

	Convención para Pseudocódigo	Convención para PHP
Tipos de Datos	<p>En el Pseudocódigo el tipo de dato se declara estáticamente, antes de utilizar la variable debemos declarar su tipo.</p> <p>Otros lenguajes que declaran los tipos explícitamente son JAVA, C#, C</p> <p>ENTERO</p> <p>BOOLEAN</p> <p>FLOAT (= DECIMAL. Se utiliza el punto como separador de decimales)</p> <p>STRING (= TEXTO. Para delimitar la cadena de caracteres usaremos comillas dobles: " ")</p>	<p>En PHP, el tipo de dato de una variable se establece dinámicamente cuando se ejecuta el programa, y depende del valor que se le asigne a la variable</p> <p>int</p> <p>boolean (valores TRUE o FALSE)</p> <p>float (se utiliza el punto como separador de decimales)</p> <p>string (para delimitar la cadena de caracteres usaremos comillas dobles: " ")</p> <p>(http://php.net/manual/es/language.types.php)</p>
Variable	<p>Identificador: utilizaremos notación <u>lowerCamelCase</u>, letras mayúsculas para inicializar las palabras, excepto la primera letra. No utilizaremos guión bajo (_).</p> <p>Los identificadores deben ser nombres significativos del problema que estamos resolviendo.</p> <p>ejemplos:</p> <p><u>ladoMenor</u></p> <p><u>ladoMayor</u></p> <p><u>perimetroRectangulo</u></p> <p>Declaración de variable significa: indicar el nombre y tipo de dato de la variable.</p> <p>Inicializar variable significa: asignar el primer valor a la variable. A partir de la inicialización decimos que la variable está definida.</p>	<p>Identificador (con la misma convención <u>lowerCamelCase</u> que el Pseudocódigo) al que antepondremos el signo \$</p> <p>ejemplos:</p> <p><u>\$ladoMenor</u></p> <p><u>\$ladoMayor</u></p> <p><u>\$perimetroRectangulo</u></p>
comentarios	<p>Comentarios de varias líneas:</p> <p>(*comentario varias lineas *)</p>	<p>Comentarios de varias líneas:</p> <p>/* comentario varias lineas */</p> <p>//comentario 1 linea</p>
Instrucción de Asignación	<p>nombreVariable Expresion</p> <p>Donde Expresion puede ser:</p> <ul style="list-style-type: none"> un valor una variable con valor combinación de operandos y operadores funciones que retornen valores (tema de unidad 4) <p>Ejemplo:</p> <p><u>ladoMenor</u> 20</p>	<p>\$nombreVariable = Expresion ;</p> <p>Donde Expresion puede ser:</p> <ul style="list-style-type: none"> un valor una variable con valor combinación de operandos y operadores funciones que retornen valores (tema de unidad 4) <p>Ejemplo:</p> <p><u>\$ladoMenor</u> = 20 ;</p>

		(observación: en php las instrucciones terminan con ;)
Instrucción de Entrada	<p>Ejemplo:</p> <p>LEER (ladoMenor)</p>	<p>Ejemplo:</p> <p>\$ladoMenor = trim(fgets(STDIN)) ;</p> <p>(observación: en php las instrucciones terminan con ;)</p>
Instrucción de Salida	<p>Ejemplo:</p> <p>ESCRIBIR(" El perímetro del rectángulo es " , perimetroRectangulo)</p> <p>(observación: Para la operación de concatenación de string se utiliza coma ",")</p>	<p>Ejemplo:</p> <p>echo " El perímetro del rectángulos es " . perimetroRectangulo;</p> <p>(observación: en php las instrucciones terminan con ; Para la operación de concatenación de string se utiliza punto ".")</p>
Separación de instrucciones	Una instrucción por renglón, indentando adecuadamente las instrucciones.	Utilizando punto y coma ";" (Por prolijidad siempre conviene 1 instrucción por renglón, indentando adecuadamente las instrucciones)
Estructura de control Secuencial (se ejecuta una instrucción debajo de la otra)	<p>PROGRAMA NombrePrograma (*descripción del programa ¿Qué problema resuelve?*) Declaración de variables: Tipo nombresVariables instruccion1 ... instruccionN FIN PROGRAMA</p> <p>Ejemplo:</p>	<pre><?php /*nombrePrograma*/ /* Declaración de variables: Tipo nombresVariables */ instruccion1; ... instruccionN;</pre> <p>?></p> <p>Obs: En lenguajes como PHP, donde el tipo de las variables es dinámico, no hay declaración de tipos. Por eso comentaremos la declaración o no la incluiremos. Podemos decir que es el tipo que esperamos tenga una variable por los valores que serán asignados a dichas variables.</p>

