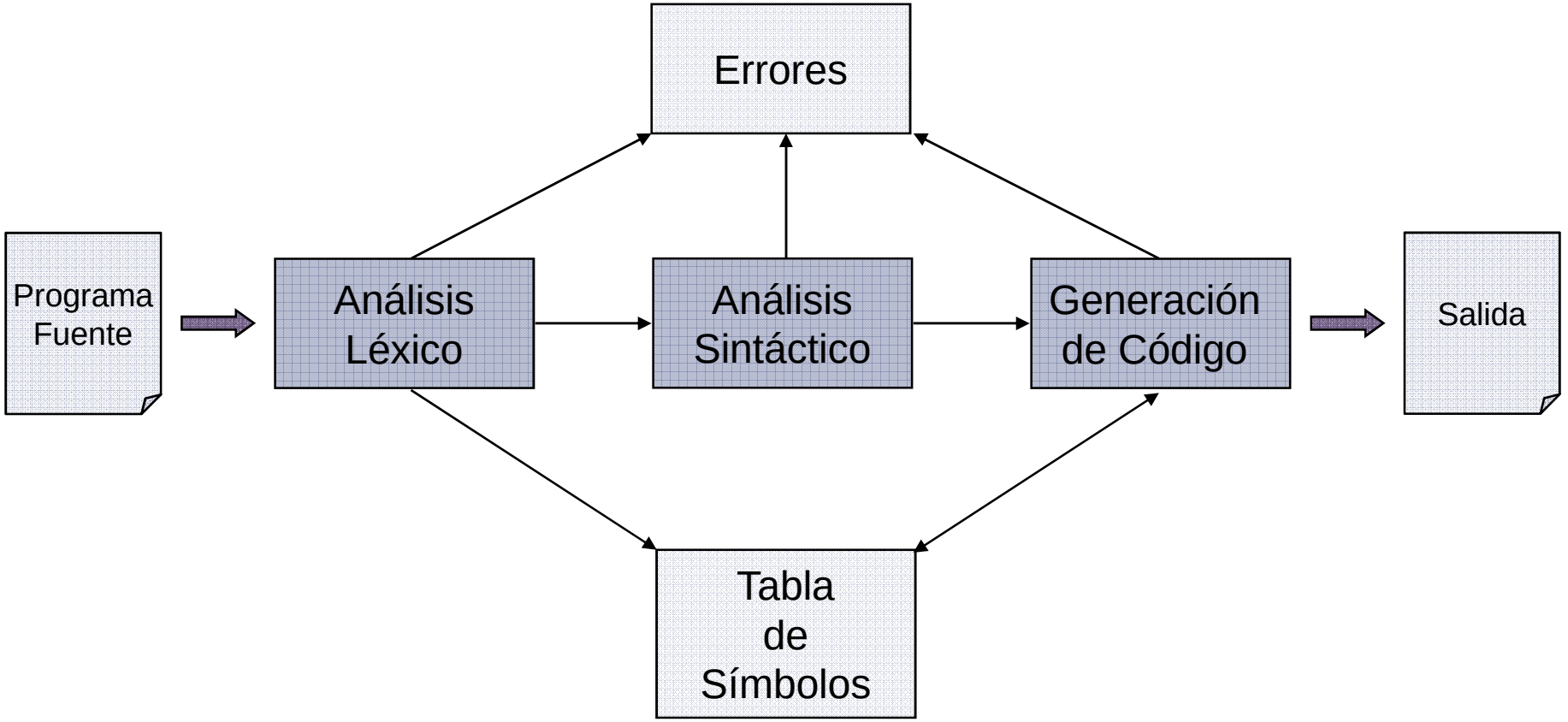
Diseño de Compiladores I

Tabla de Símbolos - Errores

Fases de la Compilación



Tablas de Símbolos

Tabla de Símbolos

Es una estructura de datos que contiene un registro para cada identificador utilizado en el código fuente, con campos que contienen información relevante para cada símbolo (atributos).

Tabla de Símbolos

- Cuando el Análisis Léxico detecta un token de tipo identificador, lo ingresa en la Tabla de Símbolos.
- Durante la Generación de Código se ingresa información para los atributos de los símbolos, y se usa esa información de diversas maneras.
- Durante la Generación de Código puede ser necesario incorporar nueva información a la Tabla de Símbolos.

