




# PROGRAMACION Y COMPUTACION

---

Departamento de Sistemas  
Universidad de la Frontera



# Capítulo 4:

## Algoritmos - PSeInt

---

### Agenda:

- Estructura de un Algoritmo en pseudocódigo
- PSeInt-PS<sub>eudo</sub> I<sub>nterprete</sub>
- Símbolos gráficos
- Operadores y funciones que soporta PSeInt
- Instrucciones secuenciales
- Instrucciones de control



# Algoritmos - PSeInt

---

Representación de un Algoritmo (**recordatorio**):

- Lenguaje natural
- Seudocódigo
- Diagramas de flujo
- Lenguajes de programación



# Algoritmos - PSeInt

---

Representación en **Lenguaje natural**

Problema: Sumar 2 números.

- Inicio suma
- Ingresar primer número
- Guardar número en variable a
- Ingresar segundo número
- Guardar número en variable b
- Sumar a y b
- Guardar resultado en c
- Mostrar c
- Fin



# Algoritmos - PSeInt

---

Representación en Lenguaje natural

Desventaja:

- Ambiguo
- Extenso



# Algoritmos - PSeInt

---

## Representación en **seudocódigo**

- La representación del algoritmo que se asemeja a los lenguajes de programación pero conserva elementos del lenguaje natural.
- La estructura de un algoritmos en pseudocódigo se compone de:
  - ✓ **Cabecera**
  - ✓ **Declaraciones**
  - ✓ **Cuerpo**



# Algoritmos - PSeInt

---

## Representación en pseudocódigo

- La **cabecera** es la parte del algoritmo que posee el nombre del algoritmo.
- Las **declaraciones** son las variables y constantes que utilizará el algoritmo para resolver el problema.
- El **cuerpo** son el conjunto de instrucciones o acciones que están entre el Inicio y el Fin del algoritmo



# Algoritmos - PSeInt

---

PSeInt-PSeudo Interprete

<http://pseint.sourceforge.net>

- PSeInt está pensado para **asistir** a los estudiantes que se inician en la **construcción de programas** o algoritmos computacionales.
- El pseudocódigo se utiliza como primer contacto para **introducir conceptos básicos** como el uso de estructuras de control, expresiones, variables, etc.
- Este software **facilita** al principiante la tarea de **escribir** algoritmos.





# Algoritmos - PSeInt

---

## PSeInt-PSeudo Interprete

- Permite **generar y editar** el diagrama de flujo del algoritmo.
- Permite la edición simultánea de **múltiples** algoritmos.
- Determina y **marca** claramente **errores** de sintaxis (mientras escribe) y en tiempo de ejecución.
- Es **multiplataforma** (probado en Microsoft Windows, GNU/Linux y Mac OS X).
- Es totalmente **libre y gratuito** (licencia GPL)



# Algoritmos - PSeInt

---

## PSeInt técnicamente:

- Es un software que **interpreta** pseudocódigo
- Es de **sintaxis** sencilla
- Maneja las **estructuras de control** básicas
- Maneja solo 3 tipos de datos básicos: **numérico**, **alfanumérico** y **lógico** (verdadero-falso).
- Maneja estructuras de datos: **arreglos**



# Algoritmos - PSeInt

---

## Estructura del algoritmo en pseudocódigo

```
Proceso SinTitulo
    accion 1;
    accion 1;
    .
    .
    .
    accion n;
FinProceso
```

- La sección "Proceso SinTitulo" es la **cabecera** del algoritmo
- La sección "acción 1, acción 1,..." es el **cuerpo** del algoritmo
- La sección de **declaraciones** se omite dado que el software (PSeInt) se encarga de asignarle el **tipo de dato** a cada **variable** según el uso.



# Algoritmos - PSeInt

---

## Estructura del algoritmo en pseudocódigo

```
Proceso SinTitulo
    accion 1;
    accion 1;
    .
    .
    .
    accion n;
FinProceso
```

- Comienza con la palabra clave *Proceso* seguida del nombre del programa (algoritmo),
- Le sigue una secuencia de *acciones* (instrucciones) cada una terminada en *punto y coma*.
- Finaliza con la palabra clave *FinProceso*.



# Algoritmos - PSeInt

---

## Estructura del algoritmo en pseudocódigo

```
Proceso SinTitulo
    accion 1;
    accion 1;
    .
    .
    .
    accion n;
FinProceso
```

- Las acciones incluyen operaciones de **entrada** y **salida**, **asignaciones** de variables, **condicionales** si-entonces o de selección múltiple y/o **ciclos** mientras, repetir o para.



# Algoritmos - PSeInt

---

Estructura del algoritmo pseudocódigo, ejemplo

```
1  Proceso suma
2      Escribir 'INGRESE PRIMER NUMERO';
3      Leer a;
4      Escribir 'INGRESE SEGUNDO NUMERO';
5      Leer b;
6      c<-a+b;
7      Escribir 'LA SUMA ES:',c;
8  FinProceso
```

Estructura del algoritmo en Java

```
public int suma(int a, int b){
    int c;
    c = a + b;
    return c;
}
```

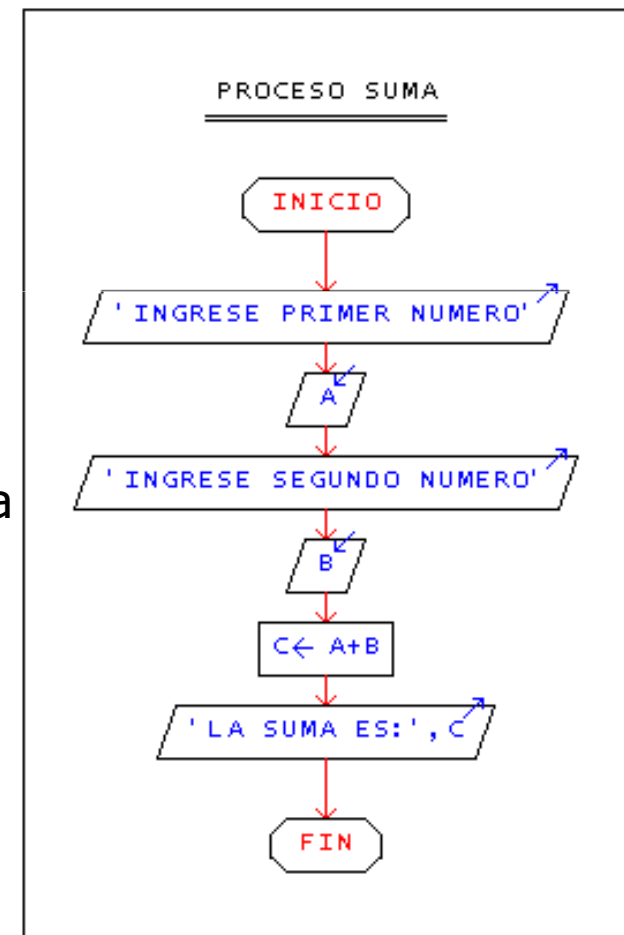
# Algoritmos - PSeInt

## Representación del algoritmo en **diagrama de flujo**

Para generar un diagrama de flujo  
En PSeInt debemos presionar el  
botón:

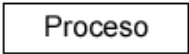
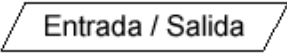
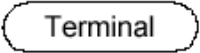

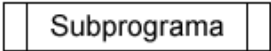




La representación mediante diagrama  
de flujo es una **descripción gráfica** de  
un algoritmo **utilizando símbolos**.



