

EXPRESIÓN DE PROBLEMAS Y ALGORITMOS

PSeInt

(Interprete de Pseudocódigo)

Introducción

- PSeInt es un desarrollo libre, que permite realizar las tareas de diseño, programación y seguimiento de algoritmos en pseudocódigo.
- Contiene funcionalidades que facilitan el diseño mediante diagramas de flujo y su posterior traducción a pseudocódigo.
- Además, incluye herramientas de depuración, como ser sugerencia de sintaxis y seguimiento de la ejecución.


Descarga e instalación

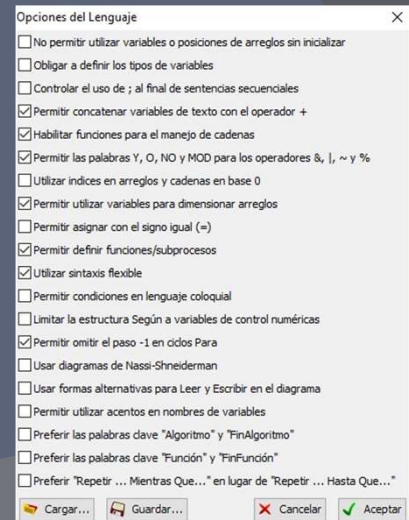
- Es un software libre, que permite colaboraciones de distintas instituciones con sus implementaciones particulares del pseudocódigo.
- La web de descarga es: <http://pseint.sourceforge.net/>
 - Hay versiones para varias plataformas
 - Última versión a la fecha: 02/02/2018
- Una vez descargado el archivo, ejecutarlo para instalar el software

Primera ejecución

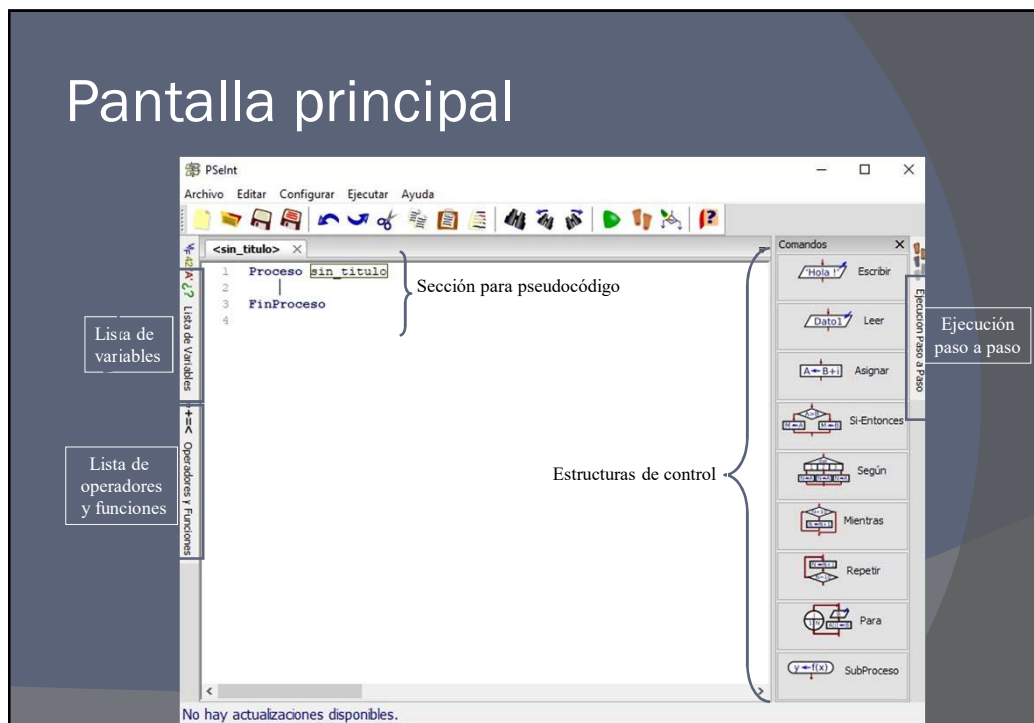
- Al ejecutarlo por primera vez, deberá configurar el perfil. Para esto, seleccione el menú “Configurar”, luego “Opciones del lenguaje (perfiles)”.
- En el cuadro de dialogo, ubique la opción “<Personalizado>” y luego pulse el botón “Personalizar...”

