Sintaxis / convenciones de Sintaxis

	ciones de Sintaxis  Convención para  Pseudocódigo	Convención para PHP
	rseudocodigo	r IIIr
Tipos de Datos	En el Pseudocódigo el tipo de dato se declara estáticamente, antes de utilizar la variable debemos declarar su tipo. Otros lenguajes que declaran los tipos explícitamente son JAVA, C#, C	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
	ENTERO BOOLEAN FLOAT (= DECIMAL. Se utiliza el punto como separador de decimales) STRING (= TEXTO. Para delimitar la cadena de caracteres usaremos comillas dobles: " ")	Integer Boolean (valores TRUE o FALSE) Float (se utiliza el punto como separador de decimales) String (para delimitar la cadena de caracteres usaremos comillas dobles: "")
		(http://php.net/manual/es/language.types .php)
Variable	Identificador: utilizaremos notación  lowerCamelCase, letras mayúsculas para inicializar las palabras, excepto la primera letra. No utilizaremos guión bajo (_). Los identificadores deben ser nombres significativos del problema que estamos resolviendo.	Identificador (con la misma convención lowerCamelCase que el Pseudocodigo) al que antepondremos el signo \$
	ejemplos: ladoMenor ladoMayor perimetroRectangulo	ejemplos: \$ladoMenor \$ladoMayor \$perimetroRectangulo
	Declaración de variable significa: indicar el nombre y tipo de dato de la variable.  Inicializar variable significa: asignar el primer valor a la variable. A partir de la inicialización decimos que la variable está definida.	
comentarios	Comentarios de varias líneas:  (*comentario varias lineas *)	Comentarios de varias líneas: /* comentario varias lineas */
		//comentario 1 linea
Instrucción de Asignación	nombreVariable <b>Expresion</b>	\$nombreVariable = Expresion;
	Donde Expresion puede ser:  un valor  una variable con valor  combinación de operandos y operadores  funciones que retornen valores (tema de unidad 4)  Ejemplo: ladoMenor 20	Donde Expresion puede ser:  un valor  una variable con valor  combinación de operandos y operadores  funciones que retornen valores (tema de unidad 4)  Ejemplo: \$ladoMenor = 20;

Instrucción de   Ejemplo:   Ejemplo:     Ejemplo:						
Ejemplo:				(observación: en php las instrucciones		
Entrada  LEER (ladoMenor)  SladoMenor = trim(fgets(STDIN));  (observación: en php las instrucciones terminan con;)  Escribilita  ESCRIBIR(" El perímetro del rectángulo es ", perimetroRectangulo)  (observación: perimetroRectangulo)  (observación: en php las instrucciones echo " El perímetro del rectángulos es " , perimetroRectangulo;  (observación: en php las instrucciones terminan con; para la operación de concatenación de string se utiliza pounto ".")  Una instrucción por renglón, identando adecuadamente las instrucciones.  Una instrucción por renglón, identando adecuadamente las instrucciones.  Una instrucción por renglón del algoritmo" ("descripción del algoritmo") ("mistrucción)  Instrucción  Poedrarción de valiables: Tipo nombresVariables linstruccions)  PIN ALGORITMO  Operadores para expresiones  Resta, suma, multiplicación, división	1	et l .				