# Algoritmos - PSeInt

## Símbolos gráficos para representar Diagramas de Flujo

<i>Símbolos gráficos más utilizados para dibujar algoritmos por medio de diagramas de flujo (ordinogramas):</i>	
<i>Símbolo</i>	<i>Descripción (significado):</i>
	Instrucción de asignación
	Instrucción de entrada o de salida
	Inicio o Fin del algoritmo
	Instrucción de control
	Llamada a un subprograma
	Indica el orden de las acciones del algoritmo
	Conector de reagrupamiento de una instrucción de control





# Algoritmos - PSeInt

---

## Operadores que soporta PSeInt

Operador	Significado	Ejemplo
<i>Relacionales</i>		
>	Mayor que	3>2
<	Menor que	'ABC'<'abc'
=	Igual que	4=3
<=	Menor o igual que	'a'<='b'
>=	Mayor o igual que	4>=5
<i>Logicos</i>		
& ó Y	Conjunción (y).	(7>4) & (2=1) //falso
ó O	Disyunción (o).	(1=1   2=1) //verdadero
~ ó NO	Negación (no).	~(2<5) //falso
<i>Algebraicos</i>		
+	Suma	total <- cant1 + cant2
-	Resta	stock <- disp - venta
*	Multiplicación	area <- base * altura
/	División	porc <- 100 * parte / total
^	Potenciación	sup <- 3.41 * radio ^ 2
% ó MOD	Módulo (resto de la división entera)	resto <- num MOD div



# Algoritmos - PSeInt

---

**Funciones** que soporta PSeInt

<i>Función</i>	<i>Significado</i>
RC(X)	Raíz Cuadrada de X
ABS(X)	Valor Absoluto de X
LN(X)	Logaritmo Natural de X
EXP(X)	Función Exponencial de X
SEN(X)	Seno de X
COS(X)	Coseno de X
TAN(X)	Tangente de X
ASEN(X)	Arcoseno de X
ACOS(X)	Arcocoseno de X
ATAN(X)	Arcotangente de X
TRUNC(X)	Parte entera de X
REDON(X)	Entero más cercano a X
AZAR(X)	Entero aleatorio entre 0 y x-1



# Algoritmos - PSeInt

---

## Comentarios de línea:

- El carácter que se usa para comentario de línea es //
- Puede ir al principio o a continuación de una instrucción

Ejemplo (PSeInt, Java)

// Este es un comentario

<instrucción>      // Comentario

## Comentarios de párrafo :

El carácter que indica comentario de párrafo es: Inicio /\* y término \*/

Ejemplo (Java)

/\* Este es un comentario de párrafo. Útil cuando se quiere  
Documentar el programa \*/



# Algoritmos - PSeInt

---

## Identificadores

- Los identificadores, o nombres de variables, deben constar **sólo de letras y números**,
- **Comienzan** siempre con una letra, y no pueden ser palabras reservadas (como para, mientras, y, no, etc...)

## FinProceso

- No puede haber instrucciones fuera del programa, aunque si comentarios.

## Constantes

- Las constantes de tipo carácter se escriben entre comillas ("").
- En las constantes numéricas, el punto ( . ) es el separador decimal.
- Las constantes lógicas son *Verdadero* y *Falso*.



# Algoritmos - PSeInt

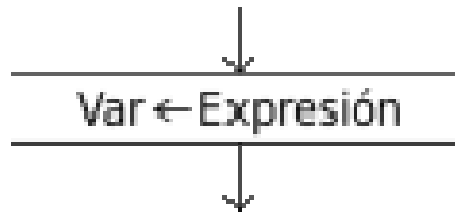
---

## Asignación

La instrucción de **asignación** permite almacenar un valor en una variable.

**<variable> ← <expresión> ;**

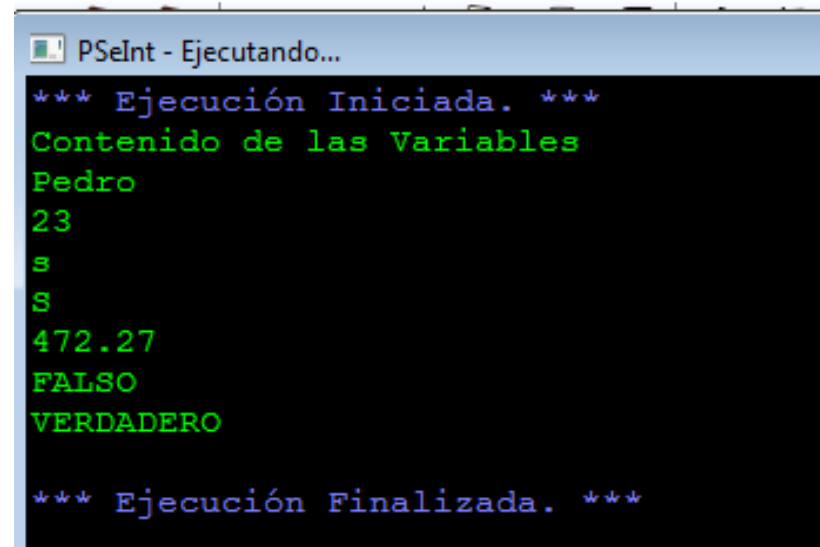
- Al ejecutarse la asignación, primero se evalúa la expresión de la derecha y luego se asigna el resultado a la variable de la izquierda.
- El tipo de la variable y el de la expresión deben coincidir.



# Algoritmos - PSeInt

## Ejemplo Prog00

```
1  Proceso Prog00
2      Nombre <- "Pedro";           // variable alfanumérica
3      Edad <- 23;                  // variable numérica (entera)
4      LetraMin <- "s";             // variable alfanumérica
5      LetraMay <- "S";             // variable alfanumérica
6      Dolar <- 472.27;             // variable numérica (real)
7      Fue1 <- (3==4);              // variable booleana
8      Fue2 <- ("a"=="a");          // variable booleana
9      Escribir ("Contenido de las Variables");
10     Escribir Nombre;
11     Escribir Edad;
12     Escribir LetraMin;
13     Escribir LetraMay;
14     Escribir Dolar;
15     Escribir Fue1;
16     Escribir Fue2;
17 FinProceso
```



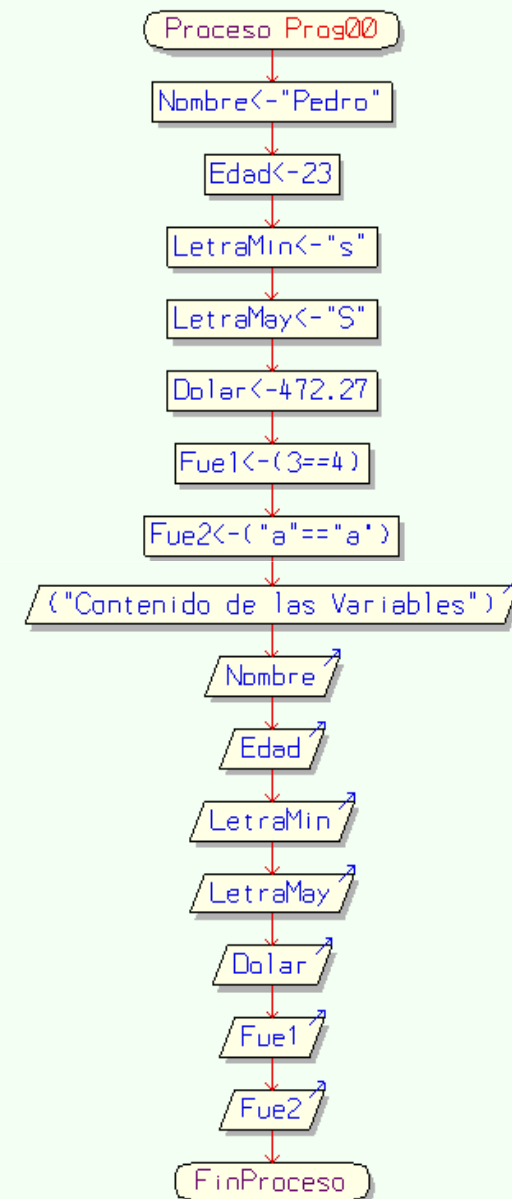
The screenshot shows a window titled "PSeInt - Ejecutando...". The output text is as follows:

```
*** Ejecución Iniciada. ***
Contenido de las Variables
Pedro
23
s
S
472.27
FALSO
VERDADERO
*** Ejecución Finalizada. ***
```

