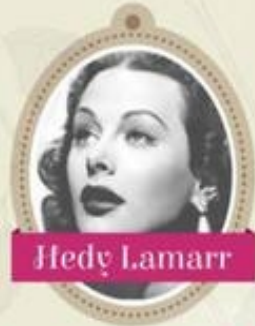


**PROGRAMACIÓN
ESTRUCTURADA EN
FORTRAN**

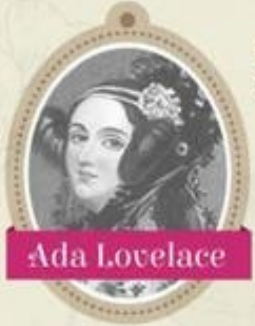
As mulleres na programación e na informática

set ENJOY SAFER TECHNOLOGY™

Actriz de Hollywood, inventora de la tecnología precursora del Wifi, Bluetooth y GPS.



Hedy Lamarr



Ada Lovelace

La primera programadora y madre de la programación informática.



Frances E. Allen

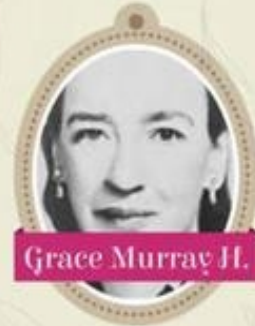
Pionera en la automatización de tareas paralelas. En 2007, recibió el premio Turing, equivalente al Nobel de Informática.



Jude Milhon

Creadora del ciberpunk, programadora, escritora, activista, defensora de los ciberderechos.

Sin las mujeres,
la informática no existiría
tal como la conocemos



Grace Murray H.

Desarrolló el primer compilador para un lenguaje de programación.



Evelyn Berezin

Inventó en 1953 el ordenador de oficina. Desarrolló el primer sistema de reserva de vuelos. Conocida como la madre de los procesadores de texto.



Lynn Conway

Pionera en el campo de diseño de chips microelectrónicos.



Top Secret Rosies

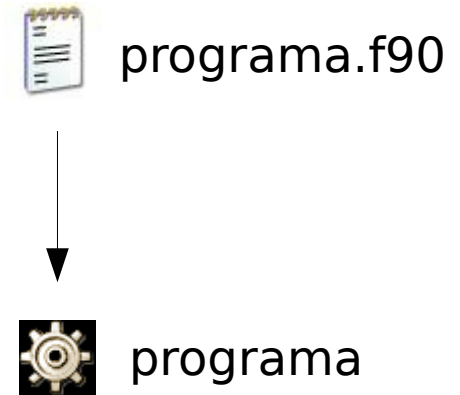
Seis especialistas en matemáticas que, en 1946, programaron el primer computador ENIAC.

Proceso de programación en Fortran

- Escritura dun programa fonte *programa.f90* co **editor** *kate*
- Compilación do programa dende unha **terminal** de comandos co compilador *f95*:

▶ *f95 programa.f90 -o programa*

- Corrección de erros de compilación
- Creación de programa executábel: *programa*
- Execución de *programa* (na terminal): *programa*
- Corrección de erros de execución e lóxicos (edición, compilación, execución)
- Coidado: non poñas *programa.f90* no canto de *programa*: se o fas, sobrescribes o *programa.f90* co executable



Programa básico en Fortran

- Comeza con sentenza: “*program nome*”
- Remata con “*end program nome*”
- “*nome*” é o nome do programa: non pode haber variábeis nin subprogramas con ese nome
- Logo de *program*:
 - declaración de variábeis (ao principio)
 - sentenzas executábeis: operacións, entrada / saída, ...

Programa básico en Fortran

- Sentenza “*stop*”: remata a execución
stop 'mensaxe': remata e imprime a mensaxe
- Exemplo de programa básico:

```
program basico  
integer :: x  
x=5  
print *, x  
end program basico
```

Imprime o nº 5 na terminal de comandos



```
delgado : bash — Konsole  
Ficheiro  Editar  Vista  Marcadores  Configuración  Axuda  
delgado : bash  
delgado@ctdesk209:~$ cat programa.f90  
program basico  
integer :: x  
x=5  
print *,x  
end program basico  
delgado@ctdesk209:~$ f95 programa.f90 -o programa.out  
delgado@ctdesk209:~$ programa.out  
5  
delgado@ctdesk209:~$ _
```