LEER (ladoMenor)  LEER (ladoMenor)  SladoMenor = trim(fgets(STDIN));  (observación: en php las instrucciones terminan con;)  Ejemplo:  ESCRIBIR(" El perímetro del rectángulo es ", perimetroRectangulo)  (observación: Para la operación de concatenación de string se utiliza coma ",")  Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucción por renglón, identando adecuadamente las instrucciones.  Secuencial (se ejecuta una instrucción de valualies: Tipo nombres/variables linstruccions)  Instrucción de bajo de la otra)  Operadores para expresiones  Poperadores para expresiones  Resta, suma, multiplicación, división + * * / * / * / * + * / * / * / * / * /		Ejemplo:		Ejemplo:		
terminan con;	Entrada	LEER (ladoMenor)		\$ladoMenor = trim(fge	ts(STDIN));	
Ejemplo:   Ejemplo:   Ejemplo:   Ejemplo:   Ejemplo:   EsCRIBIR(" El perímetro del rectángulo es ", perimetroRectangulo)   (observación:   para la operación de concatenación de string se utiliza coma ",")   (observación: en php las instrucciones terminan con;   Para la operación de concatenación de string se utiliza punto ".")   Utilizando punto y coma ","   (Por prolijidad siempre conviene 1 instrucción por rengión, identando adecuadamente las instrucciones.   (Por prolijidad siempre conviene 1 instrucción por rengión, identando adecuadamente las instrucciones)   (Pescripción del algoritmo")   (Pescripción del				1	as instrucciones	
ESCRIBIR(" El perímetro del rectángulo es ", perimetroRectangulo;  (observación: Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucción por renglón, identando adecuadamente las instrucciones.  Una instrucción por renglón, identando adecuadamente las instrucciones.  Estructura de control (Prescripción del algoritmo") (pedescripción del algoritmo) (prescripción del algoritmo) (prescr	Instrucción de	Fiemplo:				
ESCRIBIR(" El perímetro del rectángulo es ", perimetroRectangulo)   (observación: Para la operación de concatenación de string se utiliza coma ",")   Separación de instrucciones decuadamente las instrucciones.   Una instrucción por renglón, identando adecuadamente las instrucciones.   Una instrucción por renglón, identando adecuadamente las instrucciones.   Utilizando punto y coma ","		Ejempio.		Ljempio.		
perimetroRectangulo)  (observación: Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  Estructura de control Secuencial (se ejecuta una instrucción debajo de la otra)  Instrucción  debajo de la otra)  Operadores para expresiones  Operadores para expresiones  Pasa la operación de concatenación de string se utiliza punto ".")  ALGORITMO nombreAlgoritmo (l'descripción del algoritmo*)   Instrucción   I	<b>J</b>	ESCRIBIR(" El perímetro del rectángulo es "	_	echo " El perímetro del rectángulos es ".		
(observación: Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  Logicario de control decuadamente las instrucciones.  ALGORITMO nombreAlgoritmo [1/descripción del algoritmo*] Declaración de variables: Tipo nombresVariables [Instrucción por renglón, identando adecuadamente las instrucciones)  ALGORITMO nombreAlgoritmo [1/descripción del algoritmo*] Declaración de variables: Tipo nombresVariables [Instrucción debajo de la otra)  PRIN ALGORITMO  Operadores para expresiones  Pesudocódigo PHP, donde el tipo de las variables es dinámico, no hay declaración de tipos. Por eso comentaremos la declaración o no la incluíremos. Podemos decir que es el tipo que esperanis reing una variable por los valores que serán asignados a dichas variables.  Pseudocódigo PHP  Resta, suma, multiplicación, división - + * / - + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * / + + * /			,	·		
Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  Una instrucción por renglón, identando adecuadamente las instrucciones.  Utilizando punto y coma ";" (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)  Estructura de control Secuencial (se ejecuta una instrucción de valiables: Tipo nombres/variables   instrucción   instrucción de valiables: Tipo nombres/variables   instruccion1; instrucción   instrucción   instruccionN; instrucc		<b>8</b> 3, 3,				