Configuración del perfil

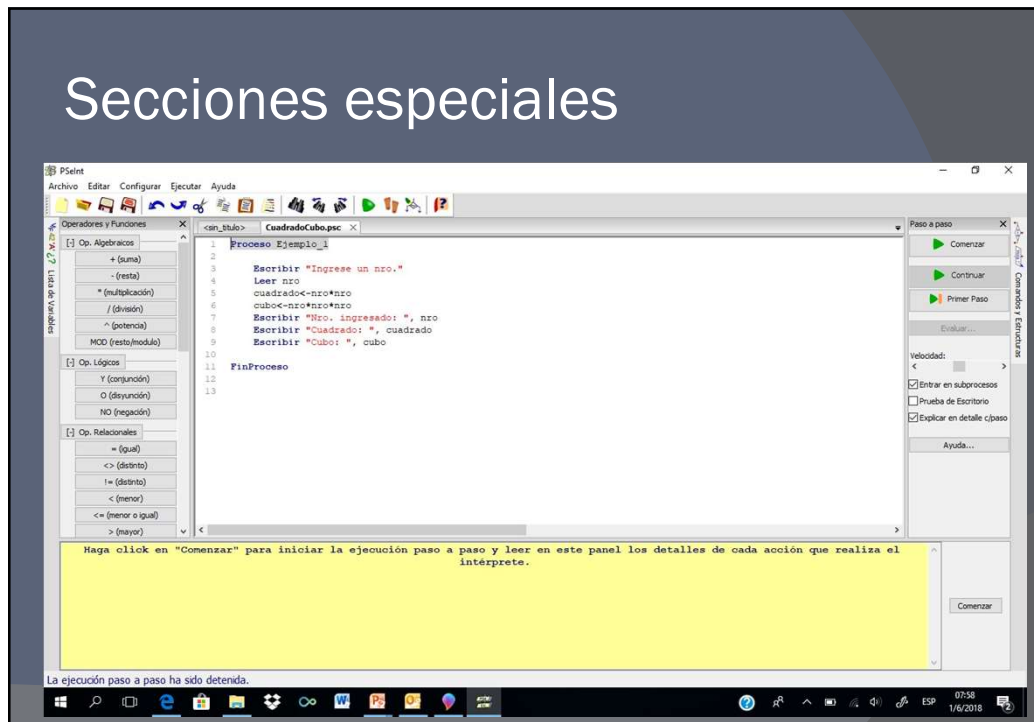
- El perfil debe quedar configurado de la siguiente forma: 
- Luego, guarde la configuración en un archivo.
- Finalmente, pulse el botón "Aceptar"



Pantalla principal



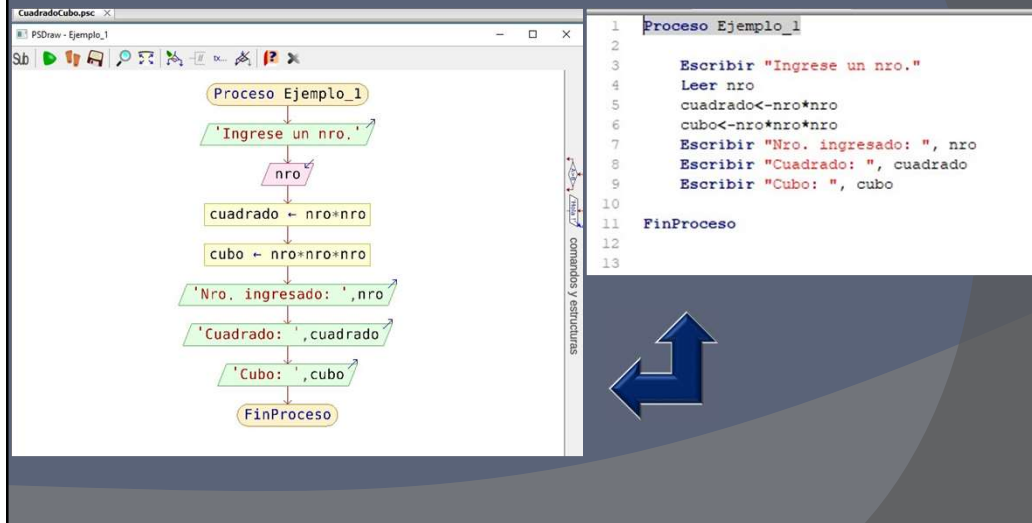
Secciones especiales



Programación del algoritmo (1)

