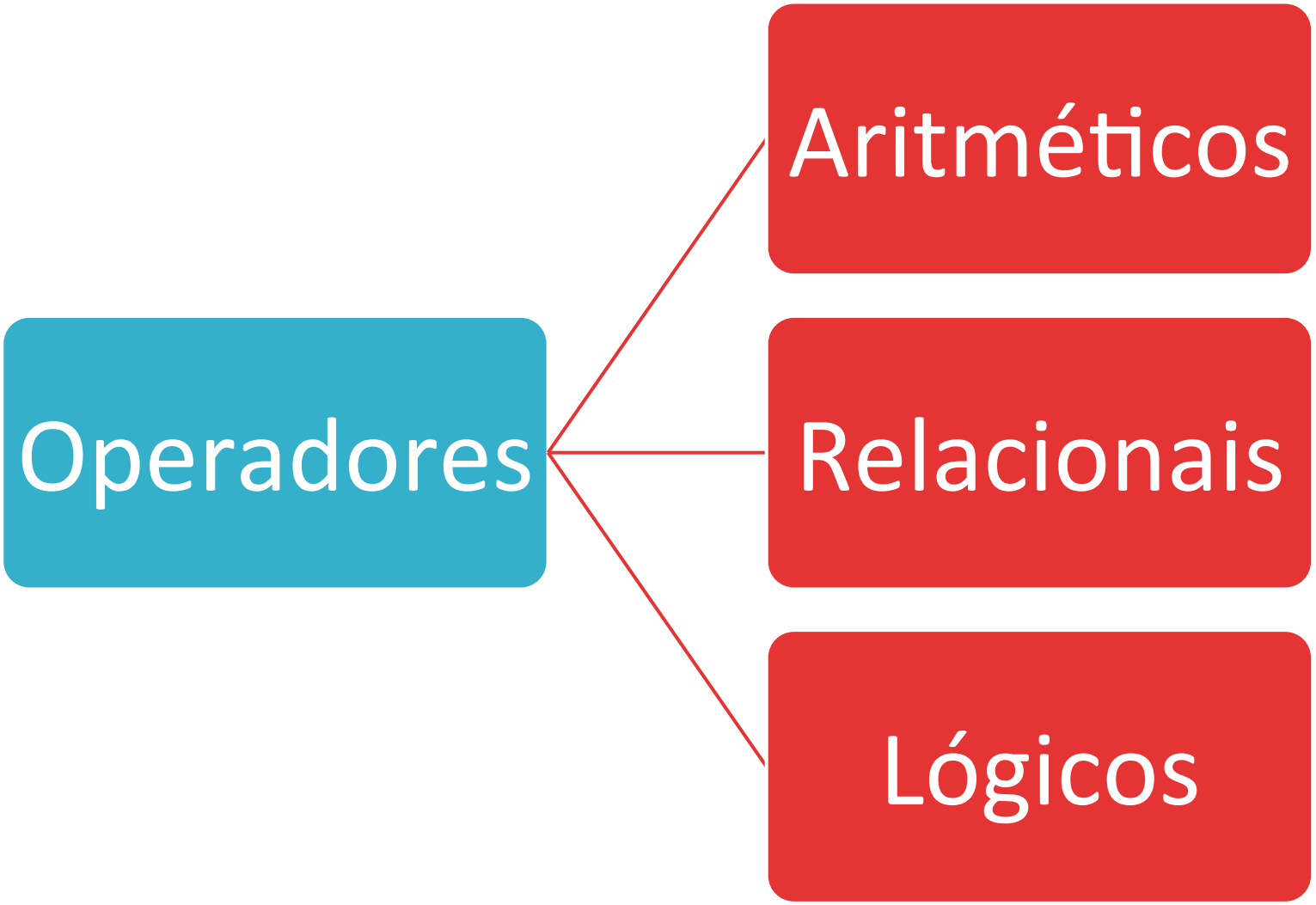
- Expressões que resultam em apenas dois valores (verdadeiro/falso, sim/não, zero/um) são conhecidas como expressões booleanas.
- Este nome vem do matemático inglês George Boole (1815–1864), que lançou os fundamentos da lógica matemática.



Como montar uma condição?

- Uma condição resulta da combinação dos seguintes elementos:
  - Operandos (valores aritméticos)
  - Operadores (sinais que representam operações)

Tipos de operadores



Operadores Relacionais

- São utilizados para a estabelecer relação de comparação entre valores numéricos.

