

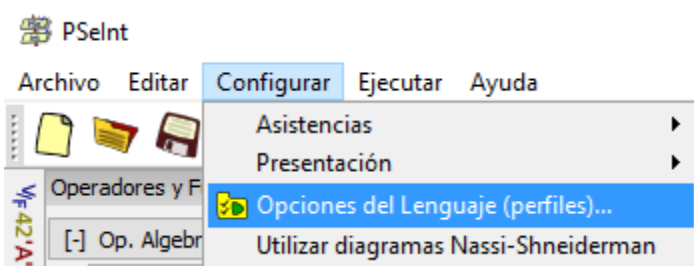
Programación con PSEINT

1.- Introducción

El programa utiliza pseudocódigo, un lenguaje de programación ficticio cuya principal misión es que el programador pueda centrarse en los aspectos lógicos de la programación, dejando el apartado técnico para cuando se vea la sintaxis de un lenguaje de programación verdadero. Además, es un entorno de desarrollo de programación o IDE (Integrated Development Environment). Los programas pueden escribirse en un simple editor de textos. Sin embargo, un IDE ayuda a la programación: coloreando las palabras clave, mostrando las instrucciones clave cuando las tenemos a medio escribir (Similar a cuando escribimos mensajes con el móvil), resaltando los errores, etc...

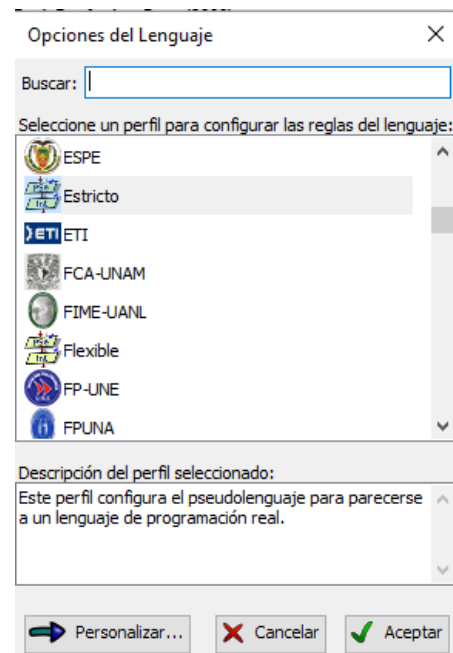
Podemos descargar PSEINT en la siguiente página: <http://pseint.sourceforge.net/>

En PSeInt podemos elegir entre 2 perfiles:



El perfil Estricto, hace que PSEINT deba seguir las normas de la mayoría de los lenguajes de bajo nivel.

El perfil Flexible es menos riguroso y no da error si nos saltamos algunas de esas reglas.



En la asignatura funcionaremos en modo estricto porque, si bien es más complicado, facilitará el aprendizaje de cualquier otro lenguaje real. Aunque PSeInt subraya con rojo los errores de sintaxis, también permite verificarlos mediante **Ejecutar/Verificar Sintaxis**


Algunas de las normas comunes a otros lenguajes a seguir en el perfil estricto son:

- Al igual que muchos lenguajes como C o Javascript, las sentencias siempre finalizan en punto y coma.
- Los identificadores, o nombres de variables, deben constar sólo de letras, números y/o guión_bajo (_), comenzando siempre con una letra. No puede contener símbolos(@, #, ▼, [,], =, etc), ni ñes (ñ ó Ñ) y no puede tener espacios en blancos. .
- Los identificadores, o nombres de variables, no pueden coincidir con las palabras reservadas del lenguaje (instrucciones).
- Se pueden crear variables del mismo tipo en una sola línea separadas por comas **Definir n1,n2,suma Como Entero;**
- Se pueden introducir comentarios después del ; de una línea, o en una línea separadas, mediante el uso de la doble barra (//). Todo lo que precede a //, hasta el fin de la línea, no será tomado en cuenta al interpretar el algoritmo.
- No puede haber instrucciones fuera del proceso (antes de PROCESO, o después de FINPROCESO), aunque sí comentarios.
- Las estructuras no secuenciales pueden anidarse. Es decir, pueden contener otras adentro, pero la estructura contenida debe comenzar y finalizar dentro de la contenedora.