Para la operación de concatenación de string se utiliza coma ",")  Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  Una instrucción por renglón, identando adecuadamente las instrucciones.  Utilizando punto y coma ";" (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)  Estructura de control Secuencial (se ejecuta una instrucción de valiables: Tipo nombres/variables   instrucción   instrucción de valiables: Tipo nombres/variables   instruccion1; instrucción   instrucción   instruccionN; instrucc		(observación:		(observación: en php la	(observación: en php las instrucciones	
utiliza coma ",")  Separación de string se utiliza punto ".")  Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  ### Utilizando punto y coma ";" (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)  #### ALGORITMO nombreAlgoritmo   (*descripción del algoritmo*)   Declaración de lagoritmo*)   Declaración de variables: Tipo nombresVariables   instruccion1; instruccion1; instruccion1; instruccion8;   Tipo nombresVariables "/ instruccion1; instruccion1; instruccion1; instruccion8;   Tipo nombresVariables "/ Obs: En lenguajes como PHP, donde el tipo de las variables es dinámico, no hay declaración de to la incluierenos, Podemos decir que es el tipo que esperamos tenga una variable por los valores que serán asignados ad dichas variables.  #### Operadores para expresiones  #### Operadores para expresiones    Resta, suma, multiplicación, división		l '	g se			
Separación de instrucciones  Una instrucción por renglón, identando adecuadamente las instrucciones.  Utilizando punto y coma ";" (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)  **RAGORITMO nombreAlgoritmo**   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)  **AGORITMO nombreAlgoritmo**   (Por prolijidad siempre conviene 1 instrucciones)  **Peblacración de variables: Tipo nombresVariables instruccions;   instruccion1;   instruccion1;   instruccionN;   instruccio		<u> </u>			1	
Una instrucción por renglón, identando adecuadamente las instrucciones.  Estructura de control Secuencial (se ejecuta una instrucción de la devalables: Tipo nombresVariables   Instrucción		, ,		· · · · · · · · · · · · · · · · · · ·		
adecuadamente las instrucciones.    (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones)   (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones     (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones     (Por prolijidad siempre conviene 1 instrucción por renglón, identando adecuadamente las instrucciones     (Por prolijidad siempre convienes     (Por prolijidad siempre convienes     (Por prolijidad siempre convienes)   (Por prolijidad siempre convienes)   (Por prolijidad siempre convienes)   (Por prolijidad sietus)     (Por prolijicanis)     (Por proliji	Separación de	Una instrucción por renglón, identando				
Estructura de control Secuencial (se ejecuta una instrucción de Valiables: Tipo nombres/variables / instruccion)  Secuencial (se ejecuta una instrucción debajo de la dortra)  Operadores para expresiones  Pseudocódigo  Resta, suma, multiplicación, división  Concatenación de cadenas de caracteres (coma)  Comparación igual, distinto  Comparación mayor , mayor igual  Comparación mayor , mayor igual  Raiz cuadrada de E √(E)  Operador Condicional / Ternario  Instrucción por renglón, identando adecuadamente las instruccions)  **Php**  **Php**  **Pectaración de variables: Tipo nombres/variables / instruccion1;  instruccionN;  **Php**  **Pobs: En lenguajes como PHP, donde el tipo de las variables es dinámico, no hay declaración de tipos. Por eso comentaremos la declaración on la incluiremos. Podemos decir que es el tipo que esperamos tenga una variable por los valores que serán asignados a dichas variables.  **Operadores para expresiones**  Resta, suma, multiplicación, división  Concatenación de cadenas de caracteres (coma)  Comparación igual, distinto  = <> = + * /  Comparación mayor , mayor igual  > > =   > > =    Comparación menor , menor igual  < <=   < < =    Lógicos / Booleanos   AND OR NOT   &&   !    Raiz cuadrada de E √(E)   raiz(E)   sgrt(E)    Operador Condicional / Ternario   SI E1 ENTONCES E2 SINO E3   E1? E2 : E3  Uso del operador ternario:  PROGRAMA Mayoria Edad ("determinar si una persona es mayor de edad*)	instrucciones			· · · · · · · · · · · · · · · · · · ·		