Operador	Operação	Exemplos
==	Igual a	3 == 3 20 == 18
>	Maior que	5 > 4 10 > 11
<	Menor que	3 < 6 9 < 7
≥	Maior ou igual a	5 ≥ 3 4 ≥ 4
≤	Menor ou igual a	3 ≤ 5 7 ≤ 7
≠	Diferente de	8 ≠ 9 2 ≠ 2

Operadores relacionais × aritméticos

:: Formato

Operadores Aritméticos

Operadores Relacionais

Operadores relacionais × aritméticos

:: Prioridade

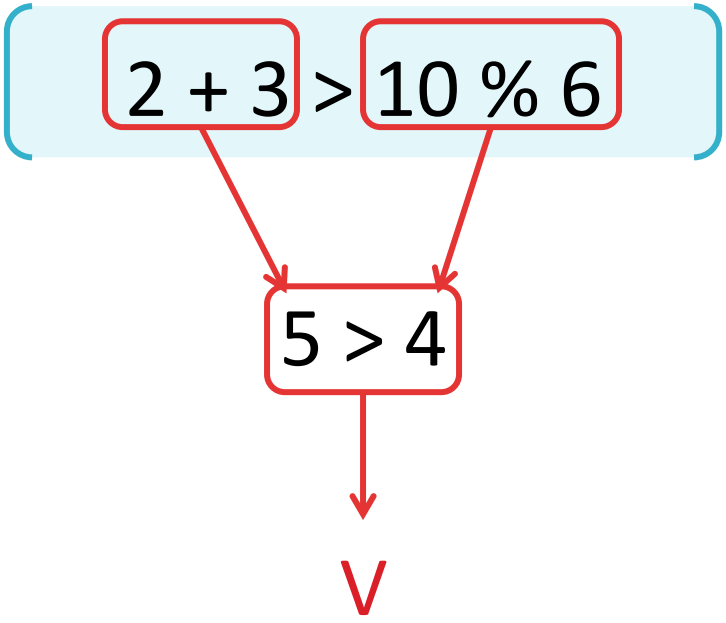
Prioridade	Operador
1	Parênteses mais internos
2	Operadores aritméticos
3	Operadores relacionais

Da esquerda para a direita

Na dúvida, use parênteses.

Como avaliar uma condição?

- Operadores relacionais têm menor prioridade que os operadores aritméticos.



Como avaliar uma condição?
:: Exercício

Considere: $x \leftarrow 2$, $y \leftarrow 3$, $z \leftarrow 7$

Expressão	Verdadeiro	Falso	Mal formada
x + y > 6			
x - 1 + y == 4			
x ** y == x * y			
y - 5 = z - 9			
1 - z ≠ 4 < 11			
x + 8 % z ≥ y * 6 – 15			

Como avaliar uma condição?
:: Exercício

Considere: x ← 2, y ← 3, z ← 7

Expressão	Verdadeiro	Falso	Mal formada
x + y > 6		X	
x - 1 + y == 4	X		
x ** y == x * y		X	
y - 5 = z - 9			X
1 - z ≠ 4 < 11			X
x + 8 % z ≥ y * 6 – 15	X		

Qual a diferença entre os símbolos "=" e "==" ?

- O símbolo "=" indica uma atribuição de valor.
  - O valor da variável à esquerda do símbolo é modificado pelo valor à direita.

```
x = x + 1
```

- O símbolo "==" indica uma comparação de valores.
  - Nenhum valor é modificado. Eles são apenas comparados, produzindo um resultado lógico (**V** ou **F**).

```
x == 2
```

Problema 1

- Uma lata de leite em pó da marca A, com 400g, custa R$ 8,39.
- Um saco de leite em pó da marca B, com 1kg, custa R$ 20,30.
- Qual marca tem o melhor preço?

