Programadoras da NASA no programa espacial de viaxe á lúa (1969)

Katherine Johnson: **matemática** e programadora da NASA
Primeira civil en recibir a **Medalla de Ouro** do Congreso USA

Mary Jackson: enseñeira da NASA

Dorothy Vaugham: programadora en **Fortran** do primeiro ordenador da NASA

Película
“Figuras ocultas”

Operadores relacionais (I)

- Definen as relacións de equivalencia (igualdade) e de orde numérica (con caracteres, a orde alfabética)
- Operadores: < <= > >= == /=
- Exemplos: x < 3, y >= z
- Operandos numéricos, resultado lóxico
- x < y: .*true*. se x < y, .*false*. en caso contrario
- x > y: .*true*. se x > y, .*false*. en caso contrario
- x == y: .*true*. se x e y son iguais, .*false*. en caso contrario
- *x /= y*: .*true*. se x e y son distintos, .*false*. en caso contrario

Operadores relacionais (II)

- Permiten definir condicións de igualdade / desigualdade ou maior/menor sobre números ou caracteres (orde alfabética)
- Úsanse en sentenzas de selección para crear distintas "rutas" de execución do programa, dependendo do cumprimento ou non dunha condición
- Sobre vectores e matrices, aplícanse por compoñentes:
  integer :: x(3)=(/1,2,3/)
  *print *,x>2 => F F T*

```
integer :: a(2,2)
forall(i=1:2,j=1:2) a(i,j)=i*i+j
print *,mod(a,2)==0
```

- Función *count*: nº de elementos dun vector/matriz que cumpren unha relación: *count(x>5)*, *count(x!=0)*, *count(mod(x,2)==1)*
- Caracteres: *character(10) :: s='ola'*

  s=='ola': T s>'pepe': F

```
integer :: a(2,2)
forall(i=1:2,j=1:2) a(i,j)=i*i+j
print *,count(mod(a,2)==1)
```

Operadores lóxicos (I)

- Combinan unha ou dúas condicións (definidas cos operadores relacionais) mediante relacións de:
  - **Conxunción**: *condición1* e tamén *condición2*
  - **Disxunción**: *condición1* ou *condición2*
  - **Negación**: non é certo *condición1*
  - **Equivalencia lóxica**: *condición1* e *condición2* son iguais (ou ben ambas se cumpren ou ben ningunha se cumpre)
  - **Non equivalencia lóxica**: *condición1* e *condición2* son distintas (se se cumpre unha non se cumpre a outra e ao revés)

Operadores lóxicos (II)

- Operadores:
  - Conxunción: .*and*.
  - Disxunción: .*or*.
  - Negación : .*not*.
  - Equivalencia lóxica (condicións iguais): .*eqv*.
  - Non equivalencia lóxica (condicións distintas): .*neqv*.
- Operandos lóxicos (resultado de operadores relacionais ou lóxicos). Resultado lóxico