- Liñas de comentarios: con ! ao comezo da liña

Entrada e saída estándar

- Entrada de datos estándar: le datos dende o teclado e almacénaos en variábeis
 - Sentenza *read*: *read *, x*
 - Pódese producir un erro se a variábel é numérica e introducimos un carácter (p.ex.)
- Saída de datos estándar: visualiza na terminal (pantalla)
 - Sentenza *print*: *print *, 'resultado=', 2*x*
 - Con formato: *print ("n=",i0)',n*
print ("x=",f10.6)',x
 - ← Ancho do nº enteiro
 - ← Ancho e nº de decimais

Estructura de selección básica

- IF/ELSE: avalía unha serie de condicións e executa unhas sentenzas ou outras dependendo de que condicións se cumpren

```
if(condicion) then  
    sentenzas1  
else if(condición2) then  
    sentenzas2  
else  
    sentenzas3  
endif
```

```
if (x > 3) then  
    print *, 'alto'  
else  
    print *, 'baixo'  
end if
```

Estrutura iterativa *do* definida

- Permite repetir unha ou varias sentenzas un certo número de veces

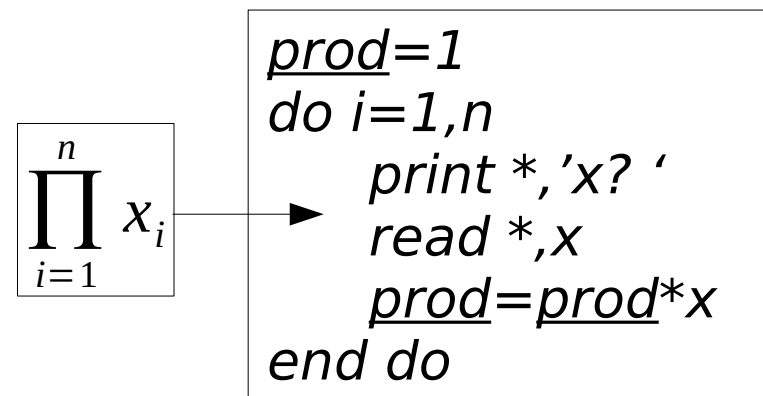
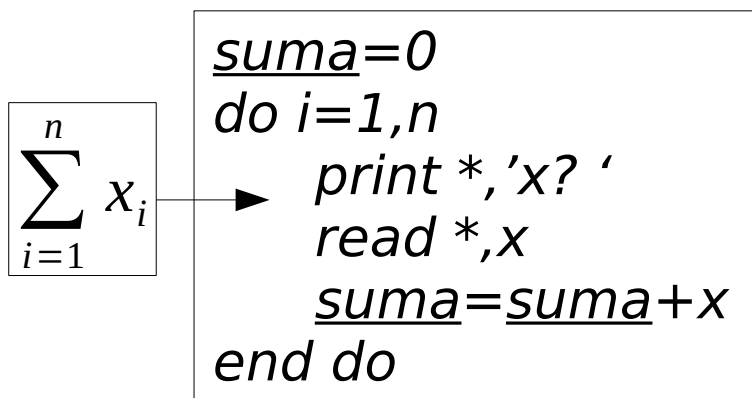
```
do var = ini, fin, paso  
    sentenzas  
end do
```

```
do i = 1, 10  
    print *, (i + 2)/3  
end do
```

- Repite as sentenzas dende que a variábel *var* adopta o valor *ini*, ata que adopta o valor *fin*.
- En cada repetición, *var* incrementábase en *paso*
- O incremento *paso* vale 1 por defecto

Acumulador

- Variable que aparece na esquerda e na dereita dunha asignación dentro dun bucle *do*: Exemplo: $x=x+i$; $p=p*j$
- Emprégase en operacións cun número variábel de operandos, que require un bucle *do*.
- O acumulador debe inicializarse co elemento neutro da operación.
- Acumulador de suma e de produto dun número variábel (n) de valores:



Estructuras iterativas *do* indefinidas

- Bucle *do/exit*: repite as sentenzas e remata cando se cumpre unha condición.

```
do
  sentenzas
  if(condición) exit
  sentenzas
end do
```

```
x=0;dx=0.1
do
  x=x+dx
  print *,x,x*x
  if(x>1) exit
end do
```

```
x=0;dx=0.1
do
  if(x>1) exit
  x=x+dx
  print *,x,sin(x)
end do
```

- Bucle *do while*: repite as sentenzas mentres se cumpre unha condición.

```
do while(condición)
  sentenzas
end do
```

```
x=0;dx=0.1
do while(x<1)
  print *,x,x*x
  x=x+dx
end do
```