adecuadamente las instrucciones)  Estructura de control  Secuencial (se ejecuta una instrucción de valiables: Tipo nombresVariables   /* Declaración de valiables: Tipo nombresVariables   /* Instruccion1     instruccion1     instruccion1     instruccion1     instruccionN;  Pin ALGORITMO  Operadores para expresiones  Pesudocódigo PHP   P						
ALGORITMO nombreAlgoritmo   (1º descripción del algoritmo*)   (1º descripc						
Secuencial (see ejecuta una instrucción debajo de la otra)  Operadores para expresiones    Resta, suma, multiplicación, división   -+*/   -+*/   -+*/       Comparación mayor , mayor igual   ->     Comparación menor , menor igual   ->     Comparación menor , menor igual   ->     Lógicos / Booleanos   Raiz cuadrada de E √(E)   Operador Condicional / Ternario   SI E1 ENTONCESE2 SINO E3   E1? E2 : E3      Instruccion   /* Declaración de variables: Tipo nombresVariables   instruccion1; instruccionN;   instruccionN;     InstruccionN   Fin ALGORITMO   7	Estructura de			php</th <th>·</th>	·	
instrucción debajo de la otra)  Operadores para expresiones  Resta, suma, multiplicación, división	control Secuencial	(*descripción del algoritmo*)				
instrucción debajo de la otra)    Instruccion debajo de la otra)   Pin ALGORITMO   Pin ALGORI	(se ejecuta una	instruccion1				
Obs: En lenguajes como PHP, donde el tipo de las variables es dinámico, no hay declaración de tipos. Por eso comentaremos la declaración de tipos. Por eso comentaremos la declaración de tipos. Por eso los valores que serán asignados a dichas variable por los valores que serán asignados a dichas variables.  Operadores para expresiones    Resta, suma, multiplicación, división	instrucción	l '		instruccionin;		
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.  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 < <= <= < <= <= <= <= <= <= <= <= <= <=	debajo de la	FIN ALGORITMO				
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       &&    !         Raiz cuadrada de E √(E)       raiz(E)       sqrt(E)         Operador Condicional / Ternario       SI E1 ENTONCES E2 SINO E3       E1? E2 : E3    Uso del operador ternario: PROGRAMA MayoriaEdad     (*determinar si una persona es mayor de edad*)	otra)			es dinámico, no hay declaraciór comentaremos la declaración o decir que es el tipo que esperar	n de tipos. Por eso no la incluiremos. Podemos mos tenga una variable por	
Resta, suma, multiplicación, división	Operadores	*	Pseudo			
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 &&    !  Raiz cuadrada de E √(E) raiz(E) sqrt(E)  Operador Condicional / Ternario SI E1 ENTONCES E2 SINO E3 E1? E2 : E3  Uso del operador ternario:  PROGRAMA MayoriaEdad (*determinar si una persona es mayor de edad*)		Resta suma multiplicación división		,		
Concatenación de cadenas de caracteres , (coma) . (punto)  Comparación igual, distinto = <> == <> Comparación mayor , mayor igual > >= >=  Comparación menor , menor igual < <= <= <=	•					
Comparación igual, distinto = <> == <> Comparación mayor , mayor igual > >= Comparación menor , menor igual < <= Lógicos / Booleanos Raiz cuadrada de E √(E) Operador Condicional / Ternario Uso del operador ternario:  PROGRAMA MayoriaEdad (*determinar si una persona es mayor de edad*)	•			)		
Comparación mayor , mayor igual > >=				·	" '	
Lógicos / Booleanos       AND OR NOT       &&    !         Raiz cuadrada de E √(E)       raiz(E)       sqrt(E)         Operador Condicional / Ternario       SI E1 ENTONCES E2 SINO E3       E1? E2: E3         Uso del operador ternario:         PROGRAMA MayoriaEdad (*determinar si una persona es mayor de edad*)			> >:	=	> >=	
Raiz cuadrada de E √(E) raiz(E) sgrt(E)  Operador Condicional / Ternario SI E1 ENTONCES E2 SINO E3 E1? E2: E3  Uso del operador ternario:  PROGRAMA MayoriaEdad  (*determinar si una persona es mayor de edad*)		Comparación menor , menor igual	< <:	=	< <=	
Operador Condicional / Ternario  SI E1 ENTONCES E2 SINO E3 E1? E2: E3  Uso del operador ternario:  PROGRAMA MayoriaEdad  (*determinar si una persona es mayor de edad*)		Lógicos / Booleanos	AND	OR NOT	&&    !	