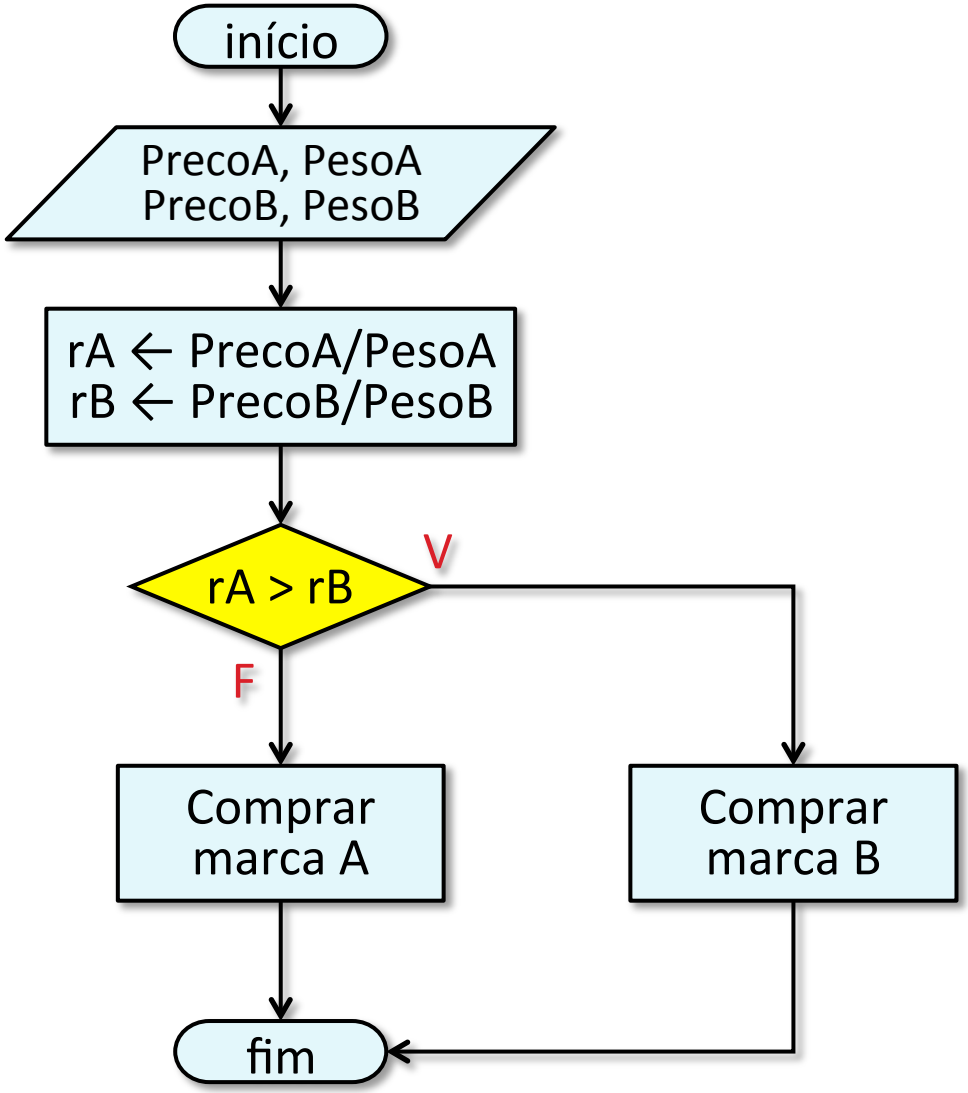
Problema 1

:: Definir Entradas e Saídas

	Grandeza	Unidade de medida	Faixa de valores
Entradas	PrecoA	R$	8,39
	PesoA	kg	0,4
	PrecoB	R$	20,30
	PesoB	kg	1,0
Saídas	Marca com menor Preço/Peso	---	A ou B

Problema 1

:: Solução



Problema 1

:: Testando a Solução (1)

```
Ler PrecoA,PrecoB,PesoA, PesoB
rA = PrecoA/PesoA
rB = PrecoB/PesoB

if rA>rB:
        print “Compre A”
else:
        print “Compre B”
```

Memória

PrecoA	8,39
PesoA	0,4
PrecoB	20,30
PesoB	1,0
rA	20,975
rB	20,30

Problema 1

:: Testando a Solução (2)

```
Ler PrecoA,PrecoB,PesoA, PesoB
rA = PrecoA/PesoA
rB = PrecoB/PesoB

if rA>rB:
      print “Compre A”
else:
      print “Compre B”
```

	Memória
PrecoA	7,00
PesoA	0,4
PrecoB	20
PesoB	1,0
rA	17,5
rB	20

Problema 2

```
Leia x,y
z ← x * y % 2

If z==0:
    l=55
else:
    l = 33
print l
```

Qual o valor de **L** para:

1. x = 6 e y = 7
2. x = 4 e y = 8
3. x = 129873645467 e
   y = 182163623686329

Problema 2

:: Solução

```
Leia x,y
z ← x * y % 2


If z==0:
    l=55
else:
    l = 33
print l
```

Se o produto x*y for par → zero
Se o produto x*y for ímpar → um

Fluxo de execução só passa aqui se (x*y) for par.

