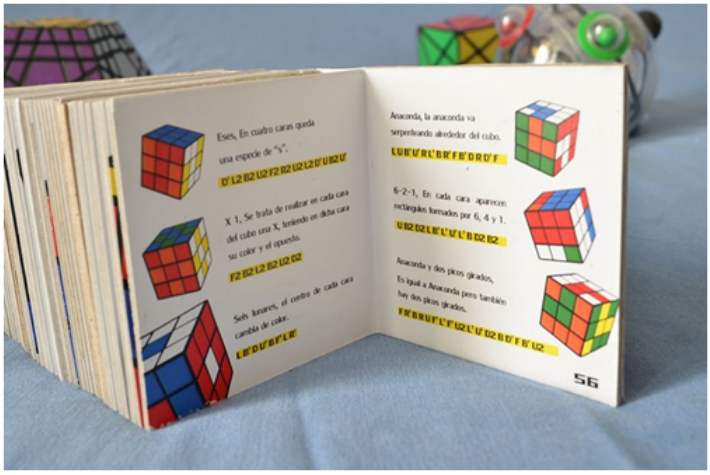
# 1.A. Introducción a la programación.

## 2. Programas y programación.

### 2.2. Algoritmos y programas.

Después de analizar en detalle el problema a solucionar, hemos de diseñar y desarrollar el algoritmo adecuado. Pero, **¿qué es un algoritmo?**

**Algoritmo:** secuencia ordenada de pasos, descrita sin ambigüedades, que conducen a la solución de un problema dado.




Los algoritmos son independientes de los lenguajes de programación y de las computadoras donde se ejecutan. Un mismo algoritmo puede ser expresado en diferentes lenguajes de programación y podría ser ejecutado en diferentes dispositivos. Piensa en una receta de cocina, ésta puede ser expresada en castellano, inglés o francés, podría ser cocinada en fogón o vitrocerámica, por un cocinero o más, etc. Pero independientemente de todas estas circunstancias, el plato se preparará siguiendo los mismos pasos.

La diferencia fundamental entre algoritmo y **programa** es que, en el segundo, los pasos que permiten resolver el problema, deben escribirse en un determinado **lenguaje de programación** para que puedan ser ejecutados en el ordenador y así obtener la solución.

**Los lenguajes de programación** son sólo un medio para expresar el algoritmo y el ordenador un procesador para ejecutarlo. El diseño de los algoritmos será una tarea que necesitará de la creatividad y conocimientos de las técnicas de programación. Estilos distintos, de distintos programadores a la hora de obtener la solución del problema, darán lugar a algoritmos diferentes, igualmente válidos.

En esencia, todo problema se puede describir por medio de un algoritmo y las características fundamentales que éstos deben cumplir son:

- Debe ser **preciso** e indicar el orden de realización paso a paso.
- Debe estar **definido**, si se ejecuta dos o más veces, debe obtener el mismo resultado cada vez.
- Debe ser **finito**, debe tener un número finito de pasos.

Pero cuando los problemas son complejos, es necesario descomponer éstos en subproblemas más simples y, a su vez, en otros más pequeños. Estas estrategias reciben el nombre de **diseño descendente** o **diseño modular (top-down design)**. Este sistema se basa en el lema **divide y vencerás**.

Para representar gráficamente los algoritmos que vamos a diseñar, tenemos a nuestra disposición diferentes herramientas que ayudarán a describir su comportamiento de una forma precisa y genérica, para luego poder codificarlos con el lenguaje que nos interese. Entre otras tenemos:

· **Diagramas de flujo:** Esta técnica utiliza símbolos gráficos para la representación del algoritmo. Suele utilizarse en las fases de análisis.

· **Pseudocódigo:** Esta técnica se basa en el uso de palabras clave en lenguaje natural, constantes, variables, otros objetos, instrucciones y estructuras de programación que expresan de forma escrita la solución del problema. Es la técnica más utilizada actualmente.

· **Tablas de decisión:** En una tabla son representadas las posibles condiciones del problema con sus respectivas acciones. Suele ser una técnica de apoyo al pseudocódigo cuando existen situaciones condicionales complejas.