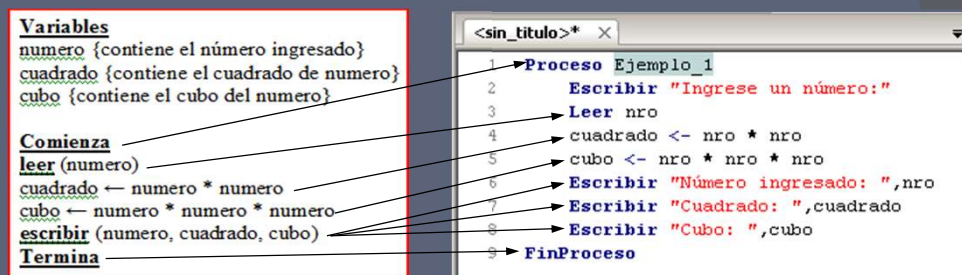
- PSeInt incluye 2 maneras de programar un algoritmo:
 - La primera es comenzando con el diagrama de flujo: desde los comandos de la barra derecha, se puede diseñar el diagrama de flujo y luego, el software generará el pseudocódigo de manera automática.
 - La segunda es la inversa de la primera: a partir del pseudocódigo es posible generar automáticamente el diagrama de flujo (tecla F7).

Programación del algoritmo (2)



Ejemplo 1: Secuenciación

- Dado un número, calcular sus potencias cuadrada y cúbica.



Sentencias de selección (1)

- PSeInt incluye las sentencias de selección simple, doble y múltiple.
- Estas estructuras pueden agregarse al diagrama de flujo y/o al pseudocódigo desde la barra de comandos.



Sentencias de selección (2)

- Otra alternativa es ingresarlas manualmente, considerando la sintaxis básica:

Selección Simple	Selección doble	Selección múltiple
<pre> Si <u>expresion logica</u> Entonces <u>acciones por verdadero</u> Fin Si </pre>	<pre> Si <u>expresion logica</u> Entonces <u>acciones por verdadero</u> Sino <u>acciones por falso</u> Fin Si </pre>	<pre> Segun <u>variable numerica</u> Hacer opcion 1: <u>secuencia de acciones 1</u> opcion 2: <u>secuencia de acciones 2</u> opcion 3: <u>secuencia de acciones 3</u> De Otro Modo: <u>secuencia de acciones don</u> Fin Segun </pre>

Ejemplo 2: Selección doble

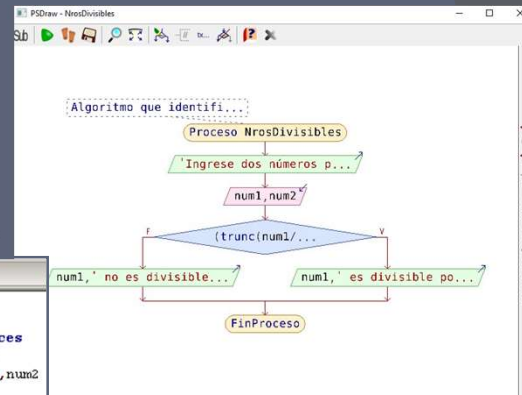
- Determinar si un número es divisible por otro

Variables
num1, num2 {contienen los valores ingresados por el usuario}

Comienza
Leer (num1, num2)
Si (num1 / num2) * num2 = num1 entonces
 Escribir "num1 es divisible por num2"
Sino Escribir "num1 no es divisible por num2"
Fin si
Termina

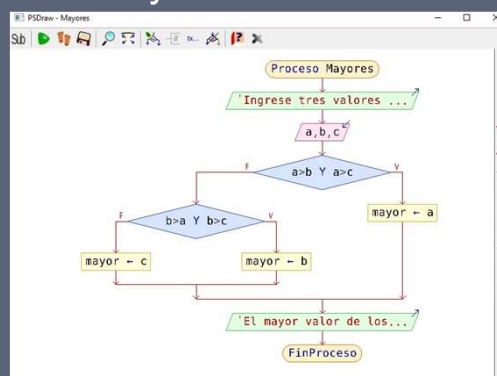
```

1 Proceso Ejemplo_2
2   Leer num1, num2
3   Si (trunc(num1 / num2) * num2 = num1) entonces
4       Escribir num1, " es divisible por ", num2
5   Sino Escribir num1, " no es divisible por ", num2
6   Fin si
7   FinProceso
  