x	y	x*y	L
Par	Par	Par	55
Par	Ímpar	Par	55
Ímpar	Par	Par	55
Ímpar	Ímpar	Ímpar	33

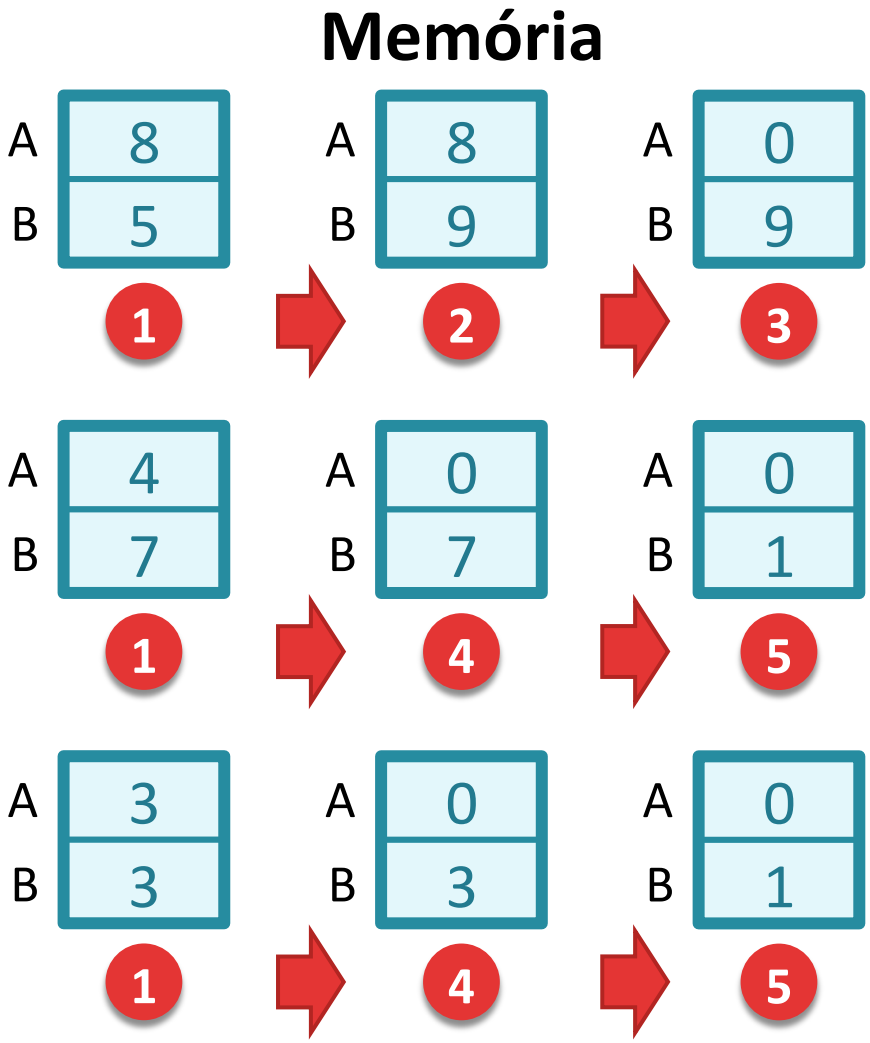
Problema 3

```
Leia A,B

if A>B:
    B = A + 1
    A = 0
else:
    A=0
    B = A+1
print A, B
```

A condição está servindo para alguma coisa, já que A e B parecem ser sempre os mesmos nos dois ramos?

Problema 3
:: Solução

```
1  Leia A,B

   if A>B:
        B = A + 1   2
        A = 0       3
   else:
        A=0         4
        B = A+1     5
   print A, B
```



Problema inicial
:: Nova tentativa de solução

- Dois jogadores lançam dados. Ganha aquele que tirar a face com maior número.
- Como determinar quem ganhou? Ou se houve empate?

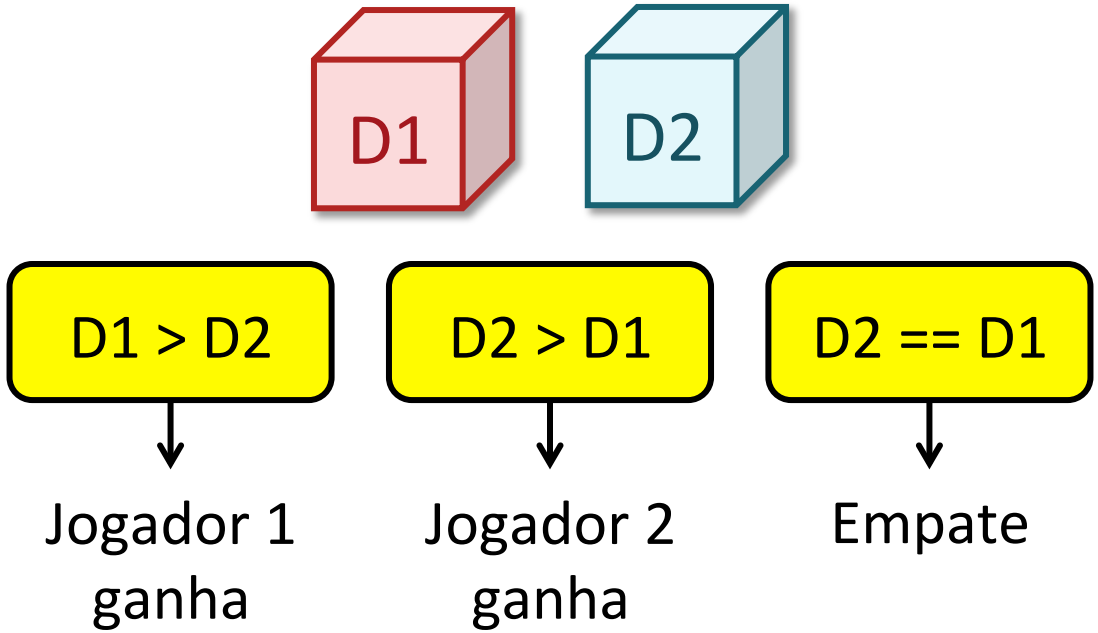
```
Ler D1,D2

if D1>D2:
        print " Jogador 1
venceu"
else:
        print "Jogador 2
Venceu"
```