- En las constantes numéricas, el punto (.) es el separador decimal.

Algunas de las normas de PSEINT no comunes a otros lenguajes de programación son:

PSeInt no es case sensitive, por lo tanto colocar Escribir con mayúsculas y minúsculas es lo mismo y no genera errores de ningún tipo, pero por respeto a la sintaxis mostrada por los botones se debe escribir con mayúscula inicial, evitando así errores de formato.

Para ejecutar un programa puedes usar el ícono de ejecutar  o pulsar F9:

Cualquier programa en PSEINT tiene que tener al menos un Algoritmo (versiones antiguas utilizan la palabra **Proceso**).

```
1  Algoritmo sin_titulo
2
3  FinAlgoritmo
4
```

Conviene asignar un nombre al algoritmo, sustituyendo **sin_titulo** por un nombre que tenga relación con el programa. El nombre del pseudocódigo en ninguna sintaxis puede tener espacios y en sintaxis estricta tampoco caracteres acentuados.

No confundir el nombre del algoritmo con el del archivo que guardemos. Pueden ser distintos, aunque recomendamos que coincidan para tenerlo mejor localizado.

2.- Empezando a programar con PSEINT

A continuación, deberás ir escribiendo y probando todos los siguientes programas en PSEINT. De nada vale escribirlo sin entender lo que estás haciendo, ya que no sabrás realizar las prácticas que se manden y fallarás el examen. Pregunta las dudas al profesor.

```
1  Algoritmo MiPrimerPrograma
2      Escribir "Tengo";
3      Escribir 18;
4      Escribir "años";
5  FinAlgoritmo
```

La palabra reservada Escribir escribe en la pantalla lo que lleva escrito a continuación.

Si quisiéramos que todo apareciese en la misma línea deberíamos utilizar la instrucción reservada **Sin Saltar**. Modifica el programa para que quede así:

```
1  Algoritmo MiPrimerPrograma
2      Escribir "Tengo " Sin Saltar;
3      Escribir 18 Sin Saltar;
4      Escribir " años";
5  FinAlgoritmo
```

También es válido y más visual concatenar elementos mediante el separador coma.

```
1  Algoritmo MiPrimerPrograma
2      Escribir "Tengo ",18," años";
3  FinAlgoritmo
```

Los textos van entre comillas, los números no. Podemos escribir un número entre comillas y será mostrado sin error, sin embargo, el lenguaje lo reconocerá como un texto y no como un número. *Nota: Las comillas deben ser siempre simples y nunca tipográficas.*

Para comprobar este hecho, prueba a escribir el siguiente código.

```
1  Algoritmo MiSegundoPrograma
2      Escribir "2*6" Sin Saltar;
3      Escribir "=" Sin Saltar;
4      Escribir 2*6;
5  FinAlgoritmo
```

En la línea dos, "2*6" es considerado texto y no realiza ninguna operación matemática, mientras que en la línea cuatro sí. Recordar que en programación, el signo * sirve para multiplicar.

3.- Variables

PSEINT utiliza un número muy limitado de tipos de datos:

Tipo de dato	Explicación
Entero	Nº sin decimales
Real	Nº con decimales
Caracter o cadena	Cadenas de caracteres es decir cualquier letra, palabra frase u oración.
Logico	Cuando necesitamos guardar una expresión lógica. Esta tipo de variable ocupa muy poca memoria ya que sólo puede tener 2 valores (verdadero o falso), ojo sin comillas.

En el modo estricto debemos declarar las variables al principio del algoritmo o proceso. Esto se hace así:

Definir Nombre_variable Como Tipo_de_dato;

Algunos ejemplos de declaración de variables:

```
Definir AlturaMontaña Como Entero;
Definir NotaExamen Como Real;
Definir Pais Como Caracter;
Definir Casado Como Logico;
```

Una vez definida la variable, para asignarle un valor utilizamos los signos <-

En el siguiente ejemplo asignamos el valor 25 a la variable edad. Por ejemplo:

```
AlturaMontaña<-3718
NotaExamen<-6.75
Pais<-"España"
Casado<-Verdadero
```