```



Ejemplo 3: Selecciones anidadas

- Dados 3 números, determinar cual es el mayor

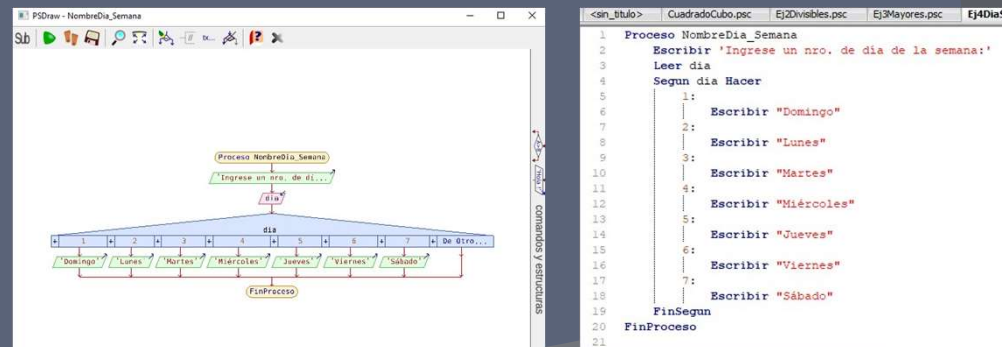


```

1 Proceso Mayores
2   Escribir "Ingrese tres valores numéricos:"
3   Leer a, b, c
4   Si a > b y a > c Entonces
5       mayor <- a
6   Sino
7       Si b > a y b > c Entonces
8           mayor <- b
9       Sino
10          mayor <- c
11      FinSi
12  FinSi
13  Escribir "El mayor valor de los ingresados es: ", mayor
14  FinProceso
  
```

Ejemplo 4: Selección múltiple

- Dado un número, indicar el día de la semana asociado



Depuración

- PSeInt tiene una función que chequea la sintaxis antes de ejecutar un algoritmo.
- Además, permite realizar la prueba y depuración de los algoritmos mediante 3 formas de pruebas de ejecución:
 - Ejecutar: ejecuta todo el algoritmo sin pausas
 - Ejecutar paso a paso: ejecuta el código con pausas
 - Ejecutar explicada: explica las sentencias que se están ejecutando

Sentencias de iteración (1)

- PSeInt incluye los 3 tipos de bucles vistos en la materia: *Mientras*, *Para* y *Repetir*.
- La estructura *Mientras* tiene la misma sintaxis que la utilizada en pseudocódigo, mientras que la *Repetir* y *Para* tienen pequeñas variaciones.

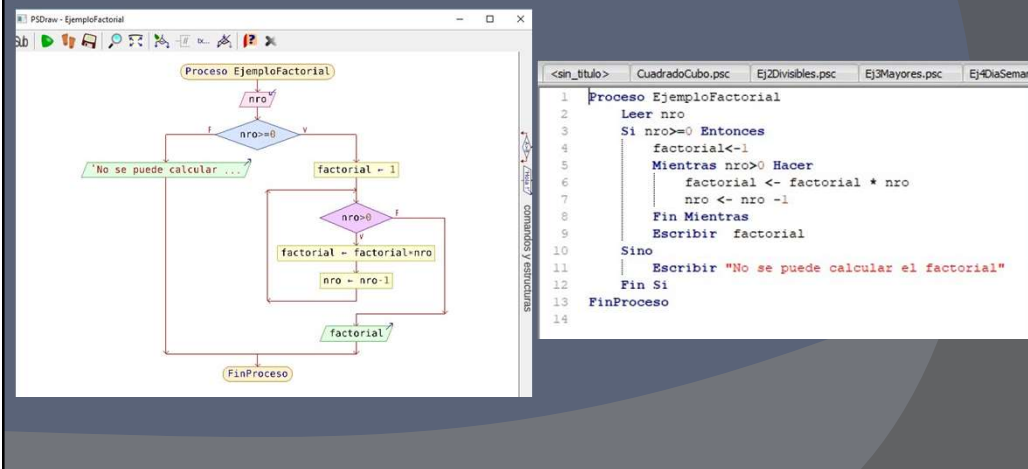
Sentencias de iteración (2)

- La sintaxis de cada una es:

Mientras	Repetir	Para
<pre> Mientras <u>expresion logica</u> Hacer <u>secuencia de acciones</u> Fin Mientras </pre>	<pre> Repetir <u>secuencia de acciones</u> Hasta Que <u>expresion logica</u> </pre>	<pre> Para <u>variable numerica</u> <- <u>valor inicial</u> Hasta <u>valor final</u> Con Paso <u>paso</u> Hacer <u>secuencia de acciones</u> Fin Para </pre>