Condições encadeadas

- Condições encadeadas (ou aninhadas) ocorrem quando há necessidade de se testar uma condição interna a outra, a partir de uma combinação de decisões.
- Tal situação pode ocorrer em virtude do leque de possibilidades apresentadas em um problema.

Condições encadeadas :: Problema inicial

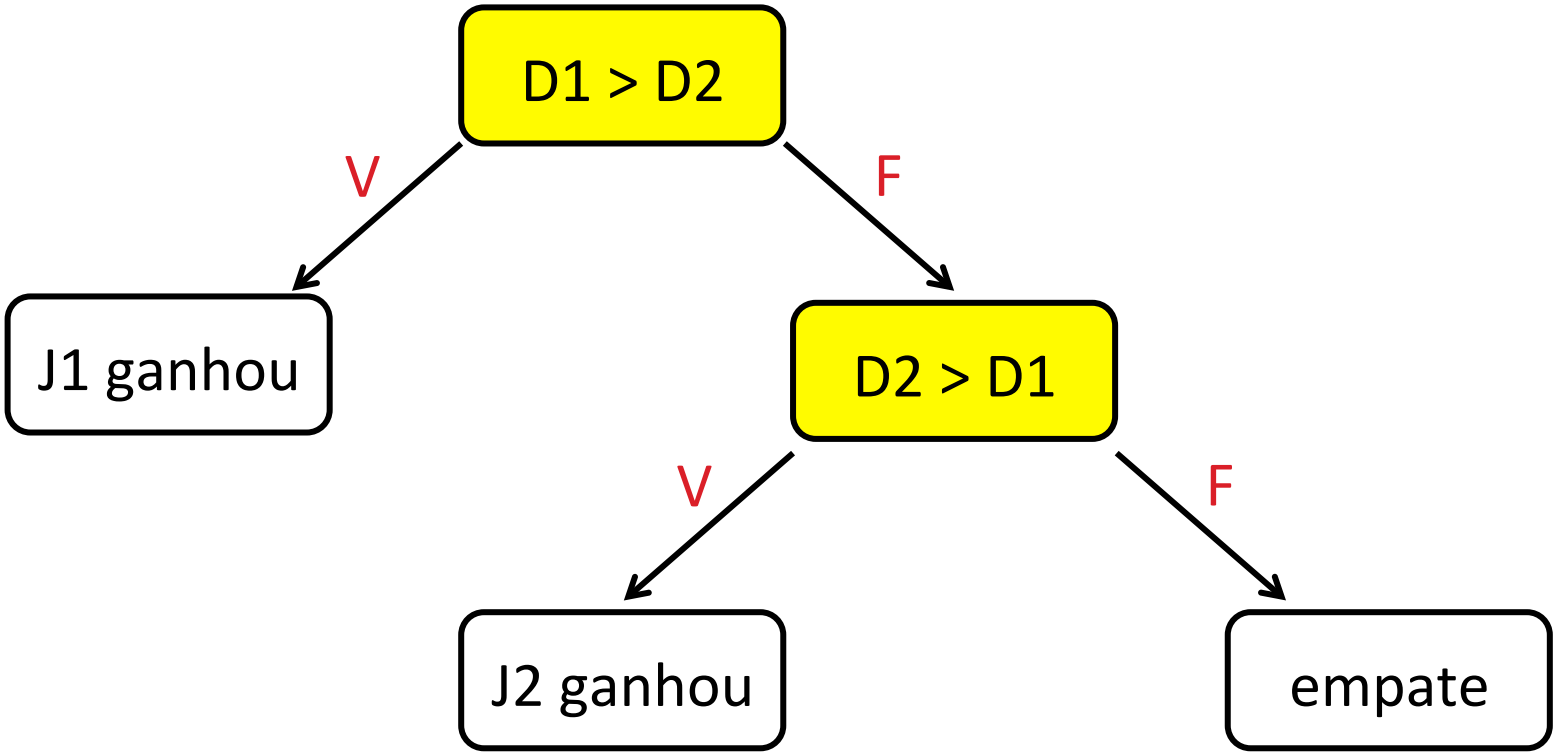
- No problema inicial, temos 03 situações possíveis:



- Porém, o algoritmo pode testar apenas uma única condição por vez.

Árvore de decisão

- Para estruturar corretamente o encadeamento das condições, devemos montar uma árvore de decisão.