Algunas de las normas sobre utilización de variables a seguir en el perfil estricto, comunes a otros lenguajes, son:

- Los identificadores, o nombres de variables, deben constar sólo de letras, números y/o guión bajo (_), comenzando siempre con una letra. No puede contener símbolos (@, #, ▼, [,], =, etc.), ni ñes (ñ ó Ñ) y no puede tener espacios en blancos. .
- Los identificadores, o nombres de variables, no pueden coincidir con las palabras reservadas del lenguaje .PSEINT *colorea de azul Las palabras reservadas.*
- Se pueden crear variables del mismo tipo en una sola línea separadas por comas
Definir n1,n2,suma Como Entero;

Ejemplo de uso de variables

Este programa es poco útil, ya que siempre suma los mismos valores, el dos y el tres. Lo más lógico es que el propio programa nos pregunte qué números queremos sumar.

```
1 Algoritmo SumaConVariables
2   Definir n1 Como Entero;
3   Definir n2 Como Entero;
4   n1<-2;
5   n2<-3;
6   Escribir "La suma es ",n1+n2;
7   FinAlgoritmo
```

Esto se consigue con **la instrucción Leer**. Cuando ejecutamos un programa, esta instrucción espera a que escribamos con el teclado y, cuando pulsemos la tecla intro, asignará lo escrito a la variable que acompaña a la instrucción.

```
1 Algoritmo SumaConVariables
2   Definir n1 Como Entero;
3   Definir n2 Como Entero;
4   Escribir "Dime un número";
5   Leer n1;
6   Escribir "Dime un número";
7   Leer n2;
8   Escribir "El resultado de la suma es ",n1+n2;
9   FinAlgoritmo
```

Si quisiéramos guardar el resultado para usarlo más adelante deberíamos utilizar otra nueva variable, amplía el algoritmo para que el resultado se almacene en otra variable

4.- Selección simple. “Si Entonces”

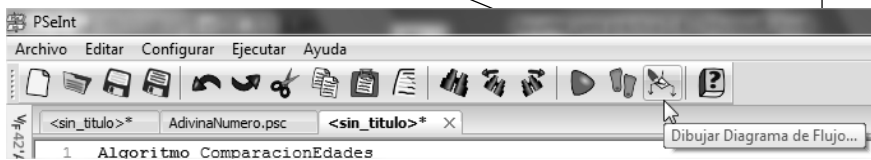
Esta función es básica en programación. Lo que viene a decir es que si ocurre “algo” que haga “algo”. Un ejemplo.

```

Algoritmo ComparacionEdades
    Definir EdadAlumno1 Como Entero;
    Definir EdadAlumno2 Como Entero;
    Escribir "Dime la edad del Alumno1";
    Leer EdadAlumno1;
    Escribir "Dime la edad del Alumno1";
    Leer EdadAlumno2;
    Si EdadAlumno1=EdadAlumno2 Entonces
        Escribir "Ambos alumnos tienen la misma edad";
    FinSi
    Si EdadAlumno1>EdadAlumno2 Entonces
        Escribir "El alumno1 es mayor que el alumno2";
    FinSi
    Si EdadAlumno1<EdadAlumno2 Entonces
        Escribir "El alumno2 es mayor que el alumno1";
    FinSi
FinAlgoritmo
  
```

Realiza el diagrama de flujo del algoritmo.

Compruébalo con el que te muestra el programa.



El anterior programa es válido pero es bastante ineficiente, ya que obliga al programa a comprobar todas las situaciones incluso aunque se cumpla la primera (edades iguales), Para mejorar el programa se utiliza la función Si... Sino. Cuando se cumple una de las condiciones deja de comprobar todo lo demás.

```

Algoritmo ComparacionEdades
    Definir EdadAlumno1 Como Entero;
    Definir EdadAlumno2 Como Entero;
    Escribir "Dime la edad del Alumno1";
    Leer EdadAlumno1;
    Escribir "Dime la edad del Alumno1";
    Leer EdadAlumno2;
    Si EdadAlumno1=EdadAlumno2 Entonces
        Escribir "Ambos alumnos tienen la misma edad";
    Sino
        Si EdadAlumno1>EdadAlumno2 Entonces
            Escribir "El alumno1 es mayor que el alumno2";
        Sino
            Si EdadAlumno1<EdadAlumno2 Entonces
                Escribir "El alumno2 es mayor que el alumno1";
            FinSi
        FinSi
    FinSi
FinAlgoritmo
  