# Algoritmos - PSeInt

## Ejemplo Prog00 –Diagrama de Flujo

```
PSInt - Ejecutando...
*** Ejecución Iniciada. ***
Contenido de las Variables
Pedro
23
s
S
472.27
FALSO
VERDADERO
*** Ejecución Finalizada. ***
```



# Algoritmos - PSeInt

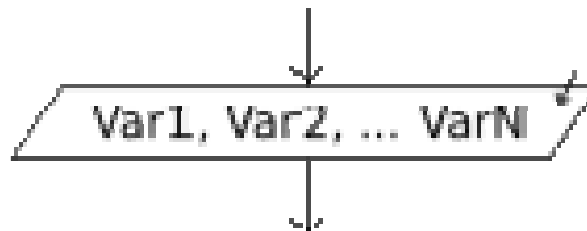
---

## Entradas

La instrucción **Leer** permite ingresar información desde el ambiente.

**Leer** <variable> , <variable2> , ... , <variableN> ;

- Esta instrucción lee N valores desde el ambiente (en este caso el teclado) y los asigna a las N variables mencionadas.
- Pueden incluirse una o más variables, por lo tanto el comando leerá uno o más valores.





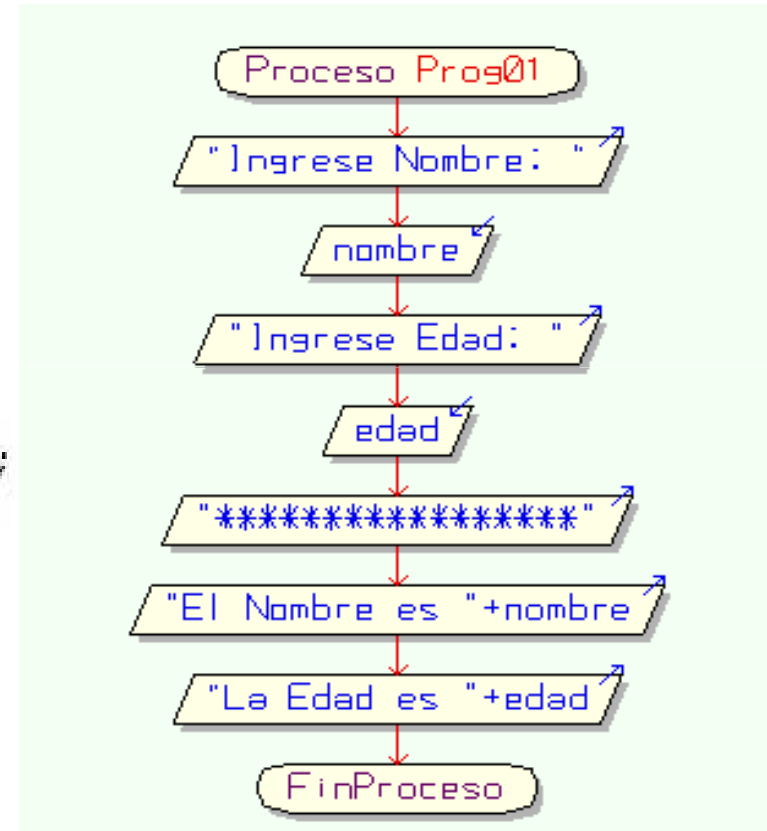
# Algoritmos - PSeInt

## Ejemplo Progr01

```
1  Proceso Progr01
2      Escribir "Ingrese Nombre: ";
3      Leer nombre
4      Escribir "Ingrese Edad: ";
5      Leer edad
6      Escribir "*****";
7      Escribir "El Nombre es "+nombre;
8      Escribir "La Edad es "+edad;
9  FinProceso
10
```

```
*** Ejecución Iniciada. ***
Ingrese Nombre:
> Pedro
Ingrese Edad:
> 25
*****
El Nombre es Pedro
La Edad es 25

*** Ejecución Finalizada. ***
```





# Algoritmos - PSeInt

---

## Salidas

La instrucción **Escribir** permite mostrar valores al ambiente.

**Escribir** <expr1> , <expr2> , ... , <exprN> ;

- Esta instrucción imprime al ambiente (en este caso en la pantalla) los valores obtenidos de evaluar N expresiones.
- Dado que puede incluir una o más expresiones, mostrará uno o más valores.



# Algoritmos - PSeInt

## Ejemplo Prog02

```
1  Proceso Prog02
2      Escribir ( "Ingrese Nombre, Apellido, Edad: ");
3      Leer Ciudad, Apellido, Edad;
4      Escribir ("=====");
5      Escribir (Ciudad+Apellido+Edad);
6      Escribir (Ciudad+" "+Apellido+" "+Edad);
7      Escribir (Ciudad);
8      Escribir (Apellido);
9      Escribir (Edad);
10     Escribir Ciudad, Apellido, Edad;
11 FinProceso
```

```
*** Ejecución Iniciada. ***
Ingrese Nombre, Apellido, Edad:
> Pedro
> Pereira
> 20
=====
PedroPereira20
Pedro Pereira 20
Pedro
Pereira
20
PedroPereira20
*** Ejecución Finalizada. ***
```



# Algoritmos - PSeInt

---

## Condicional Si-Entonces-Sino

La secuencia de instrucciones ejecutadas por la instrucción **Si-Entonces-Sino** depende del valor de una condición lógica.

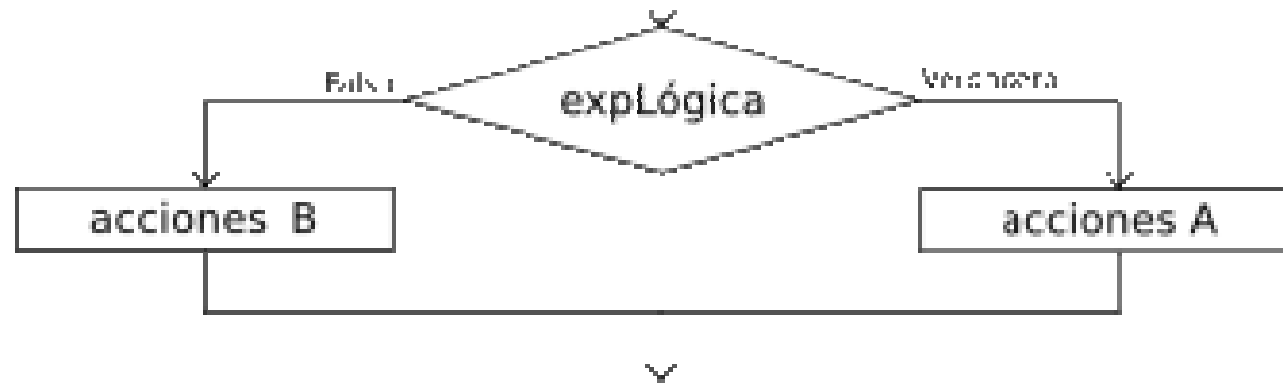
```
Si <condición> Entonces
    <instrucciones>
Sino
    <instrucciones>
FinSi
```

Se evalúa la **condición** y se ejecutan las instrucciones que le siguen al **Entonces** si la condición es **verdadera**, o las instrucciones que le siguen al **Sino** si la condición es **falsa**. La condición debe ser una expresión lógica, que al ser evaluada retorna *Verdadero* o *Falso*.

