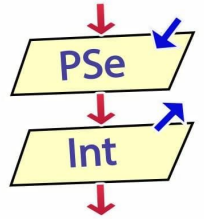
CODO A CODO INICIAL
Clase 5

Algoritmia 5

BA

Estructuras de control I
Estructuras condicionales



Les damos la bienvenida

Vamos a comenzar a grabar la clase

Clase 04

Algoritmia 4 - Entrada y salida de datos

- Entrada y salida de valores por pantalla.
- Ejemplos de instrucciones Leer y Escribir combinadas con operadores.

Clase 05

Algoritmia 5 - Estructuras condicionales I.

- Estructuras condicionales en la programación.
- Operadores relacionales.
- Si....Entonces y Si...Entonces...Sino

Clase 06

Algoritmia 6 - Estructuras condicionales II

- Estructuras no secuencuenciales en programación.
- Anidamiento de estructuras: uso de condicionales anidados.

Operadores relacionales

Los operadores relacionales en programación son herramientas clave para **comparar valores** y **evaluar condiciones**. Incluyen los símbolos “==”, “>”, “<”, “>=” , “<=” y “!=”, permitiendo establecer relaciones entre variables y valores. Estos operadores son fundamentales en **estructuras condicionales**, donde determinan la ejecución de bloques de código según si una condición dada es **verdadera** o **falsa**. Su uso es esencial para la toma de decisiones y el control del flujo en programas, brindando la capacidad de responder dinámicamente a diferentes situaciones.

¿Cuáles son los operadores relacionales?

Operador	Significado	Ejemplo
<	menor que	5 < 7 (Verdadero)
<=	menor o igual que	5 <= 4 (Falso)
>	mayor que	3 > 4 (Falso)
>=	mayor o igual que	13 >= 13 (Verdadero)
!=	distinto que	15 != 15 (Falso)
==	igual que	10 == 10 (Verdadero)

Operadores relacionales en PSeInt

En la pantalla de **PSeInt** los operadores relacionales se visualizan mediante los siguientes símbolos, cuando los escribimos de la manera anterior:

- ≠ (distinto que)
- < (menor que)
- ≤ (menor o igual que)
- > (mayor que)
- ≥ (mayor o igual que)

Operadores relacionales en PSeInt | Ejemplo

```
Algoritmo operadoresRelacionales

    Escribir 5 > 7 // mayor que
    Escribir 5 < 7 // menor que
    Escribir 7 == 7 // igual que
    Escribir 7 ≠ 7 // distinto que
    Escribir 8 ≥ 7 // mayor o igual que
    Escribir 8 ≤ 7 // menor o igual que

FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
FALSO
VERDADERO
VERDADERO
FALSO
VERDADERO
FALSO
*** Ejecución Finalizada. ***
```

Expresiones booleanas

Una expresión con operadores relacionales es siempre una **expresión booleana**. Es decir, su resultado solo puede arrojar como resultado un valor **Verdadero** o **Falso**.

El resultado de una expresión booleana es siempre, entonces, un **valor lógico**. Las expresiones aritméticas, en cambio, devuelven como resultado un valor numérico.

```
Algoritmo expBooleanas

    variable1 = 3 < 4
    Escribir variable1

    variable2 = 5 == 4
    Escribir variable2

FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
VERDADERO
FALSO
*** Ejecución Finalizada. ***
```

Expresiones booleanas | Ejemplos y aclaraciones

```
variable1 = 3 < 4
Escribir variable1
```

El resultado de esta operación relacional es **Falso**, un valor lógico.

```
variable1 = 27 + 10
Escribir variable1
```

El resultado de esta operación aritmética es 37, un valor numérico.

```
variable1 = 27 == 10
Escribir variable1
```

No debemos confundir el **operador relacional ==** con el **operador de asignación =**.

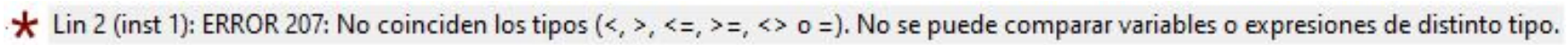
- **Operador de asignación (=):** Se utiliza para asignar un valor a una variable. El resultado de una operación se asigna a la variable de la izquierda.
- **Operador de igualdad (==):** Se utiliza para comparar dos valores. Devuelve un valor lógico (Verdadero o Falso) según si los valores son iguales o no.