Uso del operador ternario:  PROGRAMA MayoriaEdad  (*determinar si una persona es mayor de edad*)		Raiz cuadrada de E √(E)	raiz(E)		sgrt( E )	
PROGRAMA MayoriaEdad  (*determinar si una persona es mayor de edad*)		Operador Condicional / Ternario	SI E1	ENTONCES E2 SINO E3	E1? E2: E3	
(*determinar si una persona es mayor de edad*)		Uso del operador ternario:				
(*determinar si una persona es mayor de edad*)		PROGRAMA MayoriaEdad				
		·				
i Doolean esiviavor, Enjero edad <u>amno</u> domore mensale						
·		Boolean esMayor, Entero edad, String nombre,mensaje ESCRIBIR( "Ingrese su nombre" ) LEER(nombre) ESCRIBIR( "Ingrese su edad" )				
ESCRIBIR( "Ingrese su edad" )						
LEER(edad)						
esMayor <- (edad >= 18)						
mensaje <- SI esMayor ENTONCES nombre," es mayor de edad" SINO nombre," es menor de edad "		mensaje <- SI esMayor ENTONCES nombre,	' es mayo	or de edad" SINO nombre,	" es menor de edad "	

	ESCRIBIR (mensaje)			
	FIN PROGRAMA			
Modularización : Funciones	(**  *Descripcion del módulo sin retorno (¿qué hace?)  *)  MODULO nombre(tipo1 pf1, tipo2 pf2,,tipoN pfN) RETORNO Ø   instruccion1     instruccionN	/** *Descripcion del modulo sin retorno (¿qué hace?) * @param tipo \$pf1 * @param tipo2 \$pf2 * * @param tipo \$pfN		
	FIN MODULO	<pre>function nombre(\$pf1, \$pf2,,\$pfN){   instruccion1;    instruccionN; }</pre>		
	(**  * Descripcion del módulo con retorno (¿qué hace?)  *)  MODULO nombre(tipo1 pf1, tipo2 pf2,,tipoN pfN) RETORNO tipo  instruccion1     instruccionN   RETORNO(E) FIN MODULO	/** *Descripcion del modulo con retorno (¿qué hace?) * @param tipo1 \$pf1 * @param tipo2 \$pf2 * * @param tipoN \$pfN * @return tipo */ function nombre(\$pf1, \$pf2,,\$pfN){ instruccion1;		
	(obs.: tipo puede ser boolean, integer, float, string, etc.)	instruccionN; return E; }		
Alternativa: Instrucción IF	SI (condición) ENTONCES  instruccion1  instruccionN FIN SI	<pre>if (condición){   Instrucción1;   InstrucciónN; }</pre>		
	SI (condición) ENTONCES  instruccionA1  instruccionAN SINO  instruccionB1  instruccionBN FIN SI	<pre>if (condición){    InstrucciónA1;    InstrucciónAN; } else{    InstrucciónB1;    InstrucciónBN; }</pre>		
	SI (condición) ENTONCES  instruccionA1  instruccionAN OTRO-SI (condiciónB) ENTONCES  instruccionB1  instruccionBN OTRO-SI (condiciónM) ENTONCES  instruccionM1  instruccionMN FIN SI	<pre>if (condiciónA){    InstrucciónA1;    InstrucciónAN; } elseif (condiciónB){    InstrucciónB1;    InstrucciónBN; } elseif (condiciónM){    InstrucciónM1;    InstrucciónMN; }</pre>		
	SI (condición) ENTONCES	if (condiciónA){		

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