**Operadores
para
expresiones**

	Pseudocódigo	PHP
Resta, suma, multiplicación, división	- + * /	- + * /
Módulo o Resto de división	MOD	%
Concatenación de cadenas de caracteres	, (coma)	. (punto)
Comparación igual, distinto	= <>	== <>
Comparación mayor , mayor igual	> >=	> >=
Comparación menor , menor igual	< <=	< <=
Lógicos / Booleanos	AND OR NOT	&& !
Raíz cuadrada de E \sqrt{E}	raiz(E)	sqrt(E)
Operador Condicional / Ternario	SI E1 ENTONCESE2 SINO E3	E1? E2 : E3

Concepto	Pseudocódigo	php
Valor absoluto: $ a $	abs(a)	abs(a)
Potencia: a^b	potencia(a,b)	pow(a,b)

Uso del operador ternario:

PROGRAMA MayoriaEdad

(*determinar si una persona es mayor de edad*)

Boolean esMayor, Entero edad, String nombre,mensaje

ESCRIBIR("Ingrese su nombre")

LEER(nombre)

ESCRIBIR("Ingrese su edad")

LEER(edad)

esMayor <- (edad >= 18)

mensaje <- SI esMayor ENTONCES nombre," es mayor de edad" SINO nombre," es menor de edad "

ESCRIBIR (mensaje)

FIN PROGRAMA

	<p>Inicio del PROGRAMA</p> <p>PROGRAMA MayoríaEdad</p> <p>(*determinar si una persona es mayor de edad*)</p> <p>Boolean esMayor, Entero edad, String nombre,mensaje → Declaración de variables</p> <p> ESCRIBIR("Ingrese su nombre") LEER(nombre) ESCRIBIR("Ingrese su edad") LEER(edad) esMayor <- (edad >= 18) mensaje <- Si esMayor ENTONCES nombre, " es mayor de edad" SINO nombre, " es menor de edad " ESCRIBIR (mensaje) </p> <p>FIN PROGRAMA → Fin del PROGRAMA</p> <p>Bloque de instrucciones del programa</p> <p>Comentario entre simbolos (* y *) indicando ¿Qué problema resuelve el programa?</p>
Modularización : Funciones	<pre> (** *Descripcion del módulo sin retorno (¿qué hace?) *) MODULO nombre(tipo1 pf1, tipo2 pf2,...,tipoN pfN) RETORNO ∅ instruccion1 ... instruccionN FIN MODULO (** * Descripcion del módulo con retorno (¿qué hace?) *) MODULO nombre(tipo1 pf1, tipo2 pf2,...,tipoN pfN) RETORNO tipo instruccion1 ... instruccionN RETORNO(E) FIN MODULO (obs.: tipo puede ser boolean, integer, float, string, etc.) </pre> <pre> /** *Descripcion del modulo sin retorno (¿qué hace?) * @param tipo \$pf1 * @param tipo2 \$pf2 * ... * * @param tipo \$pfN */ function nombre(\$pf1, \$pf2,...,\$pfN){ instruccion1; ... instruccionN; } /** *Descripcion del modulo con retorno (¿qué hace?) * @param tipo1 \$pf1 * @param tipo2 \$pf2 * ... * @param tipoN \$pfN * @return tipo */ function nombre(\$pf1, \$pf2,...,\$pfN){ instruccion1; ... instruccionN; return E; } </pre>
Alternativa: Instrucción IF	<pre> SI (condición) ENTONCES instruccion1 instruccionN FIN SI SI (condición) ENTONCES instruccionA1 instruccionAN SINO instruccionB1 </pre> <pre> if (condición){ Instrucción1; InstrucciónN; } if (condición){ InstrucciónA1; InstrucciónAN; } else{ InstrucciónB1; } </pre>