```

Cuando queremos comprobaciones más complejas, podemos utilizar los operadores lógicos Y, O, NO. Veamos el siguiente ejemplo:

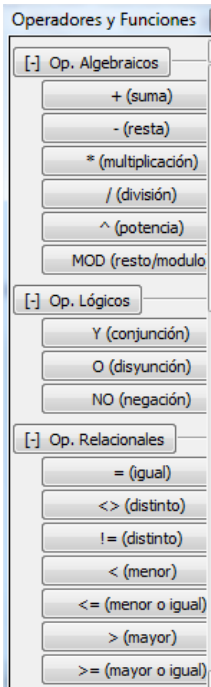
```

Algoritmo Beca
    Definir EdadAlumno Como Entero;
    Escribir "Dime la edad del Alumno";
    Leer EdadAlumno;
    Si EdadAlumno>=18 Y EdadAlumno<=25 Entonces
        Escribir "El alumno SÍ recibe beca";
    Sino
        Escribir "El alumno NO recibe beca";
    FinSi
FinAlgoritmo
  
```

Realizar el diagrama de flujo

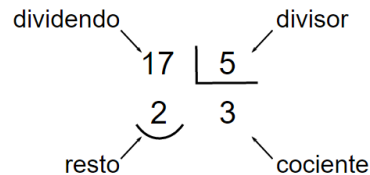
5.- Operaciones con PSEINT

PSEINT ofrece una ayuda de las operaciones matemáticas que puede utilizar en el lateral izquierdo.



Aparte de las operaciones más comunes, suma, resta, multiplicación, etc. Tenemos alguna otra:

La función **MOD** devuelve el resto de una división;



Si escribimos en PSEINT...

Ejemplo:

Escribir 17 MOD 5; //Mostrará en pantalla 2

Para calcular el cociente, utilizamos la función **TRUNC**. Lo que calcula esta función es la parte entera de un número.

Ejemplo:

Escribir TRUNC(7.25); //Mostrará en pantalla 7

Luego la parte entera de una operación división mostrará el cociente.

Ejemplo:

Escribir TRUNC(17/2); Mostrará en pantalla 8

La función **REDON**, redondea un número real al entero más cercano por abajo o por arriba:

Escribir REDON (3.456); mostrará 3

Escribir REDON (3.51); mostrará 4

La función **Azar(x)**, devuelve un número entero aleatorio desde 0 y x-1.

La función **Aleatorio (x,y)** devuelve un número entero aleatorio entre x e y.

1	Algoritmo Sorteo	1	Algoritmo Sorteo
2	Escribir azar(1000)	2	Escribir Aleatorio(1,10);
3	FinAlgoritmo	3	FinAlgoritmo
4		4	

Realizar un algoritmo que simule el lanzamiento de una moneda, puede salir caro o cruz. El ordenador nos pregunta qué elegimos, simula el lanzamiento de la moneda y nos dice si hemos acertado.

Realizar un algoritmo que dado el dividendo y divisor nos devuelva el cociente y el resto de la división.

FUNCIONES CON CADENAS DE CARACTERES

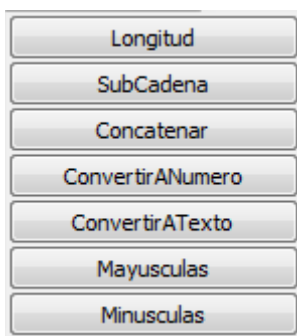
Permiten manipular las cadenas de caracteres.

Escribe el siguiente algoritmo para comprobar qué hacen las distintas funciones.

```

1  Algoritmo cadenasCaracteres
2      Escribir Longitud("Esto es una cadena de caracteres");
3      Escribir Subcadena("murciélago",3,6);
4      Escribir Concatenar("Esto es","un murciélago");
5      Escribir Mayusculas("pepe");
6      Escribir Minusculas("BURRITO");
7      Escribir ConvertirANumero("123");
8      Escribir ConvertirATexto(123);
9  FinAlgoritmo
10

```



Devuelve la longitud de una cadena de caracteres

Devuelve la parte de la cadena entre las posiciones indicadas

Devuelve una cadena unión de dos.