Problema inicial
:: Solução Final

```
Ler D1,D2

if D1>D2:
        print " Jogador 1
venceu"
else:
        ...
```

Problema inicial
:: Solução Final

```
Ler D1,D2

if D1>D2:
        print " Jogador 1 venceu"
else:
        if D2>D1:
                print "Jogador 2 Venceu"
        else:
                print "Empate"
```

Problema inicial
:: Testando Solução Final

```
Ler D1,D2

if D1>D2:
        print " Jogador 1 venceu"
else:
        if D2>D1:
                print "Jogador 2 Venceu"
        else:
                print "Empate"
```

D1 = 6 e D2 = 1
D1 = 1 e D2 = 4
D1 = 5 e D2 = 5

Problema inicial

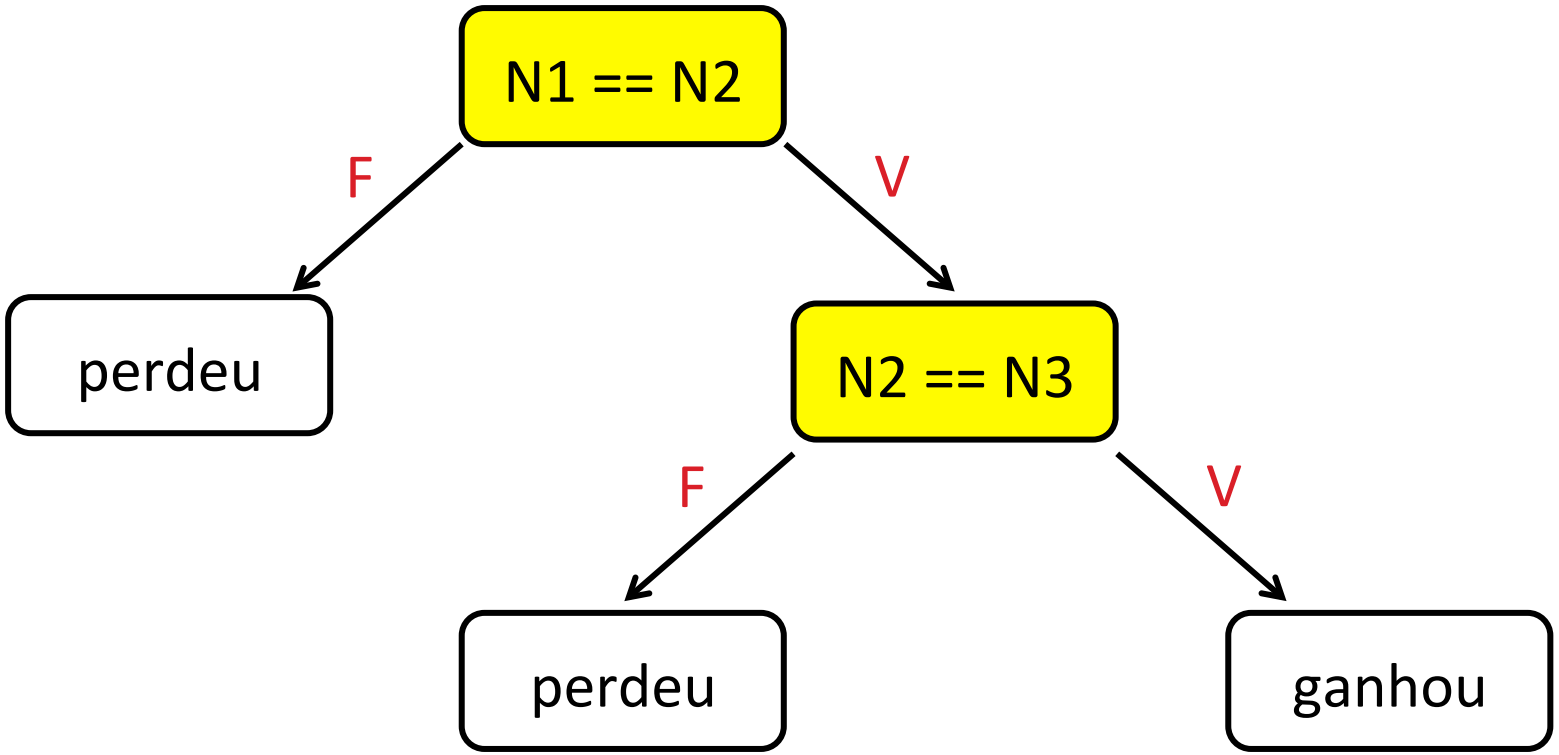
:: Existem outras soluções possíveis?

```
Ler D1,D2

if ?:
        print ?
else:
        if ?:
                print ?
        else:
                print ?
```

Problema 4

- Projete um algoritmo para uma máquina caça-níquel que gere 3 números aleatórios.
- Se os três números forem iguais, o jogador ganha. Caso contrário, ele perde.