¿Qué pasa con la Tabla de Símbolos?

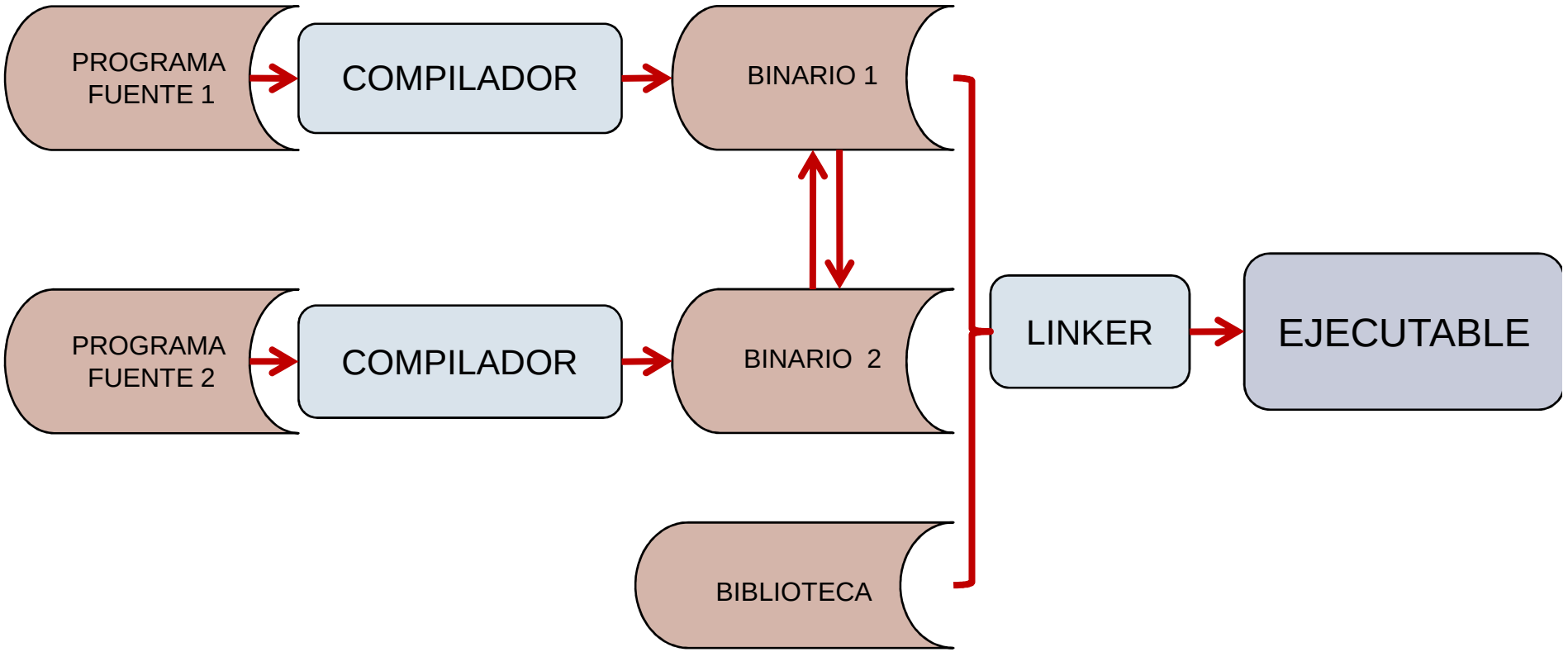
- La Tabla de Símbolos perdura en tiempo de ejecución para los intérpretes.
- La Tabla de Símbolos muere en tiempo de ejecución para los compiladores.

Tabla de Símbolos
Tiempo de vida

- La Tabla de Símbolos perdura en tiempo de ejecución para los intérpretes.
- La Tabla de Símbolos muere en tiempo de ejecución para los compiladores.
  - ***FALSO si hay compilaciones separadas.***

Tabla de Símbolos
Tiempo de vida

Compilaciones separadas



Durante la vinculación, se necesita la Tabla de Símbolos debido a las referencias entre un Binario y otro

Tabla de Símbolos
Tiempo de vida

Compilaciones separadas

- En los programas fuentes existen elementos que no se conocen porque están definidos en otros fuentes.
- Estructura de un binario:

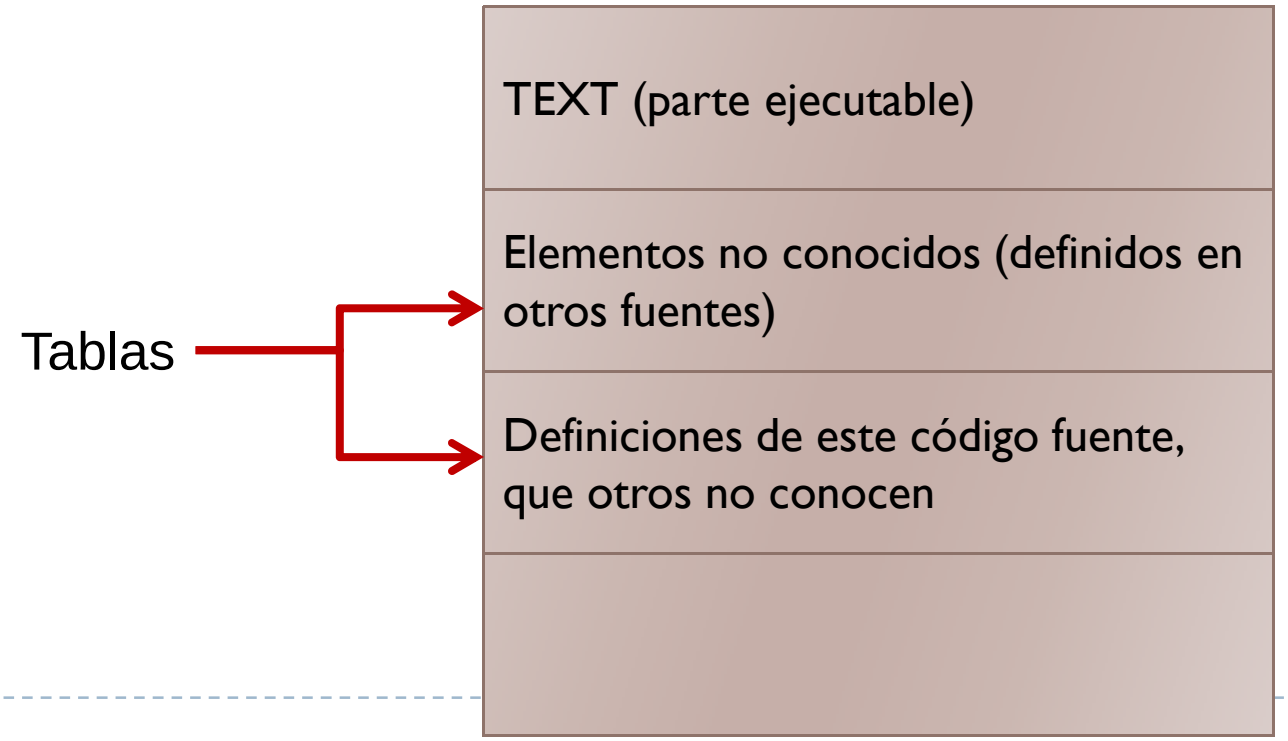


Tabla de Símbolos
Tiempo de vida

Compilaciones separadas

- Las tablas son pedazos de la Tabla de Símbolos.
- El Linker completa las referencias cruzadas, borra las tablas y produce la salida a partir del TEXT.

- La Tabla de Símbolos muere al finalizar la vinculación.
  - ***FALSO si hay vinculación dinámica***

Tabla de Símbolos
Tiempo de vida

Vinculación dinámica

- Se efectúa en tiempo de ejecución.
- La Tabla de Símbolos muere al finalizar la carga de las dll's o elf's.
  - ***FALSO si hay debugging.***

Tabla de Símbolos
Tiempo de vida

Entonces …

- ¿Qué diferencia existe en el uso de la Tabla de Símbolos entre Compiladores e Intérpretes?
  - Los intérpretes necesitan la Tabla de Símbolos durante la ejecución.
  - Los programas compilados no necesitan la Tabla de Símbolos. La usa el Debugger.