# Algoritmos - PSeInt

---

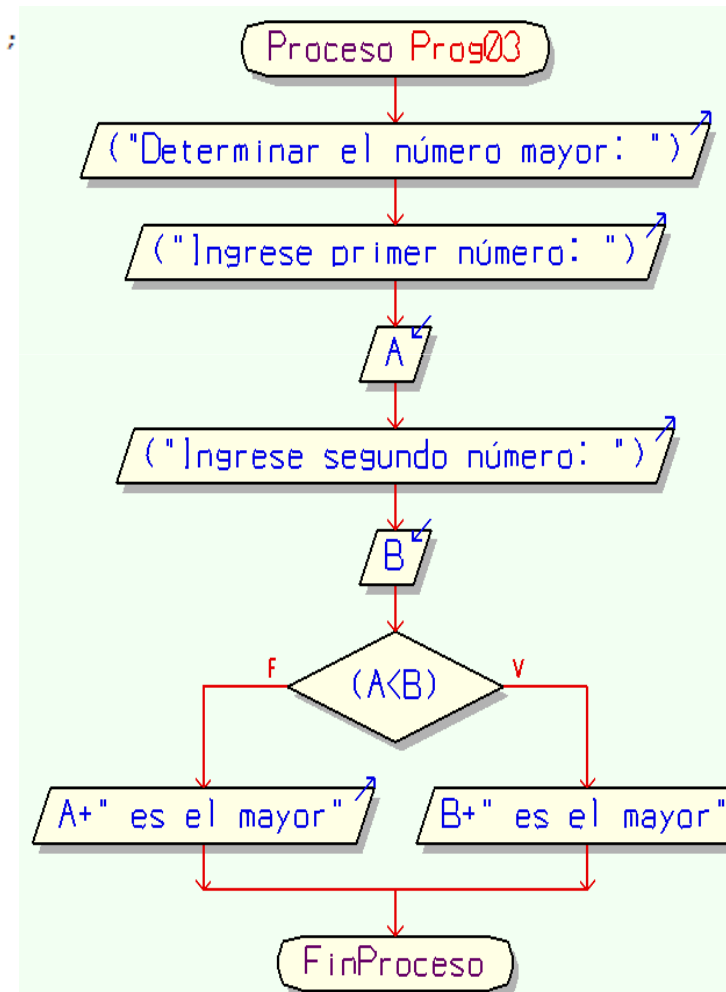
## Condicional Si-Entonces-Sino



# Algoritmos - PSeInt

```
1  Proceso Prog03
2      Escribir ("Determinar el número mayor: ");
3      Escribir ("Ingrese primer número: ");
4      Leer A
5      Escribir ("Ingrese segundo número: ");
6      Leer B
7      Si (A < B) Entonces
8          Escribir B+" es el mayor";
9      sino
10         Escribir A+" es el mayor";
11      FinSi
12  FinProceso
```

```
*** Ejecución Iniciada. ***
Determinar el número mayor:
Ingrese primer número:
> 40
Ingrese segundo número:
> 20
40 es el mayor
*** Ejecución Finalizada. ***
```





# Algoritmos - PSeInt

---

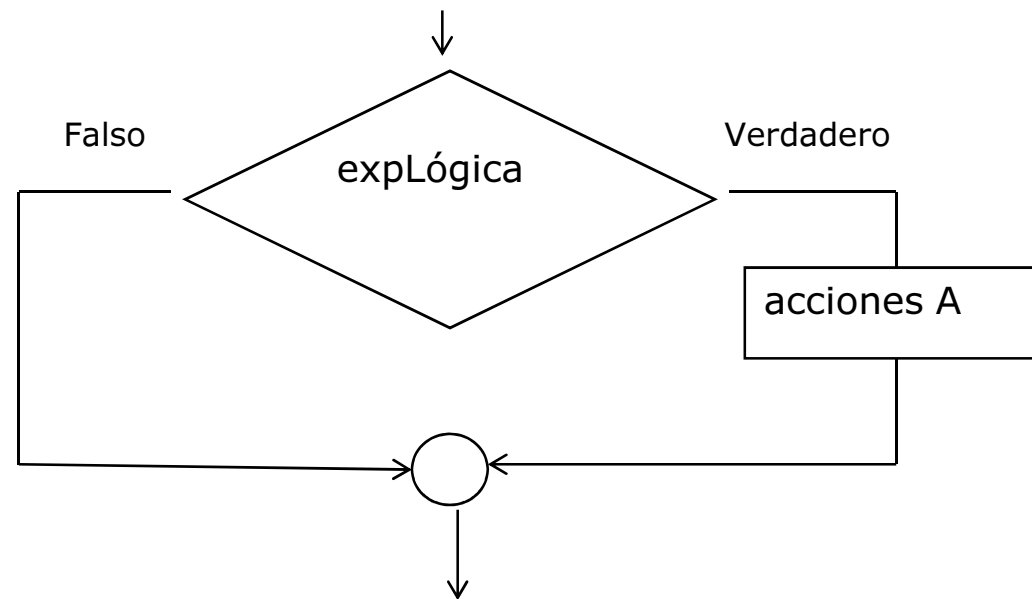
## Condicional Si-Entonces

La cláusula **Entonces** es obligatoria, pero la cláusula **Sino** puede no estar. En ese caso, si la condición es falsa no se ejecuta ninguna instrucción y la ejecución del programa continúa con la instrucción siguiente.

```
Si <condición> Entonces  
    <instrucciones>  
FinSi
```