Problema 4
:: Árvore de decisão



Problema 4

:: Solução parcial

```
Ler N1,N2,N3

if N1!=N2:
        print " Perdeu"
else:
```

Problema 4
:: Solução Final

```
Ler N1,N2,N3

if N1!=N2:
        print " Perdeu"
else:
        if N2!=N3:
                print " Perdeu"
        else:
                print " Ganhou"
```

Problema 4

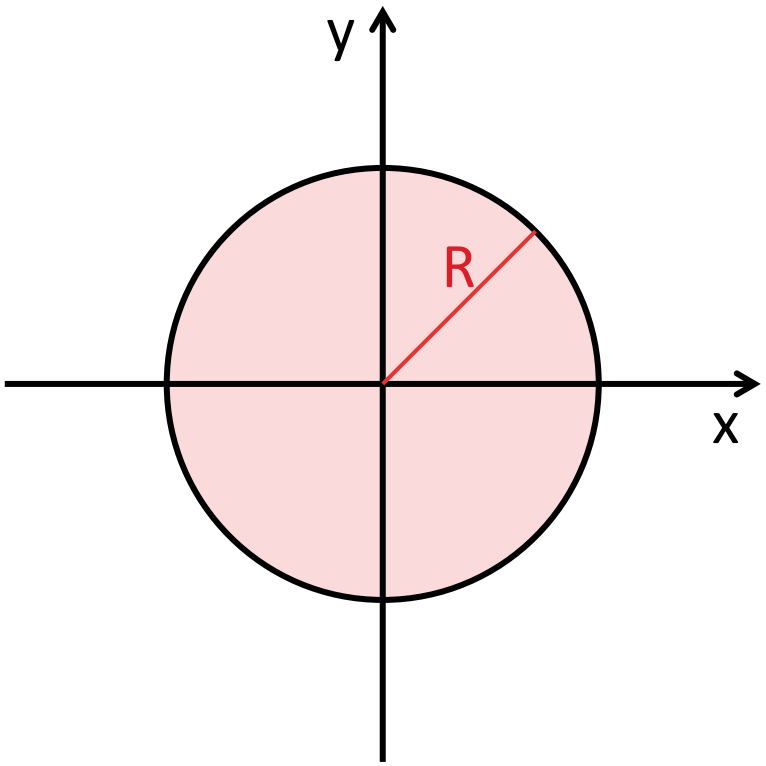
:: Testando Solução Final

```
Ler N1,N2,N3

if N1!=N2:
        print " Perdeu"
else:
        if N2!=N3:
                print " Perdeu"
        else:
                print " Ganhou"
```