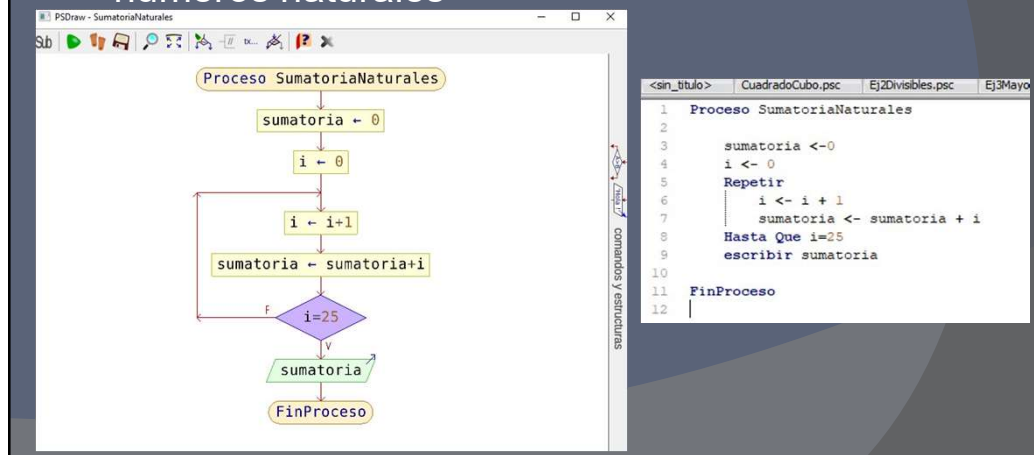
Ejemplo 5

- Calcular el factorial de un número



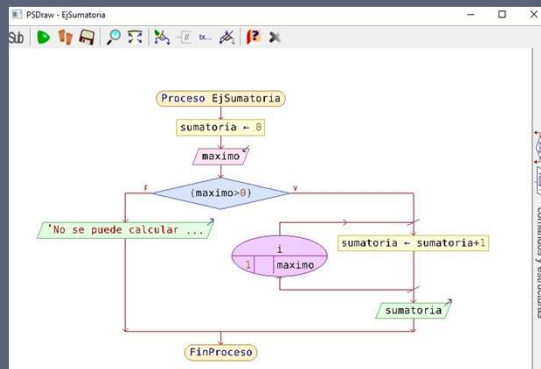
Ejemplo 6

- Calcular la sumatoria de los 25 primeros números naturales



Ejemplo 7

- Calcular la sumatoria de los n primeros números naturales

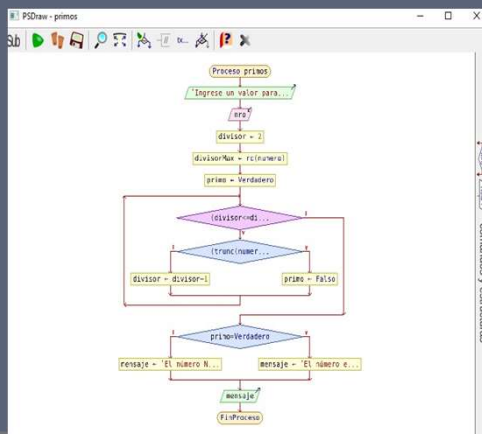


```

Proceso EjSumatoria
sumatoria <- 0
Leer maximo
Si (maximo > 0) Entonces
  Para i <- 1 Hasta maximo Hacer
    sumatoria <- sumatoria + i
  FinPara
  Escribir sumatoria
SiNo
  Escribir "No se puede calcular la sumatoria"
FinSi
FinProceso
  
```

Ejemplo 8

- Dado un número, indicar si es primo o no



```

Proceso primos
escribir "Ingrese un valor para determinar si es primo o no"
leer nro
divisor <- 2
divisorMax <- rc(numero)
primo <- Verdadero
mientras (divisor <= divisorMax y primo = Verdadero) Hacer
  si (trunc(numero/divisor) * divisor = numero) Entonces
    primo <- Falso
  SiNo
    divisor <- divisor + 1
  FinSi
FinMientras
si primo = Verdadero entonces
  mensaje <- "El número es primo"
sino
  mensaje <- "El número NO es primo"
FinSi
escribir mensaje
FinProceso
  
```