# Algoritmos - PSeInt

## Condicional Si-Entonces



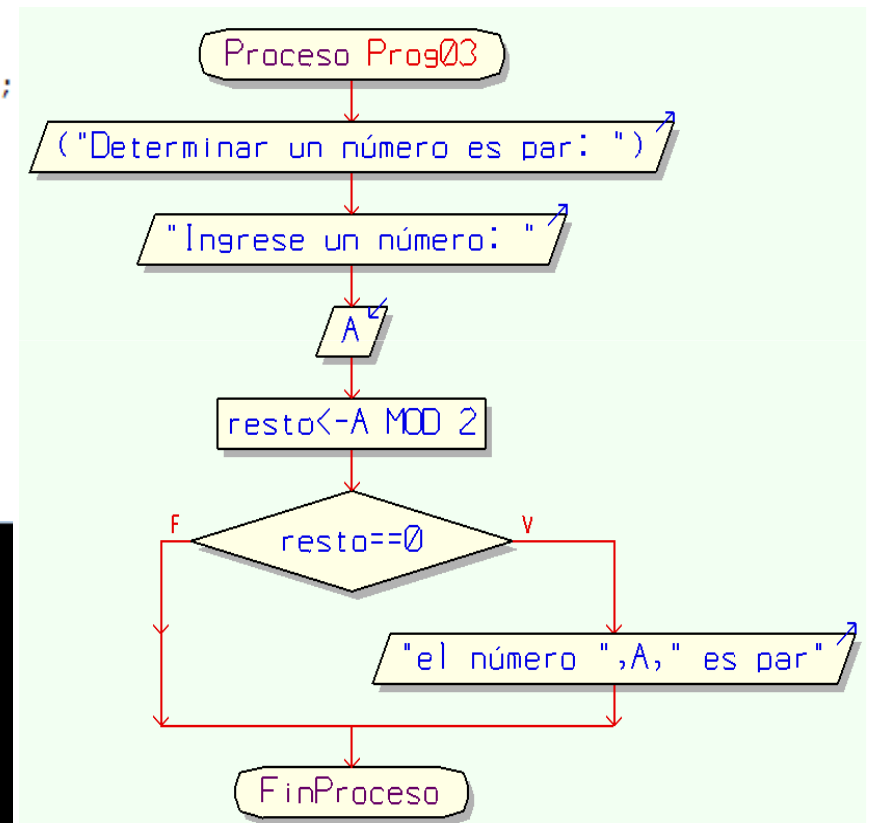


# Algoritmos - PSeInt

## Ejemplo Prog03

```
1 Proceso Prog03
2   Escribir ("Determinar un número es par: ");
3   Escribir "Ingrese un número: ";
4   Leer A;
5   resto = A MOD 2;
6   Si resto == 0 Entonces
7       Escribir "el número ",A," es par";
8   FinSi
9 FinProceso
```

```
<sin *** Ejecución Iniciada. ***
1 Determinar un número es par:
2 Ingrese un número:
3 > 1204
4 el número 1204 es par
5 *** Ejecución Finalizada. ***
6
7
```





# Algoritmos - PSeInt

---

## Selección Múltiple

La secuencia de instrucciones ejecutada por una instrucción **Segun** depende del valor de una variable numérica.

```
Segun <variable> Hacer
    <número1>: <instrucciones>
    <número2>,<número3>: <instrucciones>
    <...>
De Otro Modo: <instrucciones>
FinSegun
```

- Se evalúa el contenido de la variable y se ejecuta la secuencia de instrucciones asociada con dicho valor.
- Cada opción está formada por uno o más números separados por comas, dos puntos y una secuencia de instrucciones.



# Algoritmos - PSeInt

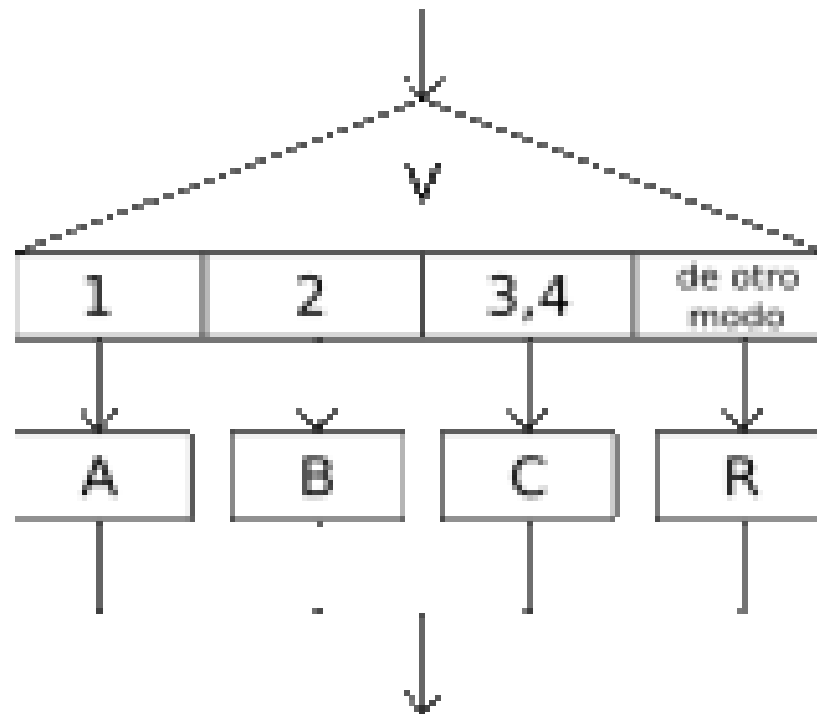
---

## Selección Múltiple

- Si una opción incluye varios números, la secuencia de instrucciones asociada se debe ejecutar cuando el valor de la variable es uno de esos números.
- Opcionalmente, se puede agregar una opción final, denominada **De Otro Modo**, cuya secuencia de instrucciones asociada se ejecutará sólo si el valor almacenado en la variable no coincide con ninguna de las opciones anteriores.
- Esta instrucción es recomendable para manejar Menu muy complejos.
- Esta opción se ocupa generalmente para capturar errores al ingresar una opción no disponible y así avisarle al usuario.

# Algoritmos - PSeInt

Selección Múltiple, diagrama de flujo:



# Algoritmos - PSeInt

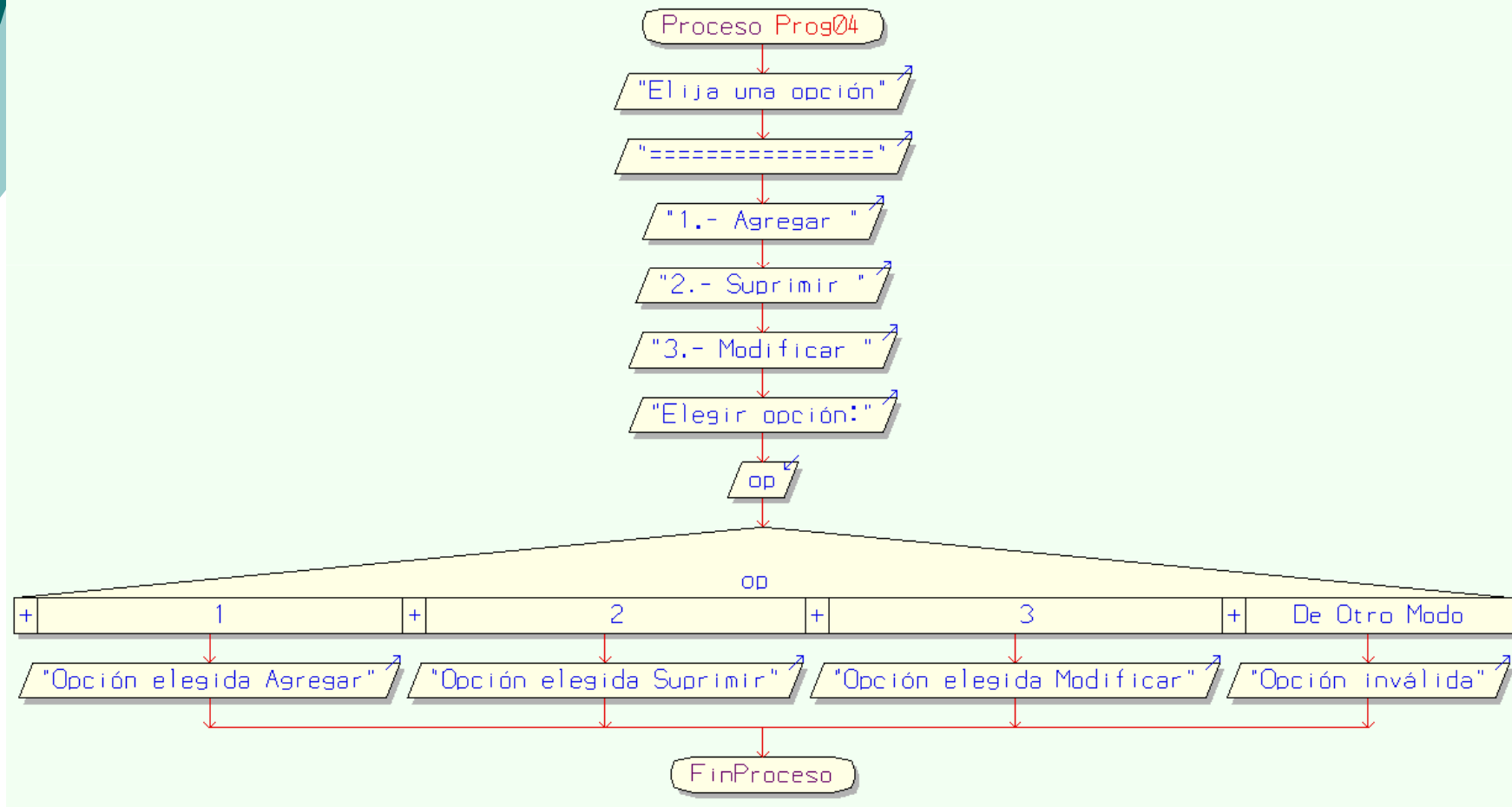
## Ejemplo Prog04

```
1  Proceso Prog04
2      Escribir "Elija una opción";
3      Escribir "=====
4      Escribir "1.- Agregar ";
5      Escribir "2.- Suprimir ";
6      Escribir "3.- Modificar ";
7      Escribir "Elegir opción:"
8      Leer op;
9
10     Segun op
11         1: Escribir "Opción elegida Agregar";
12         2: Escribir "Opción elegida Suprimir";
13         3: Escribir "Opción elegida Modificar";
14         De otro Modo: Escribir "Opción inválida";
15     FinSegun
16 FinProceso
```

```
*** Ejecución Iniciada. ***
Elija una opción
=====
1.- Agregar
2.- Suprimir
3.- Modificar
Elegir opción:
> 2
Opción elegida Suprimir
*** Ejecución Finalizada. ***
```