Expresiones booleanas | Errores por tipos de dato

Los operandos usados con un operador relacional deben ser del mismo tipo. No es posible comparar datos de diferentes tipos. Estos ejemplos darán error:

```
variable1 = "27" == 10
Escribir variable1

variable1 = Verdadero == 10
Escribir variable1
```

Estos ejemplos darán error al ejecutar, ya que no coinciden los tipos de dato que intentamos comparar:



Comparación de cadenas

Para comparar caracteres se utiliza la tabla de códigos ASCII, donde a cada caracter le corresponde un número.

La “A” es el 65, la “B” es el 66. La “a” es el 97 y la “b” el 98. Las mayúsculas tienen valores menores a las minúsculas. Si el primer caracter de una cadena es igual, entonces compara el segundo, y así sucesivamente.

El orden de los caracteres en la tabla determina la respuesta.

Caracteres ASCII de control			Caracteres ASCII imprimibles						ASCII extendido (Página de código 437)							
00	NULL	(carácter nulo)	32	espacio	64	@	96	`	128	Ç	160	á	192	└	224	Ó
01	SOH	(inicio encabezado)	33	!	65	A	97	a	129	ü	161	í	193	┴	225	ß
02	STX	(inicio texto)	34	"	66	B	98	b	130	é	162	ó	194	┬	226	Ô
03	ETX	(fin de texto)	35	#	67	C	99	c	131	â	163	ú	195	├	227	Ò

Comparación de cadenas | Desafío

```
Algoritmo comparacionCadenas

    Escribir "hola" == "Hola"
    Escribir "hola" < "zapato"
    Escribir "hola" < "Zapato"
    Escribir "hola" < "Pez"

FinAlgoritmo
```

```
*** Ejecución Iniciada. ***
FALSO
VERDADERO
FALSO
FALSO
*** Ejecución Finalizada. ***
```

¿Te animás a deducir por qué se dan estos resultados?

Combinando operadores aritméticos y relacionales

¿Cómo se analiza la siguiente expresión?

```
Escribir 2 + 3 > 4
```

Los operadores aritméticos se resuelven **antes** que los relacionales, así que primero se evalúa ***2 + 3***, que es **5**, y finalmente se evalúa ***5 > 4***, que es **Verdadero**.

En la **jerarquía** de los operadores, los operadores aritméticos, con su propia jerarquía, tienen prioridad ante los operadores relacionales.

Estructuras condicionales

Las **estructuras condicionales** en programación son un tipo de estructuras de control que permiten **controlar el flujo de ejecución** de un programa de acuerdo con condiciones específicas.

Utilizando declaraciones como "**Si... Entonces**", el flujo del programa bifurca, es decir, se dirige por diferentes caminos según si una condición es **verdadera** o **falsa**. Estas estructuras son fundamentales para la ejecución de código condicional, permitiendo que un programa tome decisiones dinámicamente en respuesta a diferentes situaciones, mejorando así la flexibilidad y adaptabilidad del software.

Retomando conocimientos previos

Estos libros de **lectura no lineal** permiten que los lectores tomen decisiones que moldean la historia.

Cada elección afecta el curso de la trama. De manera similar, el uso de la lógica condicional influye en el flujo del programa.

En ambos casos, las elecciones afectan el resultado final, ya sea en la historia del libro o en el programa informático.

Condicional simple | Ejemplo I

Este programa determina si un estudiante ha aprobado un examen, según su calificación:

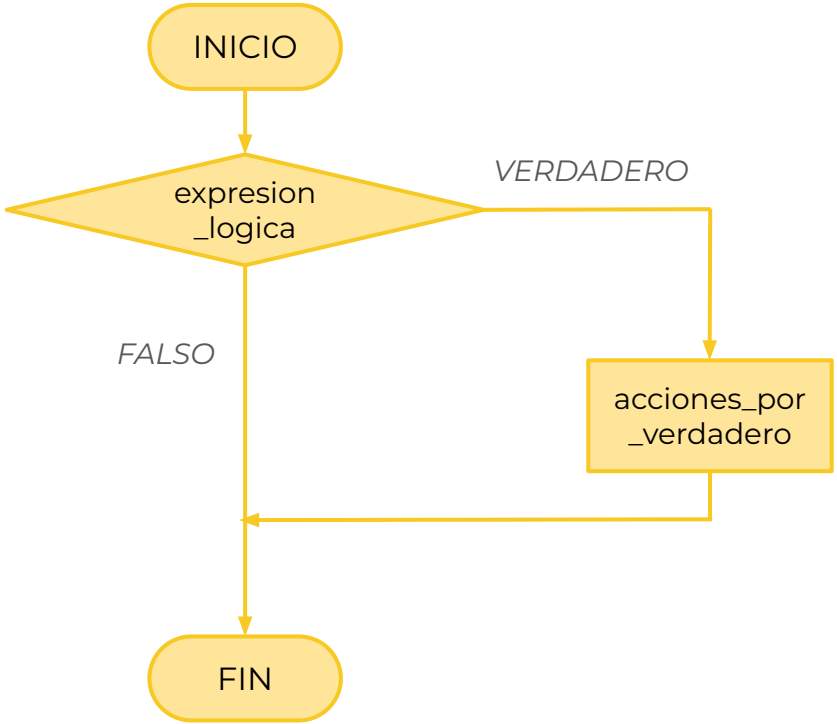

```
Algoritmo calificacionCondSimple

     Escribir "Ingrese la calificación del
estudiante"
     Leer calificacion
     Si calificacion >= 6 Entonces
          Escribir "El estudiante ha
aprobado"
     FinSi
     Escribir "Programa terminado"

FinAlgoritmo
```

Condicional simple

Este estructura ejecuta una serie de instrucciones ***sólo si*** la evaluación de determinada expresión es **VERDADERA**. Su sintaxis es:

```
Si expresion_logica Entonces
    acciones_por_verdadero
Fin Si
```

En *expresion_logica* se evalúa una condición. Si es verdadera se ejecutará el bloque *acciones_por_verdadero*.



Condicional simple | Ejemplo

El resultado (salida) de este programa es el siguiente:

```
*** Ejecución Iniciada. ***
Ingrese la calificación del estudiante
> 8
El estudiante ha aprobado
Programa terminado
*** Ejecución Finalizada. ***
```

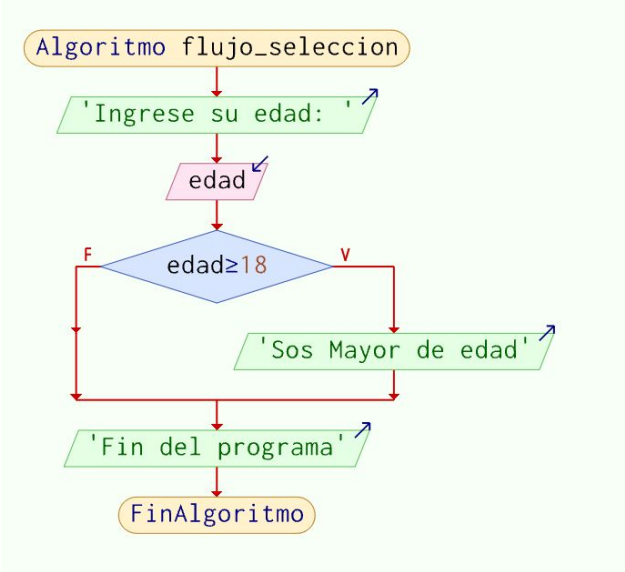
Salida cuando el usuario ingresa un valor mayor o igual a 6 (se cumple la condición)

```
*** Ejecución Iniciada. ***
Ingrese la calificación del estudiante
> 2
Programa terminado
*** Ejecución Finalizada. ***
```

Salida cuando el usuario ingresa un valor menor a 6 (no se cumple la condición)

Este programa sólo muestra una leyenda si el estudiante obtiene una nota mayor o igual a 6. A través del **condicional doble** podremos mejorarlo...