Convierte a número una cadena (debe estar formada por números).

Convierte a texto una cadena de números.

Pasa a mayúsculas o minúsculas una cadena

Realizar un algoritmo que cuente el número de vocales de una palabra

Realizar un algoritmo que cuente el número de palabras de una frase.

Realizar un algoritmo que genere frases de forma aleatoria, para el mismo se introducirán 3 sujetos, 3 verbos y 3 predicados. El algoritmo generará una frase al azar con cualquier sujeto, verbo y predicado.

Realizar un juego de 3 preguntas en las que el ordenador nos dice si hemos acertado, la respuesta, la respuesta se puede escribir con mayúsculas o minúsculas

6.- Selección múltiple, “Segun Hacer”

La siguiente instrucción permite que el programa realice unas instrucciones u otras en función del valor de una variable.

```

1  Segun variable_numerica Hacer
2      opcion_1:
3          secuencia de acciones 1
4      opcion_2:
5          secuencia de acciones 2
6      opcion_3:
7          secuencia de acciones 3
8  De Otro Modo:
9      secuencia de acciones dom
10 Fin Segun

```

Variable del algoritmo

Si la variable tiene el valor indicado se ejecutan las acciones.

Si la variable no tiene ningún valor de los anteriores ejecutará las acciones

Algoritmo SeleccionMultiple

```

Definir eleccion Como Entero;
Escribir "Menú"
Escribir "Pulsa 1 para opción 1";
Escribir "Pulsa 2 para opción 2";
Escribir "Pulsa 3 para opción 3";
Leer eleccion
Segun eleccion Hacer
    1:
        Escribir "Has elegido la opción 1";
    2:
        Escribir "Has elegido la opción 2";
    3:
        Escribir "Has elegido la opción 3";
De Otro Modo:
    Escribir "Tenías que pulsar 1, 2 ó 3";
Fin Segun
FinAlgoritmo

```

Realizar un algoritmo en el que el ordenador nos pide dos valores, luego nos pregunta qué queremos hacer

1 Sumar/2 restar/3 multiplicar/4 dividir, le introducimos la opción y el ordenador nos devuelve la operación solicitada.

7.- Iteración For Next. “Para Hasta”

Los bucles sirven para que una secuencia de instrucciones se repita varias veces.

Concretamente, los bucles “Para” se utilizan para que un trozo de algoritmo se repita el número de veces que le indiquemos.

Su utilización es:

```

Para <variable> <- <inicial> Hasta <final> Con Paso <paso> Hacer
    <instrucciones>
FinPara

```

Veamos algunos ejemplos

//En el primer ejemplo se realiza una cuenta desde el 5 hasta el 100 avanzando de 3 en 3. En la primera repetición cuenta vale 5, en la segunda repetición vale 8, en la tercera 11,

```

Algoritmo CuentaIncremental
Definir cuenta como Entero;
Para cuenta<-5 Hasta 100 Con Paso 3 Hacer
    Escribir cuenta;
FinPara

```

FinAlgoritmo

En el segundo ejemplo se realiza una cuenta desde el 50 hasta el 20 retrocediendo de 2 en 2

```

Algoritmo CuentaIncremental
  Definir cuenta como Entero;
  Para cuenta<-50 Hasta 20 Con Paso -2 Hacer
    Escribir cuenta;
  FinPara
FinAlgoritmo