# Algoritmos - PSeInt

## Ejemplo Prog04

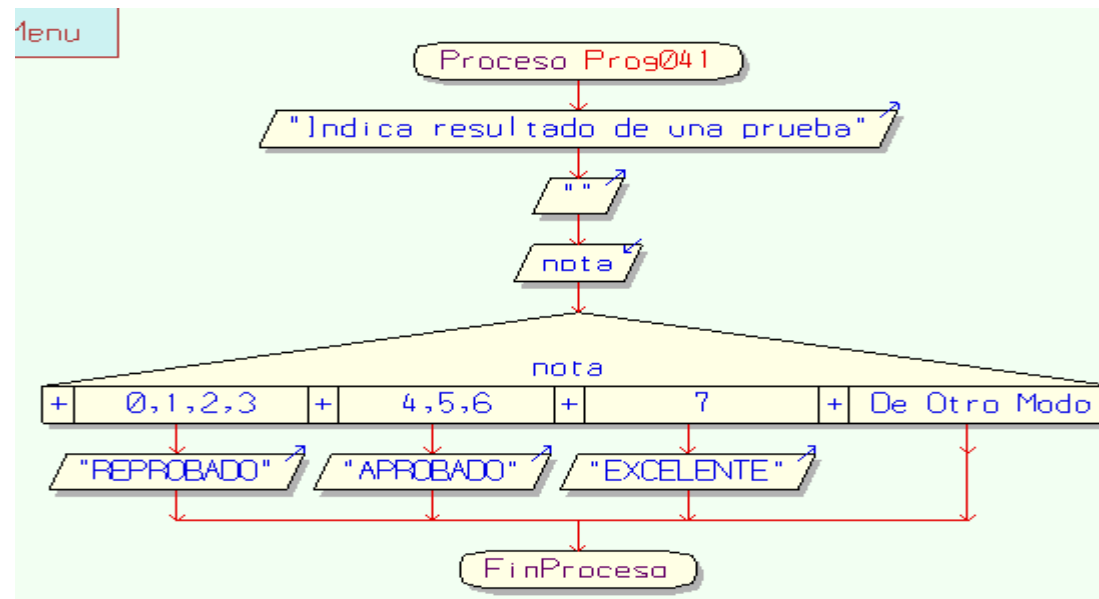


# Algoritmos - PSeInt

## Ejemplo Prog041

```
1  Proceso Prog041
2      Escribir "Indica resultado de una prueba";
3      Escribir "";
4      Leer nota;
5      Segun nota
6          0,1,2,3: Escribir "REPROBADO";
7          4,5,6: Escribir "APROBADO";
8          7: Escribir "EXCELENTE";
9      FinSegun
10 FinProceso
```

```
<si
*** Ejecución Iniciada. ***
Indica resultado de una prueba
1
2
3 > 5
4 APROBADO
5
6 *** Ejecución Finalizada. ***
```





# Algoritmos - PSeInt

---

## Mientras-Hacer:

```
Mientras <condición> Hacer  
    <instrucciones>  
FinMientras
```

- La instrucción **Mientras- Hacer** ejecuta una secuencia de instrucciones mientras una **condición sea verdadera**.
- Al ejecutarse, primero que nada la **condición se evalúa**.
- Si la condición resulta **verdadera**, se ejecuta una vez la secuencia de instrucciones que forman el cuerpo del ciclo.





# Algoritmos - PSeInt

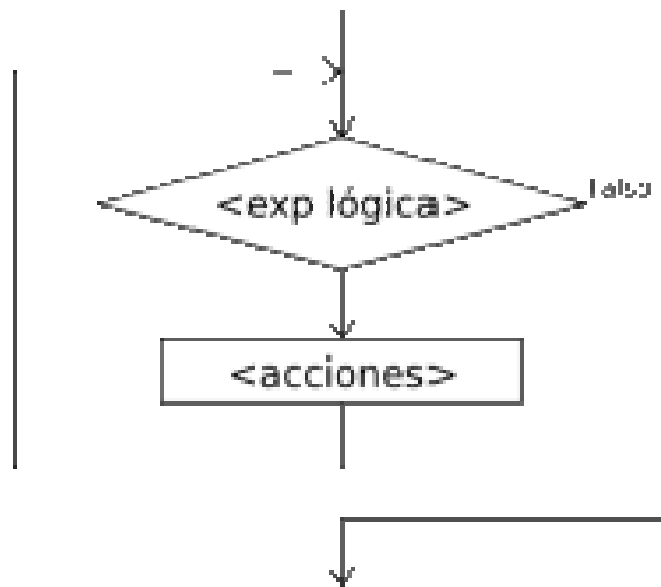
---

## Mientras-Hacer:

- Al **finalizar** la ejecución del cuerpo del ciclo **se vuelve a evaluar la condición** y, si es **verdadera**, la ejecución se **repite**.
- Estos pasos se repiten mientras la condición sea verdadera.
- Las instrucciones del cuerpo del ciclo pueden no ejecutarse **nunca**. Esto ocurre si al evaluar por primera vez la condición resulta ser **falsa**.
- Por otra parte, si la condición **siempre** es verdadera (nunca cambia), se produce un **ciclo infinito**.
- **Importante**: las instrucciones del cuerpo del ciclo deben contener alguna instrucción que modifique la o las variables involucradas en la condición, de modo que en algún momento sea falsa y finalice el ciclo.

