"1º parte" → código fuente

Programas que traducen código de lenguaje de alto nivel en lo que se puede ejecutar en compiladores.

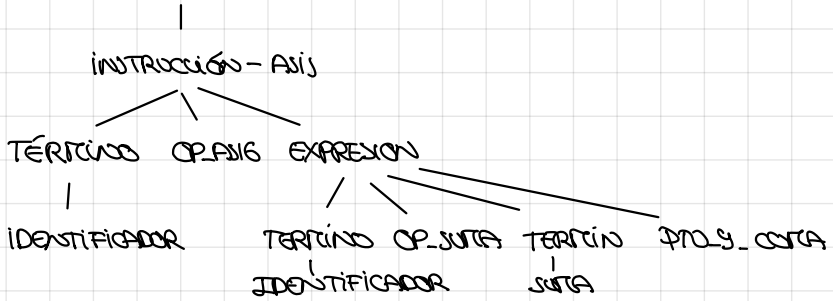
1ª etapa → análisis léxico (encuentra cuales son los caracteres).

`a = a+1;`
- IDENTIFICADOR (A)
- OP_ASIGNACIÓN (=)
- IDENTIFICADOR (A)
- OP_SUMA (+)
- ENTERO (1)
- PTO_Y_COMA (;)

⇒ TOKEN O CATEGORÍAS LÉXICAS.

2ª etapa → análisis sintáctico (usa lenguajes independientes del contexto → AUTÓMATAS CON PILAS)

Posible representación → INSTRUCCIÓN



3ª etapa → análisis semántico (coger el árbol y generar estructura análoga, especificando lo que quiere hacer ese código).

CHEQUEO DE TIPOS →
- los valores que se asignan son de tipos correctos
- sobrecarga de operadores, el compilador tiene que determinar que función tiene que hacer el operador.
- asignación automática de tipos.

→ COMPROBACIÓN ÁMBITO (TABLA SÍMBOLOS).

4ª etapa → generación de código intermedio (lenguaje a mitad de camino entre código fuente y código máquina).

CÓDIGO FUENTE	CÓDIGO INTERMEDIO	CÓDIGO MÁQUINA
+ complejidad + capacidad expresiva		- complejidad - capacidad expresiva

MULTICOMPILADORES ⇒ · windows para C
· pascal para linux
· fortran para los.

5ª etapa → optimización de código (sustitución de operadores complejos por más básicos).
↳ se realiza sobre código intermedio, con menor consumo de recursos. $a = a * 2 \Rightarrow a = a + a$

6ª etapa → · generación (+optimización) de código máquina → pasamos las instrucciones de código intermedio o código máquina.
· al generar el código, tenemos que tener en cuenta las características
· dependen de la arquitectura de la máquina.

* Para reconocer los tokens del nivel léxico se usan lenguajes regulares (Autómatas finitos)

GENERADORES DE ANALIZADORES: → LEX (LÉXICOS) → FLEX (lenguajes regulares)
→ YACC (SINTÁCTICOS) → BISON
} herramientas que vamos a usar.
sensibles al contexto

En nuestro caso no vamos a fijar en las partes léxicas y sintácticas ⇒ utilizando algoritmos para poder hacerlo.

PRÁCTICA 2 ⇒ FLEX. → dado un fichero de texto, da un archivo ejecutable.