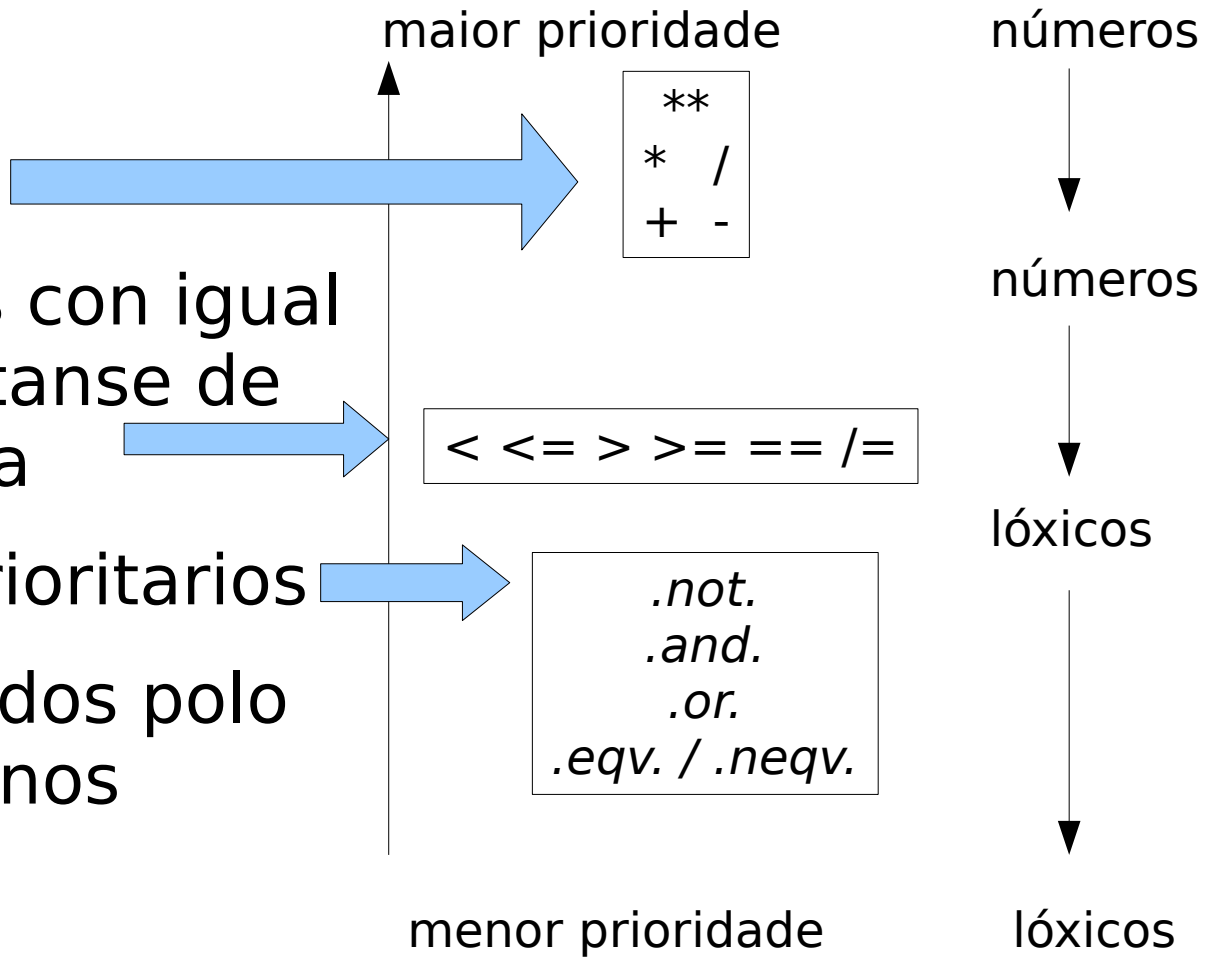
a	*.true.*	*.false.*	*.true.*	*.false.*
b	*.true.*	*.true.*	*.false.*	*.false.*
a.and.b	*.true.*	*.false.*	*.false.*	*.false.*
a.or.b	*.true.*	*.true.*	*.true.*	*.false.*
.not.a	*.false.*	*.true.*		
a.eqv.b	*.true.*	*.false.*	*.false.*	*.true.*
a.neqv.b	*.false.*	*.true.*	*.true.*	*.false.*

Exemplos de operadores lóxicos

- AND: *x>5.and.x<=7*: x debe estar en (5,7] para que se cumpra a condición
- OR: *x==2.or.x==3*: x debe ser 2 ou 3 para que a condición sexa .true.
- NOT: *.not.(x>3)* é certa so se x<=3
- EQV: *mod(x,2)==0.eqv.mod(x,3)==0*: é certa se x é múltiplo de 2 e de 3, e tamén se x non é múltiplo de 2 nin de 3
- NEQV: *mod(x,2)==0.neqv.mod(x,3)==0*: é certa se x é múltiplo de 2 pero non de 3, e tamén se x non é múltiplo de 2 pero si de 3