Condicional simple | Ejemplo II

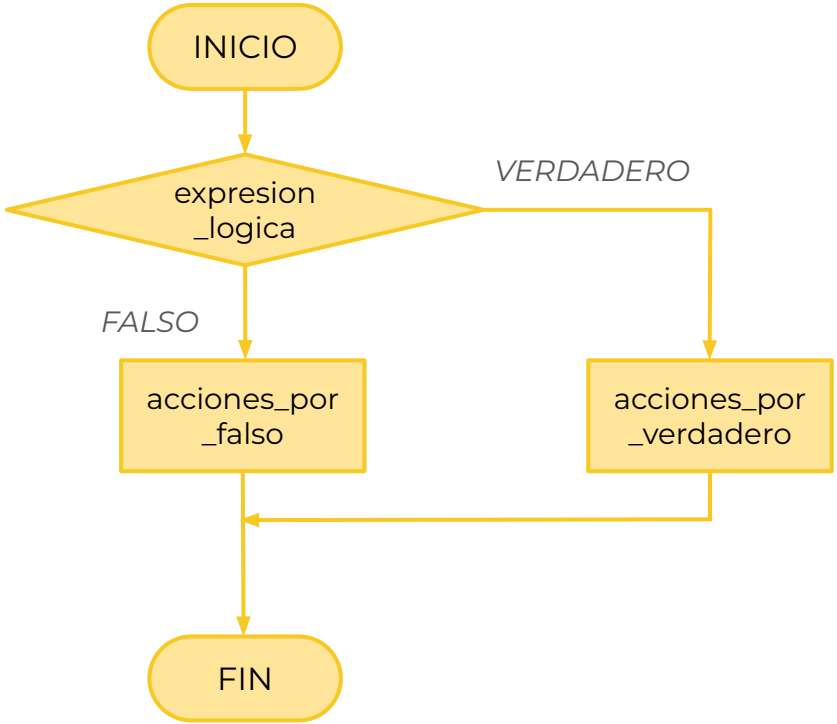



```
Algoritmo flujo_seleccion
    Escribir "Ingrese su edad: "
    Leer edad
    Si edad ≥ 18 Entonces
        Escribir "Sos Mayor de edad"
    Fin Si
    Escribir "Fin del programa"
FinAlgoritmo
```

Condicional doble

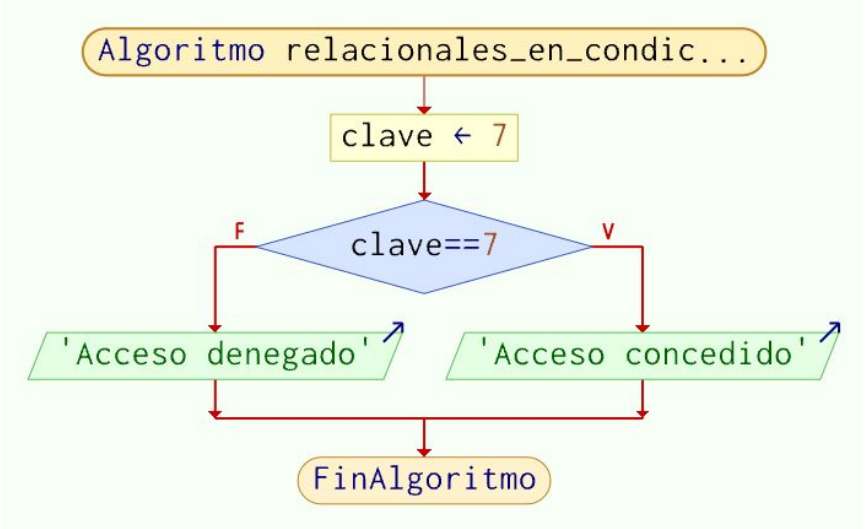
El **condicional doble** permite seleccionar la ejecución de uno de dos bloques de código. Uno se ejecuta ***sólo si*** la evaluación de la *expresion_logica* es **VERDADERA**. El otro se ejecuta ***sólo si*** la evaluación de la *expresion_logica* es **FALSA.**

```
Si expresion_logica Entonces
    acciones_por_verdadero
SiNo
    acciones_por_falso
Fin Si
```



Condicional doble | Ejemplo 1

```
Algoritmo relacionales_en_condicionales
    clave = 7;

    Si clave == 7 Entonces
        Escribir "Acceso concedido";
    SiNo
        Escribir "Acceso denegado";
    FinSi
FinAlgoritmo
```



Condicional doble | Ejemplo II


```
Algoritmo calificacionCondDoble

     Escribir "Ingrese la calificación del
estudiante"
     Leer calificacion
     Si calificacion >= 6 Entonces
          Escribir "El estudiante ha
aprobado"
     SiNo
          Escribir "El estudiante ha
reprobado"
     FinSi
     Escribir "Programa terminado"

FinAlgoritmo
```

Condicional doble | Ejemplo II

El resultado (salida) del programa anterior es el siguiente:

```
*** Ejecución Iniciada. ***
Ingrese la calificación del estudiante
> 9
El estudiante ha aprobado
Programa terminado
*** Ejecución Finalizada. ***
```