# Algoritmos - PSeInt

Mientras-Hacer:



# Algoritmos - PSeInt

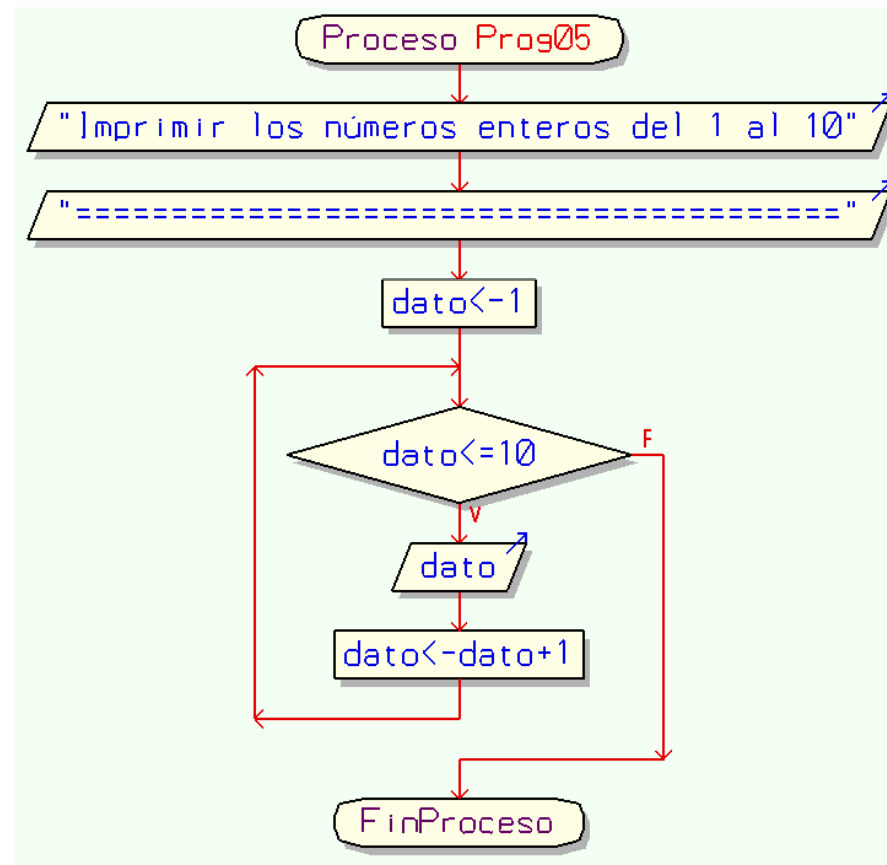
## Ejemplo Prog05

```
1  Proceso Prog05
2      Escribir "Imprimir los números enteros del 1 al 10";
3      Escribir "=====
4      dato = 1;
5      Mientras dato <= 10
6          Escribir dato;
7          dato <- dato + 1;
8      FinMientras
9  FinProceso
```

```
*** Ejecución Iniciada. ***
Imprimir los números enteros del 1 al 10
=====
1
2
3
4
5
6
7
8
9
10
*** Ejecución Finalizada. ***
```

# Algoritmos - PSeInt

## Ejemplo Prog05





# Algoritmos - PSeInt

---

## Repetir-Hasta

La instrucción **Repetir-Hasta** ejecuta una secuencia de instrucciones hasta que la condición sea verdadera.

Repetir

<instrucciones>

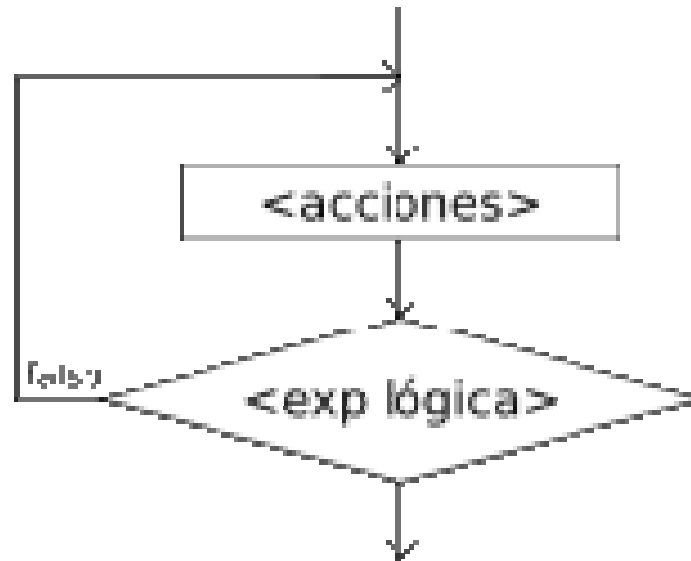
Hasta Que <condición>

- Al ejecutarse esta instrucción, la secuencia de instrucciones que forma el cuerpo del ciclo **se ejecuta una vez y luego se evalúa la condición**.
- Si la condición es **falsa**, el cuerpo del ciclo **se ejecuta nuevamente** y se vuelve a evaluar la condición.
- Esto se repite hasta que la condición sea verdadera.

# Algoritmos - PSeInt

## Repetir-Hasta

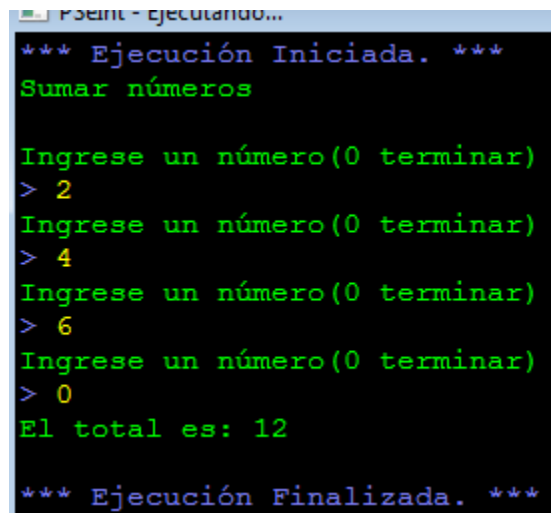
- **Important Hint:** las instrucciones del cuerpo del ciclo serán ejecutadas al menos una vez.
- Además, a fin de evitar ciclos **infinitos**, el cuerpo del ciclo debe contener alguna instrucción que **modifique** la o las variables involucradas en la condición de modo que en algún momento la condición sea **verdadera** y se finalice la ejecución del ciclo.



