**Universidad Tecnológica de Honduras**

**Ingeniería Computacional**

**INFORME**

**Alumnos**

Mariely Nicol Hiraeta Henrriquez

Gary Yamil García Mejía

Paola Michelt Rivera Sabillon

Henry Fernando Hernandez Lagos

**Asignatura**

Análisis y Diseño de Algoritmos

**Docente**

Jair Josue Hernandez Hernandez

**Fecha**

O4/3/2025

**Informe sobre la estructura de control Condicional Switch en PseInt**

**Introducción**

En la programación, las estructuras de control son fundamentales para la toma de decisiones y la ejecución de instrucciones en función de ciertas condiciones. Entre estas estructuras, el condicional "Según" en PseInt desempeña un papel clave cuando se requiere evaluar una variable y ejecutar distintas acciones según su valor. Este informe tiene como objetivo profundizar en el uso de "Según", su sintaxis, ejemplos prácticos, aplicaciones en la vida cotidiana, ventajas, desventajas y mejores prácticas.

**Objetivo General**

Comprender la estructura de control "Según" en PseInt, Analizar su sintaxis y aplicaciones en algoritmos, evaluar sus ventajas y desventajas frente a otras estructuras condiciónales.

**Objetivos Específicos**

* Explicar la sintaxis y uso de "Según" en PseInt mediante ejemplos prácticos.
* Relacionar su funcionamiento con situaciones de la vida cotidiana para una mejor comprensión.
* Identificar errores comunes en su implementación y proponer mejores prácticas para evitarlos.
* Comparar "Según" con otras estructuras condicionales como "Si - Sino" para determinar cuándo es más eficiente su uso.

**Definición del tema asignado**

La estructura de control "Según" en PseInt (equivalente a "Switch" en otros lenguajes de programación) permite seleccionar entre varias opciones posibles según el valor de una variable. Es útil cuando se necesita evaluar una variable contra distintos valores y ejecutar diferentes bloques de código según corresponda.

**Función y sintaxis en PseInt:**

Según variable Hacer

caso1:

// Código a ejecutar si la variable es caso1

caso2:

// Código a ejecutar si la variable es caso2

De Otro Modo:

// Código a ejecutar si no coincide con ningún caso

FinSegun

El "Según" evalúa el valor de la variable y ejecuta la instrucción correspondiente al caso coincidente. Si no hay coincidencia, se ejecuta la sentencia dentro de "De Otro Modo".

**Ejemplos prácticos**

**Ejemplo 1: Menú de opciones**

**Algoritmo MenuOpciones**

Definir opción Como Entero;

Escribir "Seleccione una opción:";

Escribir "1. Saludar";

Escribir "2. Despedirse";

Escribir "3. Decir '¿Cómo estás?'";

Leer opción;

Según opción Hacer;

1:

Escribir "Hola, ¿cómo estás?";

2:

Escribir "Adiós, que tengas un buen día";

3:

Escribir "¿Cómo estás?";

De Otro Modo;

Escribir "Opción no válida";

FinSegun

FinAlgoritmo

**Ejemplo 2: Días de la semana**

Definir dia Como Entero;

Escribir "Ingrese un número del 1 al 7";

Leer dia;

Segun dia Hacer

caso 1:

Escribir "Lunes";

caso 2:

Escribir "Martes";

caso 3:

Escribir "Miércoles";

caso 4:

Escribir "Jueves";

caso 5:

Escribir "Viernes";

caso 6:

Escribir "Sábado";

caso 7:

Escribir "Domingo";

De Otro Modo:

Escribir "Número inválido";

FinSegun

FinAlgoritmo

**Ejemplo 3**

Algoritmo SeleccionarBebida

Definir opcion Como Entero;

Escribir "Bienvenido a la máquina de bebidas";

Escribir "Seleccione una bebida:";

Escribir "1. Agua";

Escribir "2. Refresco";

Escribir "3. Jugo de naranja";

Escribir "4. Café";

Escribir "5. Té";

Leer opcion;

Segun opcion Hacer

1:

Escribir "Ha seleccionado Agua. ¡Disfrute su bebida!";

2:

Escribir "Ha seleccionado Refresco. ¡Disfrute su bebida!";

3:

Escribir "Ha seleccionado Jugo de naranja. ¡Disfrute su bebida!";

4:

Escribir "Ha seleccionado Café. ¡Disfrute su bebida!";

5:

Escribir "Ha seleccionado Té. ¡Disfrute su bebida!";

De Otro Modo:

Escribir "Opción no válida. Por favor, seleccione una opción del 1 al 5.";

FinSegun

FinAlgoritmo

**3. Comparación con la vida cotidiana**

El uso de Switch en programación es similar a muchas situaciones de la vida real donde elegimos entre varias opciones fijas.

**Ejemplos cotidianos**

**Ejemplo 1: Máquina expendedora**

• Imagina que introduces un número en una máquina expendedora.

• Si seleccionas "1", te da un refresco.

• Si seleccionas "2", te da agua.

• Si seleccionas un número no válido, muestra un mensaje de error.

**Ejemplo 2: Semáforo**

Según el color del semáforo:

Rojo → Detenerse

Verde → Avanzar

Amarillo → Precaución

**Ejemplo 3: Elección de menú en un restaurante**

• Si pides "1", te sirven pasta.

• Si pides "2", te sirven ensalada.

• Si pides un número incorrecto, el mesero te dice que no es válido.