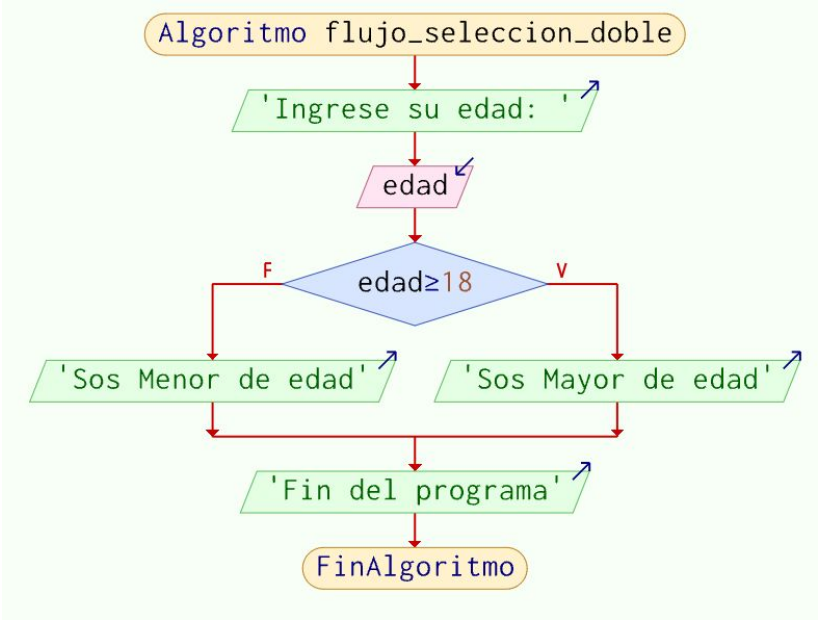
Salida cuando el usuario ingresa un valor mayor o igual a 6 (se cumple la condición)

```
*** Ejecución Iniciada. ***
Ingrese la calificación del estudiante
> 5
El estudiante ha reprobado
Programa terminado
*** Ejecución Finalizada. ***
```

Salida cuando el usuario ingresa un valor menor a 6 (no se cumple la condición)

Para pensar: ¿qué pasa si ingresamos un 11? ¿y un 0? Podemos limitar cuáles son las notas admitidas a través del **anidamiento de estructuras.**

Condicional doble | Ejemplo III




```
Algoritmo flujo_seleccion_doble
    Escribir "Ingrese su edad: "
    Leer edad
    Si edad ≥ 18 Entonces
        Escribir "Sos Mayor de edad"
    SiNo
        Escribir "Sos Menor de edad"
    Fin Si
    Escribir "Fin del programa"
FinAlgoritmo
```

Desafíos

Desafío 1: Extracción de dinero

Escribe un programa usando PSeInt y estructuras condicionales que resuelva la siguiente situación:

Un usuario necesita retirar un monto ***x*** *dinero de su cuenta bancaria.*

Teniendo en cuenta que tiene $10000 en su cuenta, solicitar al usuario el valor del monto a retirar, y a partir del saldo de su cuenta, informar si puede retirar o no el dinero que necesita.

Desafío 2: El problema del salario

A un trabajador le pagan según sus horas trabajadas y la tarifa tiene un valor por hora. Si la cantidad de horas trabajadas es mayor a 40, la tarifa por hora se incrementa en un 50% para las horas extras. Calcular el salario del trabajador dadas las horas trabajadas y la tarifa.

Escribe un programa usando PSeInt y estructuras condicionales que soliciten al operador el valor de la tarifa, la cantidad de horas trabajadas e imprima en la pantalla el salario a percibir..

Material extra

Artículos de interés

Material extra:

- Dominando las Estructuras Condicionales en PSeInt: Si Entonces, Sino y Sino Si | Victor Arana Flores

Videos:

- Condicionales en Pseint (Si Entonces | Sino) | Coders Free
- Estructuras condicionales | SI SINO | Introducción a los ALGORITMOS y la PROGRAMACIÓN | TodoCode

No te olvides de dar el presente

Recordá:

- Revisar la Cartelera de Novedades.
- Hacer tus consultas en el Foro.
- Realizar el Ejercicio de Repaso.

Todo en el Aula Virtual.

Muchas gracias por tu atención.

Nos vemos pronto.