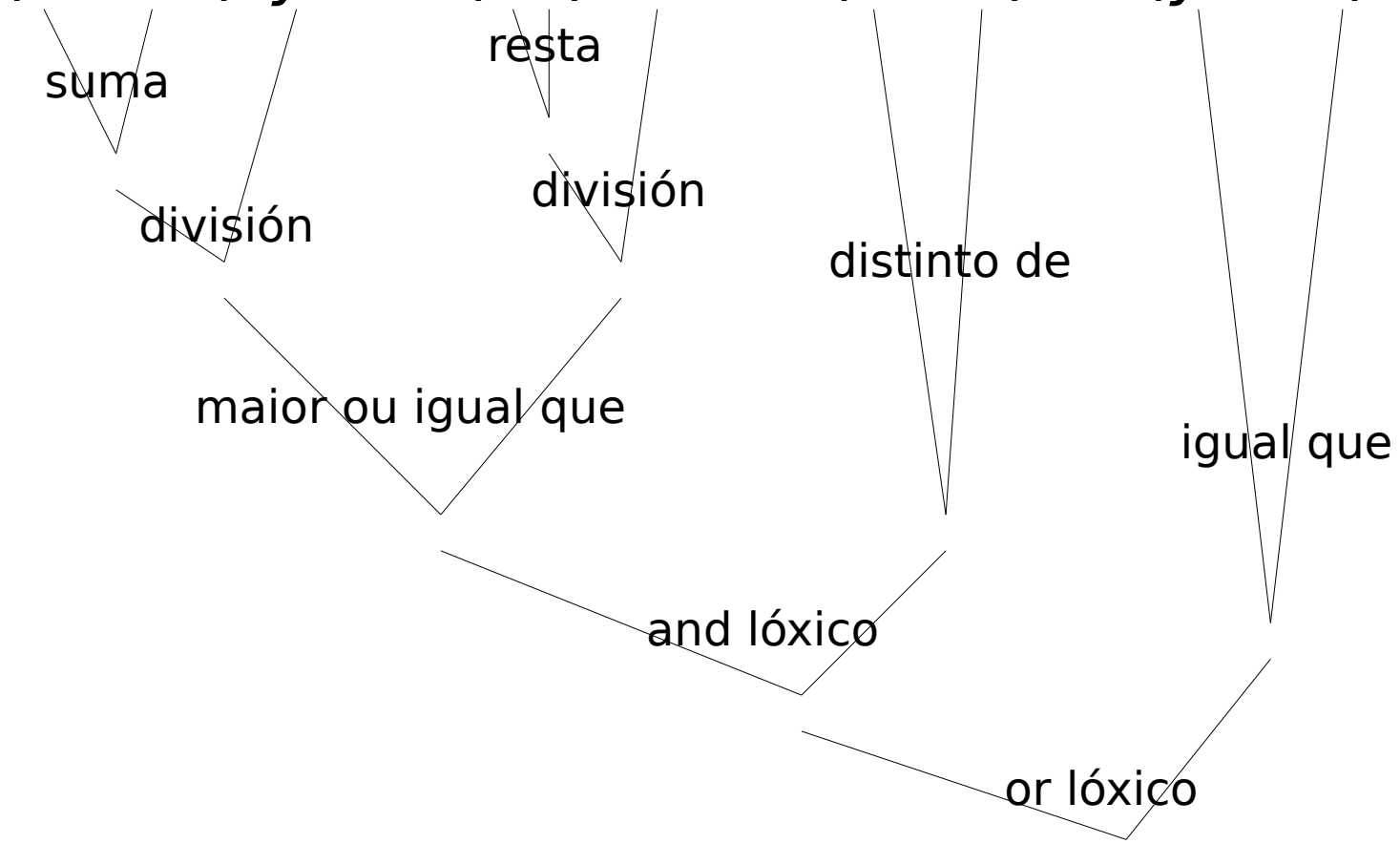
Precedencia de operadores

- Aritméticos: máis prioritarios
- Relacionais: todos con igual prioridade, execútanse de esquerda a dereita
- Lóxicos: menos prioritarios
- Operadores definidos polo usuario: os de menos prioridade



Exemplo de prioridades de execución

*(z+3.4)/y >= (z-t)/x.*and.*(x/=0).or.(y==z)*



Funcións lóxicas *all* e *any* para vectores/matrices

- *all(v/=0)*: *true.* se tódolos elementos do vector *v* son non nulos.
- *all(a>0)*: *true* se tódolos elementos da matriz *a* son positivos
- *all(a,1)*: executa o *all* por columnas (*a* debe ser *logical*)
- *all(a,2)*: executa o *all* por filas
- *any(a>5)*: *true* se *a* ten algún elemento maior que 5
- *any(mod(a,2)==0,1)*: *true* para columnas con elemento par.
- *any(a==3,2)*: *true* para as filas cun 3