```

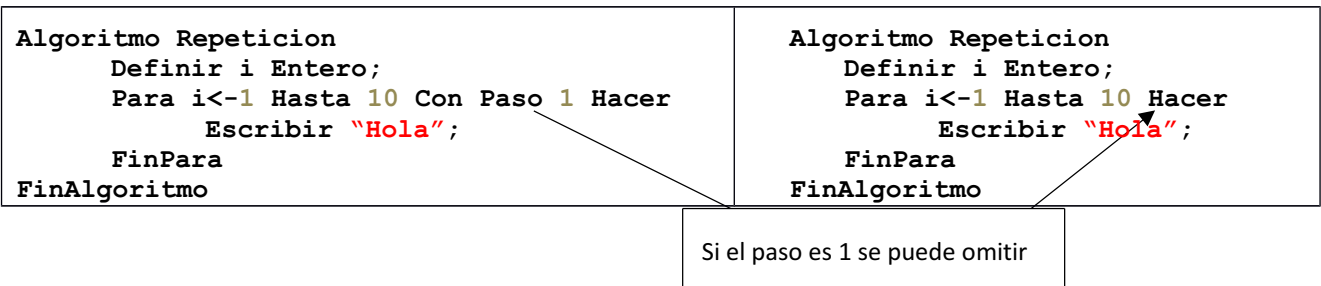
Si quiero hacer un programa completo que permita todas las posibilidades, debería escribir algo como lo siguiente:

```

Algoritmo CuentaIncremental
  Definir num, intervalo, cuenta como Entero;
  Escribir "¿Hasta qué número quieres llegar?";
  Leer num;
  Escribir "¿Cuál es el intervalo entre número y número?";
  Leer intervalo;
  Para cuenta<-1 Hasta num Con Paso intervalo Hacer
    Escribir cuenta;
  FinPara
FinAlgoritmo

```

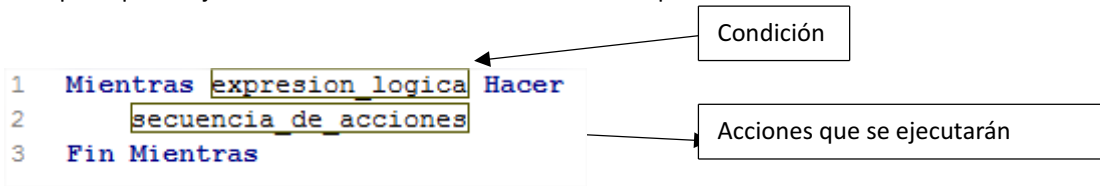
Cuando simplemente quiero que algo se repita un determinado número de veces, por ejemplo repetir Hola 10 veces:



Realizar un algoritmo que nos haga la media de una serie de números, debe de funcionar así, primero nos pregunta por el número de datos a introducir, luego los vamos introduciendo y al final nos da la media.

8.- Iteración tipo While. Mientras Hacer

Sirve para que se ejecuten unas instrucciones cuando se cumpla una condición.



Copia y comprueba el siguiente algoritmo

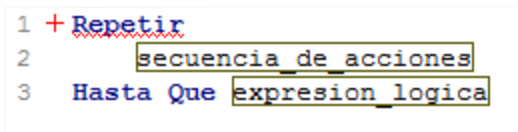
```

Algoritmo BucleSinSalida
    Definir respuesta como Caracter;
    Mientras respuesta<>"la clave" Hacer
        Escribir "Escribe la clave";
        Leer respuesta;
    Fin Mientras
    Escribir "Bravo! has sabido escribir ->la clave";
FinAlgoritmo
  
```

Realizar un algoritmo en el que el ordenador genere un número aleatorio entre 1 y 10. El ordenador nos va preguntando ¿Qué número es? Y responde “mi número es mayor” o “mi número es menor” hasta que acertamos, entonces responde “Has acertado” y se acaba el programa.

9.- Iteración tipo Do While. Repetir Hasta que

Con este tipo de estructura se ejecutará una vez todas las acciones, si se cumple la condición se saldrá del bloque si no se volverán a repetir hasta que se cumpla la condición.



```

Algoritmo ListaNumero
    Definir ultimo como entero;
    Definir contador como entero;
    contador<-0
    Escribir "¿Hasta qué número cuento?";
    Leer ultimo;
    Repetir
        Escribir contador Sin Saltar;
        Escribir "-" Sin Saltar;
        contador<-contador+1
    Hasta Que ultimo=contador;
    Escribir "Al fin acabé";
FinAlgoritmo
  
```

Modifica el algoritmo para que el programa cuente hasta el número introducido