# Algoritmos - PSeInt

## Ejemplo Prog06

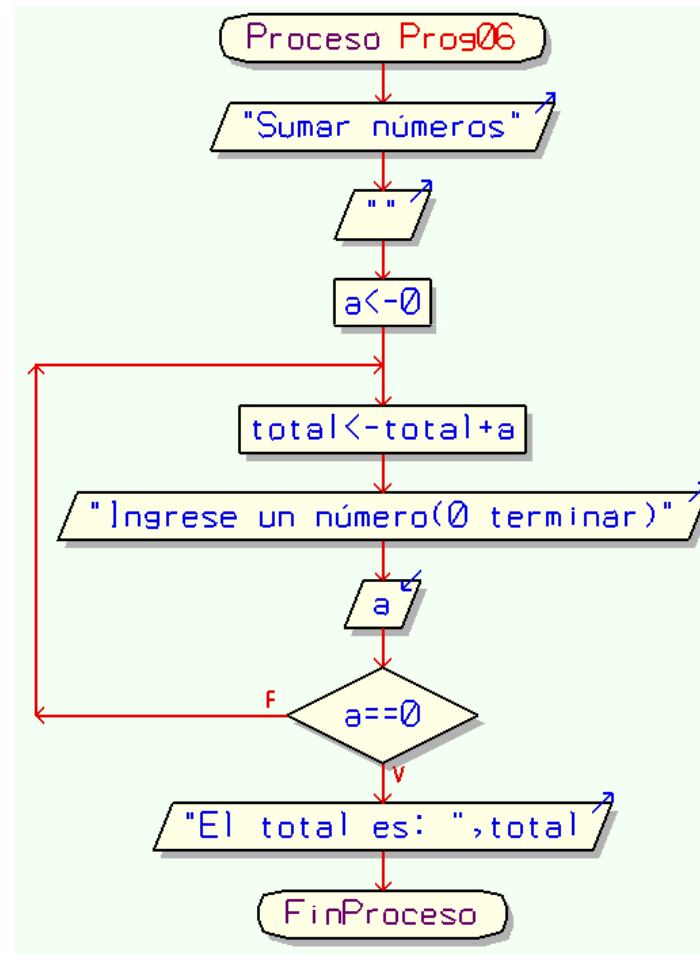
```
1  Proceso Prog06
2      Escribir "Sumar números";
3      Escribir "";
4      a=0;
5      Repetir
6          Total = Total +a;
7          Escribir "Ingrese un número(o terminar)"
8          Leer a;
9      Hasta Que a==0;
10     Escribir "El total es: ", total;
11 FinProceso
```



\*\*\* Ejecución Iniciada. \*\*\*  
Sumar números  
  
Ingrese un número(0 terminar)  
> 2  
Ingrese un número(0 terminar)  
> 4  
Ingrese un número(0 terminar)  
> 6  
Ingrese un número(0 terminar)  
> 0  
El total es: 12  
  
\*\*\* Ejecución Finalizada. \*\*\*

# Algoritmos - PSeInt

## Ejemplo Prog06







# Algoritmos - PSeInt

---

## Para

La instrucción **Para** ejecuta una secuencia de instrucciones un número determinado de veces.

Para <variable> ←<inicial> Hasta <final> ( Con Paso <paso> ) Hacer  
    <instrucciones>

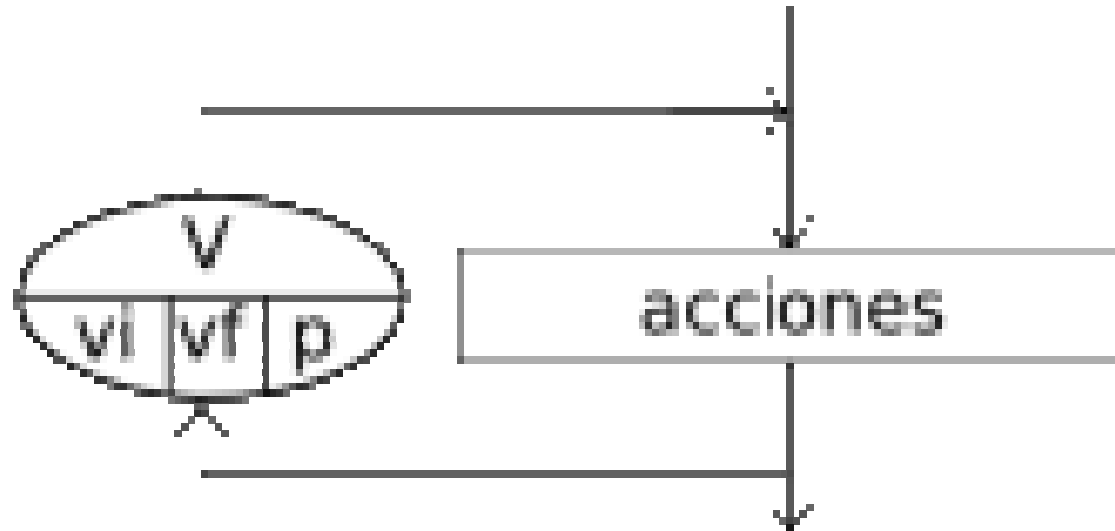
FinPara

- Al ejecutarse, la variable <variable> recibe el valor <inicial> y se ejecuta la secuencia de instrucciones que forma el cuerpo del ciclo.
- Luego la variable <variable> se **incrementa** en <paso> unidades y se evalúa si el valor almacenado en <variable> superó al valor <final>.

# Algoritmos - PSeInt

## Para

- Si esto es falso se repite hasta que <variable> supere a <final>.
- Si se omite la cláusula Con Paso <paso>, la variable <variable> se incrementará en 1 (por defecto).

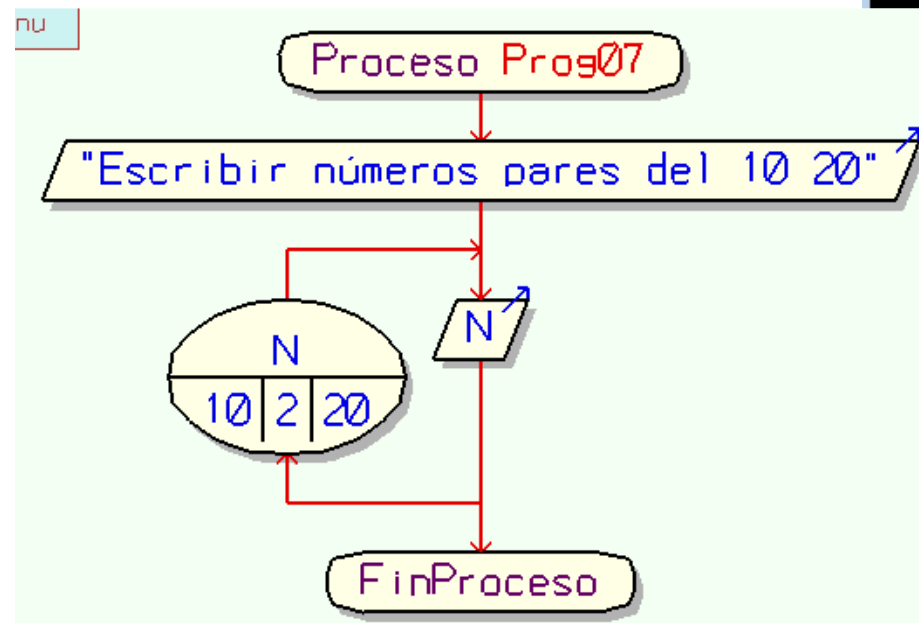


# Algoritmos - PSeInt

## Ejemplo Prog07

```
1  Proceso Prog07
2      Escribir "Escribir números pares del 10 20"
3      Para N=10 Hasta 20 Con Paso 2
4          Escribir N;
5      FinPara
6  FinProceso
```

```
*** Ejecución Iniciada. ***
Escribir números pares del 10 20
10
12
14
16
18
20
*** Ejecución Finalizada. ***
```





# Algoritmos - PSeInt

---

## Agenda:

- Estructura de un Algoritmo en pseudocódigo
- PSeInt-PS<sub>eudo</sub> I<sub>nterprete</sub>
- Símbolos gráficos
- Operadores y funciones que soporta PSeInt
- Instrucciones secuenciales
- Instrucciones de control