**4. Ventajas y desventajas**

**Ventajas y desventajas que este presenta.**

**Ventajas:**

* Mejora la claridad y organización del código.
* Evita el uso excesivo de "Si - Sino" anidados.
* Más eficiente para evaluar una misma variable con múltiples valores.

**Desventajas:**

* Solo se puede usar con valores discretos (enteros, caracteres, cadenas predefinidas).
* No permite condiciones complejas.
* Menos flexible en comparación con "Si - Sino" cuando se necesitan rangos de valores.

**5. Cuidados a tener en cuenta**

Errores comunes:

* Olvidar "De Otro Modo" para manejar casos no contemplados.
* Usar "Según" con valores no discretos.
* No utilizar "FinSegun", lo que provoca errores de sintaxis.

**Mejores prácticas:**

* Usar "Según" cuando se tienen muchas opciones con comparaciones directas.
* Incluir "De Otro Modo" para evitar resultados inesperados.
* Escribir casos claros y evitar redundancia en la lógica

.

**Consulta de conceptos generales**

**Decisiones en algoritmos**

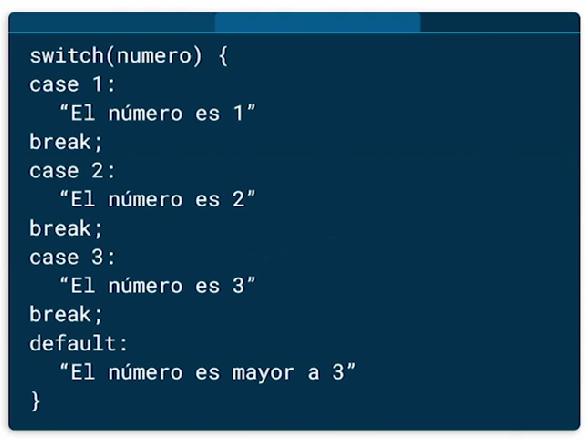
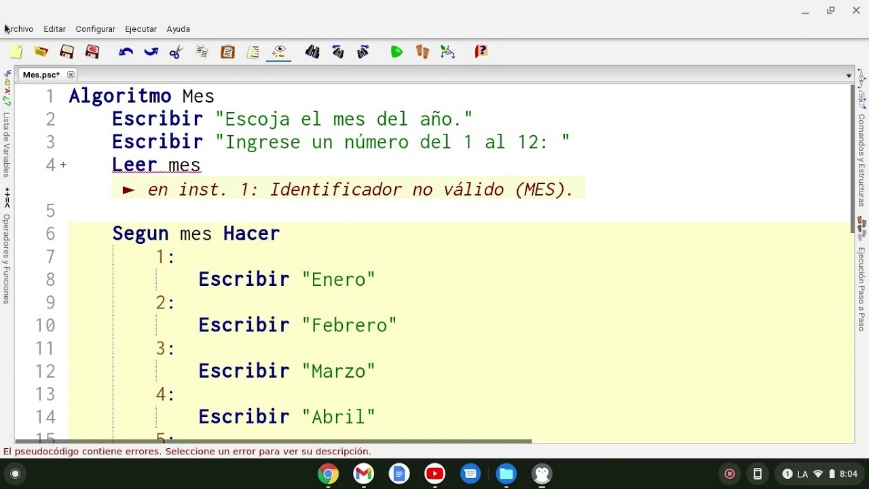
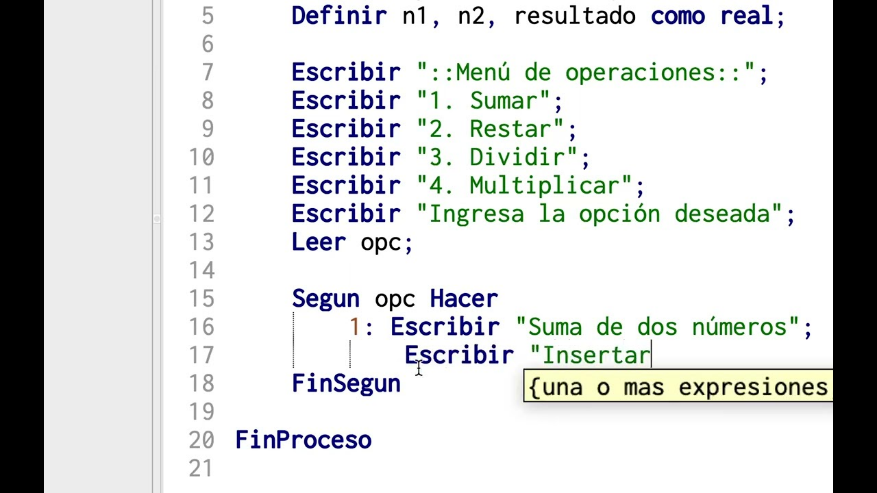
Las decisiones permiten que un programa tome diferentes rumbos según condiciones evaluadas. Se implementan con estructuras condicionales como "Si - Sino" y "Según".

**Bucles en algoritmos**

Los bucles permiten repetir acciones mientras se cumpla una condición. En PseInt existen "Mientras", "Repetir" y "Para", usados según la necesidad de iteración.

**Conclusiones**

La estructura "Según" en PseInt es útil para manejar decisiones con varias opciones de manera eficiente y ordenada. Su correcto uso permite mejorar la legibilidad del código y evitar errores en la programación.

****** Anexos**