	instruccionBN FIN SI SI (condición) ENTONCES instruccionA1 instruccionAN OTRO-SI (condiciónB) ENTONCES instruccionB1 instruccionBN OTRO-SI (condiciónM) ENTONCES instruccionM1 instruccionMN FIN SI SI (condición) ENTONCES instruccionA1 instruccionAN OTRO-SI (condiciónB) ENTONCES instruccionB1 instruccionBN OTRO-SI (condiciónM) ENTONCES instruccionM1 instruccionMN SINO instruccionN1 instruccionNN FIN SI	InstrucciónBN; } if (condiciónA){ InstrucciónA1; InstrucciónAN; } elseif (condiciónB){ InstrucciónB1; InstrucciónBN; } elseif (condiciónM){ InstrucciónM1; InstrucciónMN; } if (condiciónA){ InstrucciónA1; InstrucciónAN; } elseif (condiciónB) { InstrucciónB1; InstrucciónBN; } elseif (condiciónM) { InstrucciónM1; InstrucciónMN; } else{ InstrucciónN1; InstrucciónNN; }
--	---	--

Alternativa: Instrucción IF anidada	Las siguientes instrucciones son a modo de ejemplo, hay tanta combinaciones como problemas a resolver SI (condicion) ENTONCES SI (condición) ENTONCES instruccion_a1 instruccion_aN FIN SI instruccionN FIN SI SI (condicion) ENTONCES SI (condición) ENTONCES instruccion_a1 instruccion_aN FIN SI instruccionN SINO instruccionB1	Las siguientes instrucciones son a modo de ejemplo, hay tanta combinaciones como problemas a resolver if (condición){ if (condición){ Instrucción_a1; Instrucción_aN; } InstrucciónN; } if (condición){ if (condición){ Instrucción_a1; Instrucción_aN; } InstrucciónN; } else { InstrucciónB1;
--	--	---

	instruccionBN FIN SI SI (condicion) ENTONCES SI (condición) ENTONCES instruccion_a1 instruccion_aN FIN SI instruccionN SINO instruccionB1 SI (condición) ENTONCES instruccion_b1 instruccion_bN SINO instruccion_c1 instruccion_cN FIN SI FIN SI SI (condicion1) ENTONCES SI (condicion_a) ENTONCES instruccion_a1 instruccion_aN FIN SI instruccionN OTRO-SI (condicion2) ENTONCES instruccion_c1 SI (condición) ENTONCES instruccion_b1 instruccion_bN SINO instruccion_c11 instruccion_c1N FIN SI instruccion_c3 SINO instruccionB1 instruccionBN FIN SI	InstrucciónBN; } if (condición){ if (condición){ Instrucción_a1; Instrucción_aN; } InstrucciónN; } else { instrucciónB1; if (condición){ Instrucción_a1; Instrucción_aN; } else { Instrucción_b1; Instrucción_bN; } } } if (condición){ if (condición){ Instrucción_a1; Instrucción_aN; } InstrucciónN; } elseif(condicion2) { instrucción_c1; if (condición){ Instrucción_b1; Instrucción_bN; } else { Instrucción_c11; Instrucción_c1N; } instrucción_c3; } else { InstrucciónB1; InstrucciónBN; }
--	---	---