**Debes conocer**

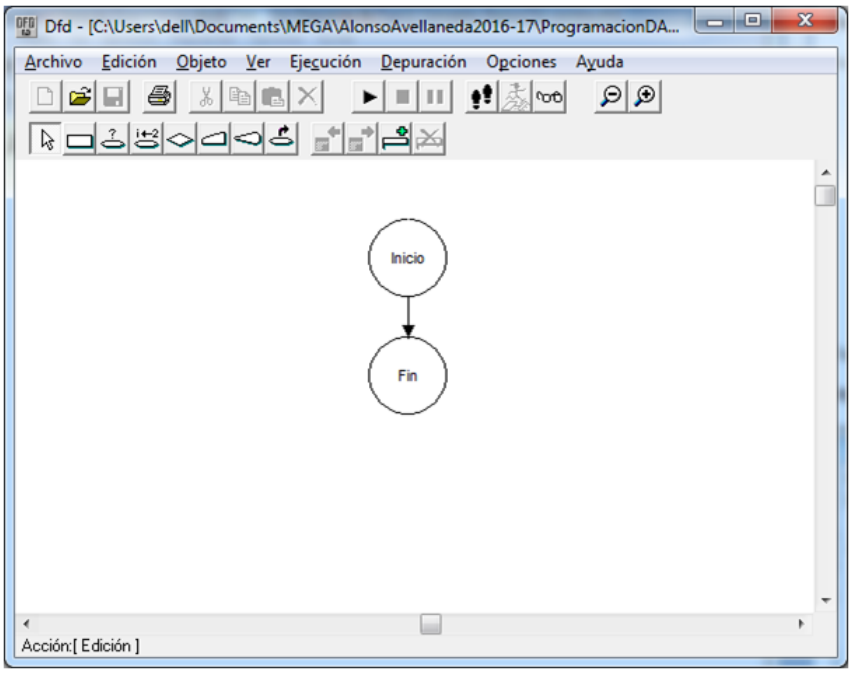
A continuación te ofrecemos dos enlaces muy interesantes:

- En el primer enlace puedes ver los elementos gráficos fundamentales que se utilizan para la generación de diagramas de flujo.

https://www.heflo.com/es/blog/modelado-de-procesos/significado-simbolos-diagrama-flujo/

· El segundo recurso, consiste en una aplicación que permite construir diagramas de flujo para la resolución de tareas mediante algoritmos. El programa se llama DFD, y se puede obtener de https://goo.gl/8M9g5c

Se trata de una aplicación portable. Tras su descarga, simplemente lanzamos el ejecutable y veremos la siguiente ventana. Se tratará de incorporar comandos de la barra de herramientas superior al panel de trabajo:

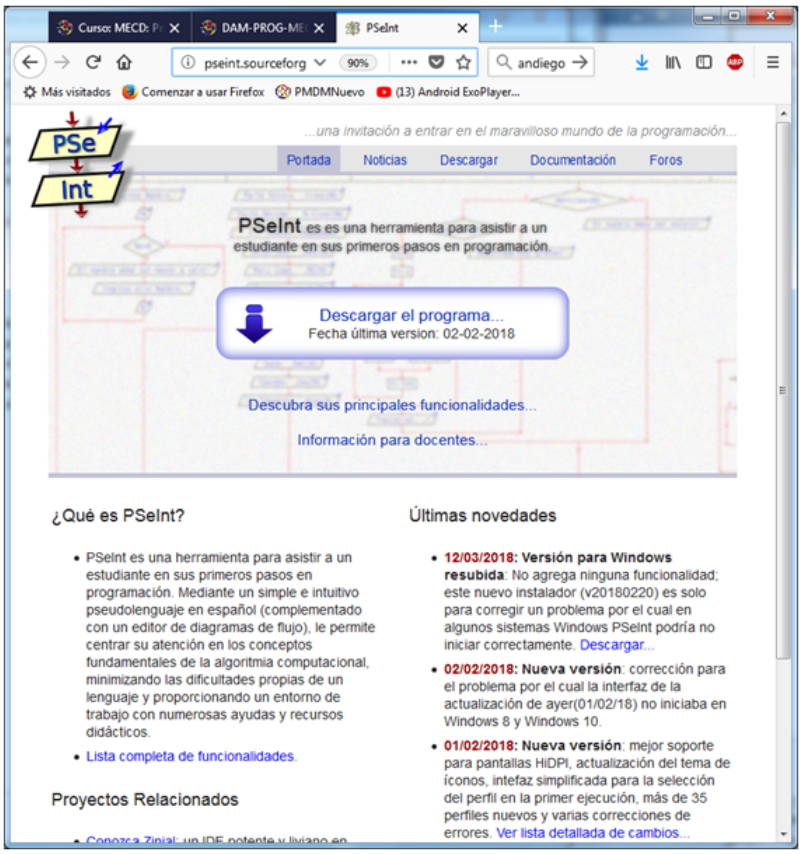



Podemos ver un video sobre cómo crear un algoritmo con esta herramienta:

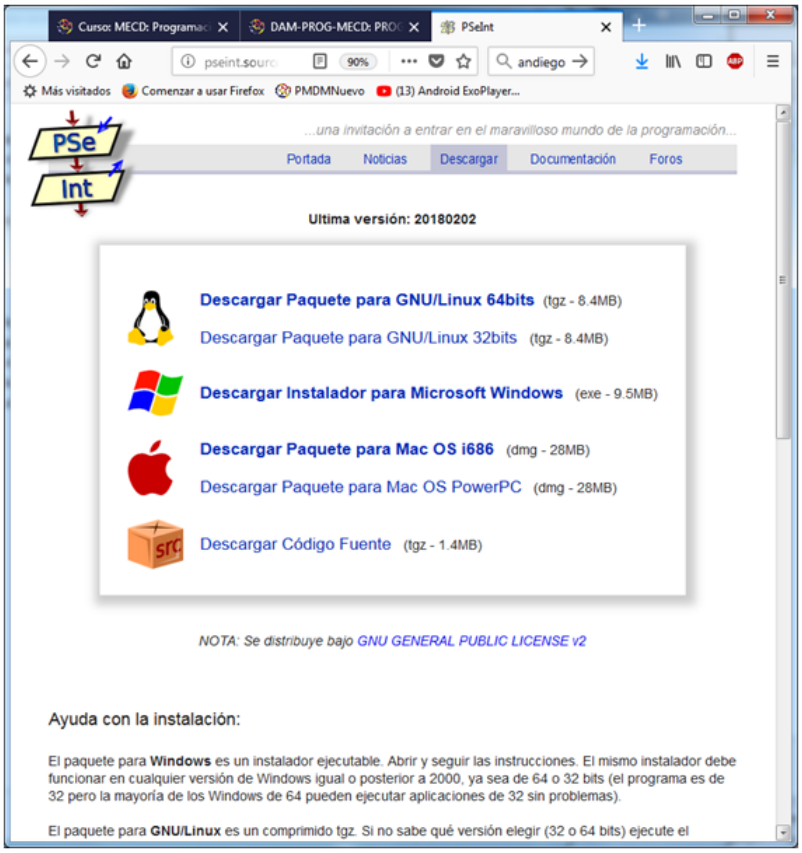
- El tercer recurso permite la construcción de un diagrama de flujo con otra herramienta gráfica y su transformación a pseudocódigo. Se trata del programa PSeInt:

http://pseint.sourceforge.net/

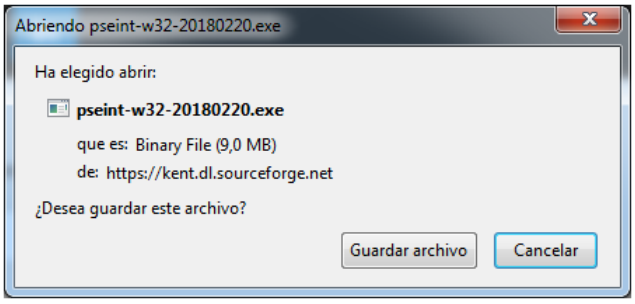
Veamos aquí su instalación y puesta en funcionamiento:



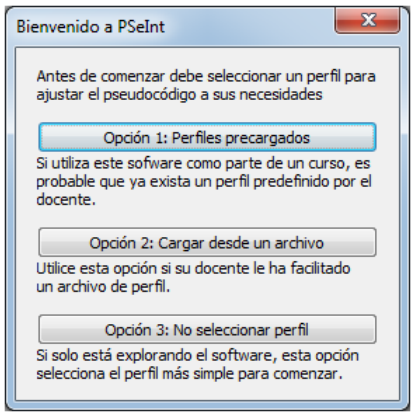

Vamos a "Descargar el programa...":



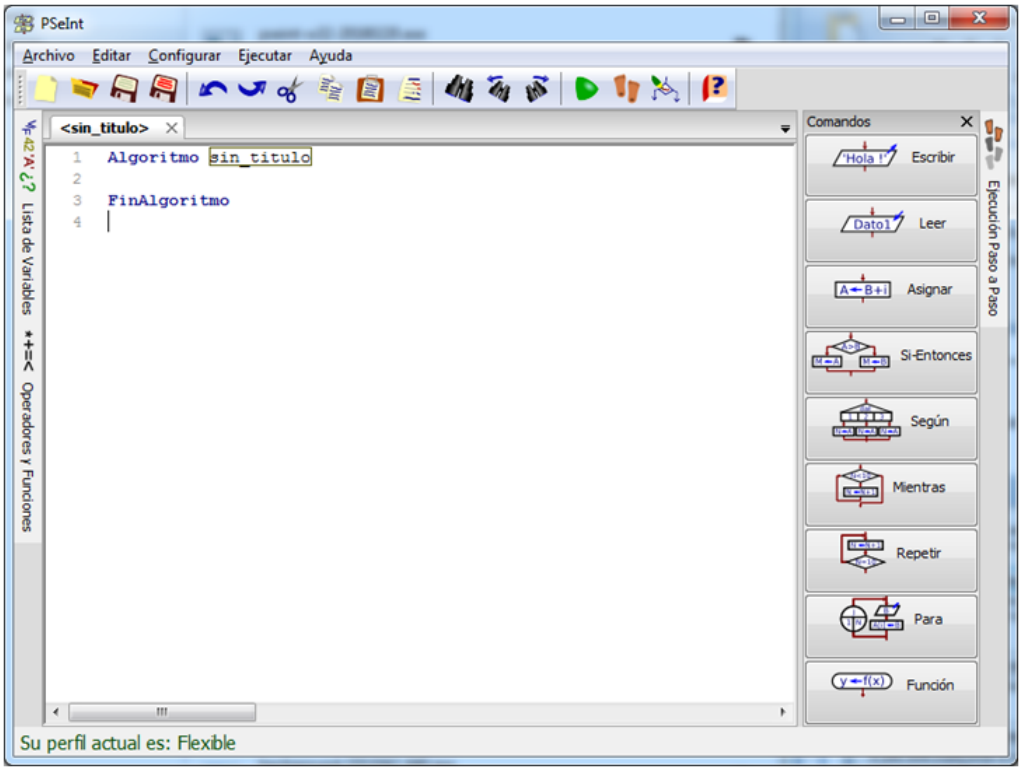

Si por ejemplo instalamos la versión para Windows:




Procedemos a su instalación, de manera básica. Si lo ejecutamos por primera vez aparecerá la ventana:



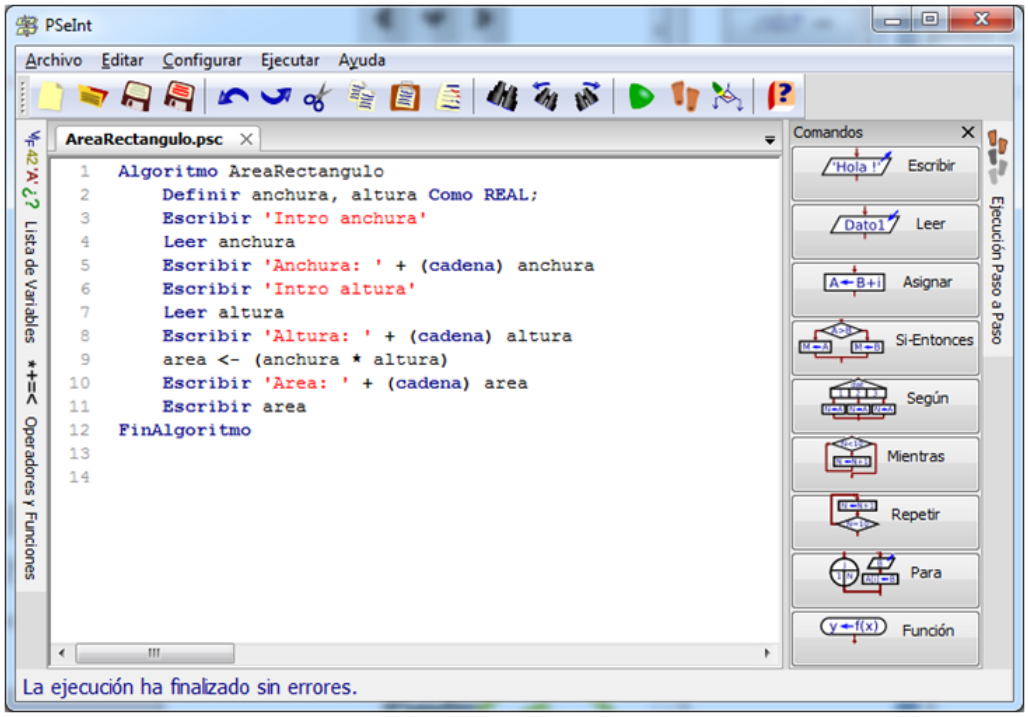

Simplemente seleccionamos la "Opción 3: No seleccionar perfil", tras lo cual, ya se cargará el entorno:




Como vemos, dispondremos de 2 paneles principales, el de la izquierda para incorporar pseudocódigo directamente, y el de la derecha, con los comandos disponibles para ayudar, si se desconoce la sintaxis en la escritura de instrucciones o estructuras de control.

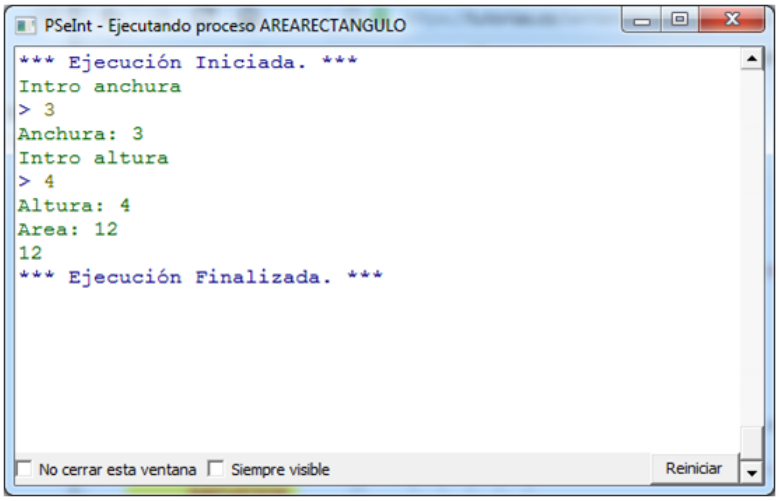
Para mayor flexibilidad en el lenguaje: Menú Configurar --> Opciones del Lenguaje (perfiles)... --> Elegir "Flexible"

Podemos ver cómo crear un sencillo algoritmo del cálculo del área de un cuadrado:




En dicho algoritmo, quizá lo más significativo sea la adaptación o "casteo" de tipos, para poder concatenar los números considerados como cadenas con cadenas de texto.

La salida resultante de este algoritmo la obtenemos al dar al icono de reproducir verde de la ventana anterior apareciendo la siguiente ventana:




## Autoevaluación

**Rellena los huecos con los conceptos adecuados:**

A los pasos que permiten resolver el problema, escritos en un lenguaje de programación, para que puedan ser ejecutados en el ordenador y así obtener la solución, se les denomina: programa .

Resolver