N1 = 1, N2 = 2, N3 = 3
N1 = 3, N2 = 2, N3 = 3
N1 = 5, N2 = 5, N3 = 4
N1 = 7, N2 = 7, N3 = 7

Problema 5

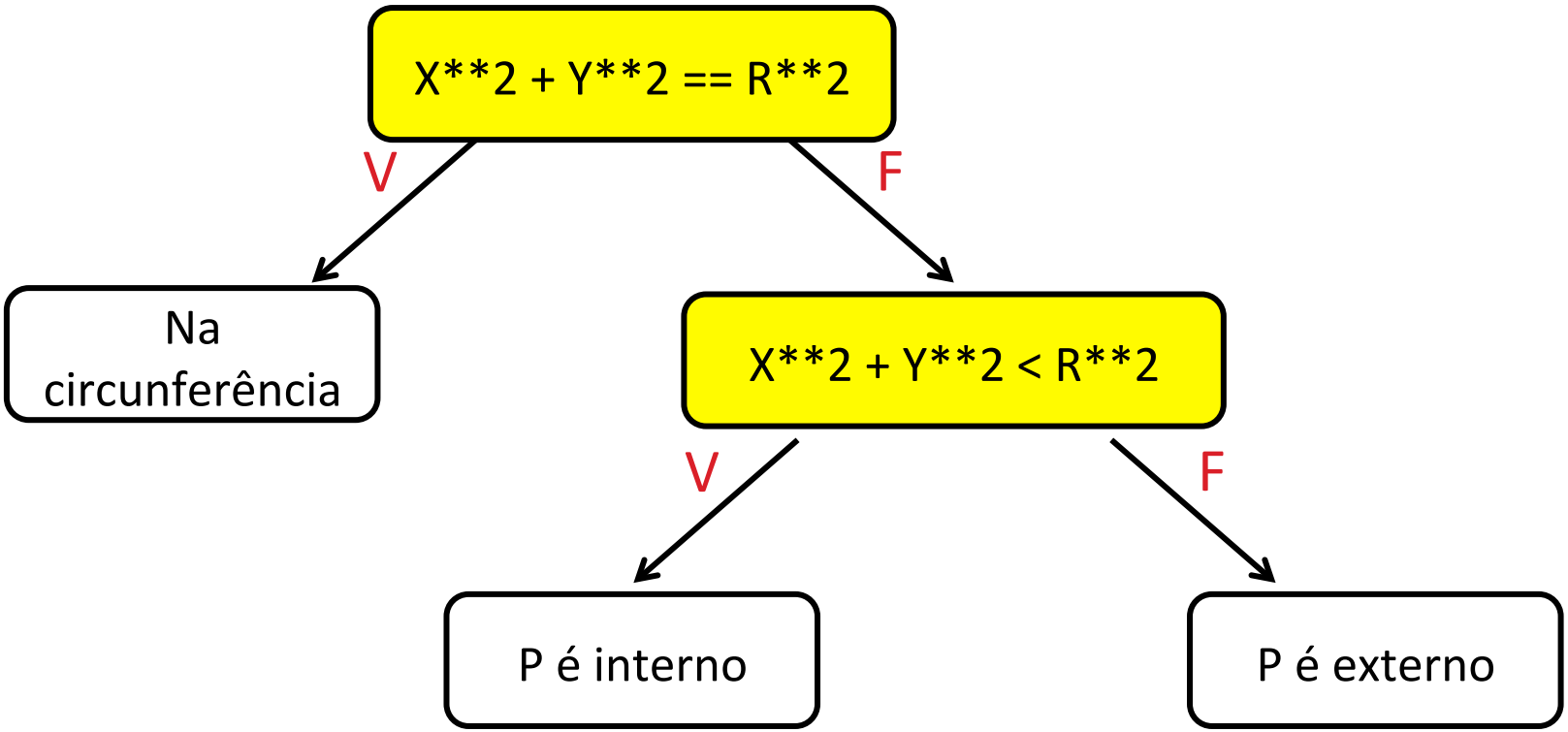
- A equação de um círculo de raio R é $x^2 + y^2 = R^2$.
- Escreva um algoritmo que, dado um ponto P qualquer, verifique se ele se encontra ou não no interior da região do plano delimitada pelo círculo.



Problema 5
:: Definir Entradas e Saídas

	Grandeza	Unidade de medida	Faixa de valores
Entradas	R	---	> 0
	Coordenada X de P	---	$\mathbb{R}$
	Coordenada Y de P	---	$\mathbb{R}$
Saídas	Localização de P	---	Interior, exterior, na circunferência

Problema 5
:: Árvore de decisão



Problema 5
:: Solução parcial

```
Ler R,X,Y

if C1:
        print " Na Circunferencia"
else:
        if...
```

1 X**2 + Y**2 == R**2

Problema 5
:: Solução Final

```
Ler R,X,Y

if C1:
        print " Na Circunferencia"
else:
        if C2:
                print "Interno"
        else:
                print "Externo"
```

C1 `X**2 + Y**2 == R**2`

C2 `X**2 + Y**2 < R**2`

Problema 5
:: Testando Solução Final

```
Ler R,X,Y

if C1:
        print " Na Circunferencia"
else:
        if C2:
                print "Interno"
        else:
                print "Externo"
```

C1 `X**2 + Y**2 == R**2`

C2 `X**2 + Y**2 < R**2`

R = 1, X = 3, Y = 4
R = 10, X = -4, Y = -3
R = 13, X = 5, Y = -12

Projeto de estruturas condicionais

- Todo comando de um fluxograma deve contribuir para a solução do problema.
- Todas as alternativas de uma estrutura condicional devem ter a possibilidade de serem executadas para alguma combinação de entradas.
- Veja o contra-exemplo a seguir.

Projeto de estruturas condicionais

:: Erros a evitar

```
Ler A

if A>0:
        print “Positivo”
else:
        if A<=0:
                print “Não positivo”
        else:
                print “Chega aqui?”
```

Projeto de estruturas condicionais

:: Correção 1

```
Ler A

if A>0:
        print “Positivo”
else:
        print “Não positivo”
```

Projeto de estruturas condicionais

:: Correção 2

```
Ler A

if A>0:
        print “Positivo”
else:
        if A<0:
                print “Negativo”
        else:
                print “Zero”
```

Referências bibliográficas

- Menezes, Nilo Ney Coutinho (2010). Introdução à Programação com Python. Editora Novatec.

- Farrer, Harry (2011). Algoritmos Estruturados, 3ª edição. Editora LTC.

- Forbellone, A. L. V.; Eberspächer, H. F. (2006) Lógica de Programação, 3ª edição. Pearson.

- HETLAND, Magnus Lie (2008). Beginning Python: From Novice to Professional. Springer eBooks, 2ª edição. Disponível em: http://dx.doi.org/10.1007/978-1-4302-0634-7.

Dúvidas?