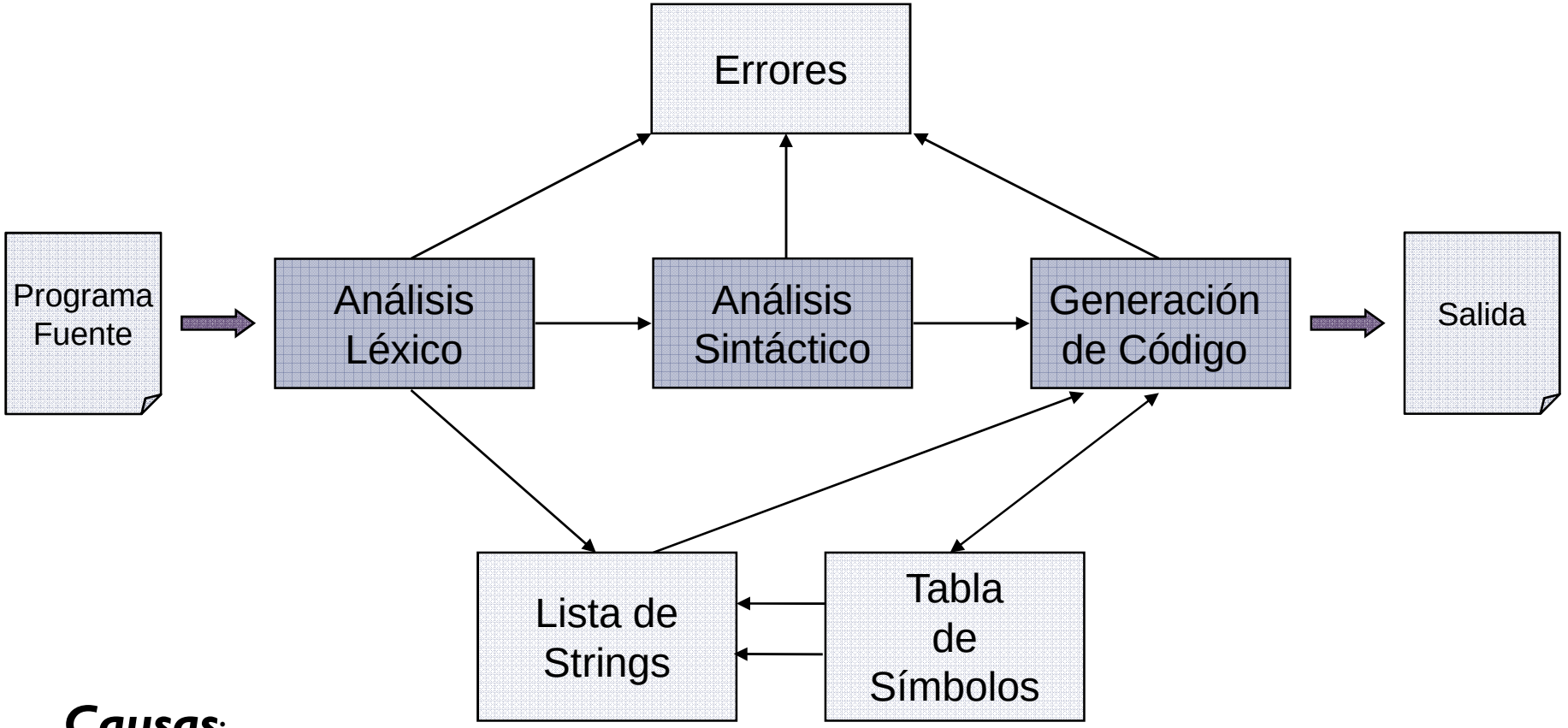
Tabla de Símbolos
Estructura y contenidos

- Si un nombre puede tener diferentes usos:
  - ¿En qué momento de la Compilación se detecta esto?
  - ¿Cómo se refleja en la Tabla de Símbolos?

- ¿ENTONCES?

  - El Analizador Léxico crea lista de Strings con los nombres de los símbolos.
  - Las entradas en la Tabla de Símbolos asociadas a cada nombre, se generan en etapas posteriores.

Fases de la Compilación



Causas:

- Diferentes espacios de nombres
- Ámbitos

Tabla de Símbolos
Estructura y contenidos

- Cada entrada contiene
  - Nombre (lexema)
  - Atributos
- Atributos (varían con el uso del símbolo):
  - Tipo
  - Uso: procedimiento, método, variable, etiqueta, arreglo
  - Posición de almacenamiento
  - (Se podría guardar también el tipo de token)

Tabla de Símbolos
Estructura y contenidos

- Según el uso, variarán los atributos:
  - Variable:
    - Lexema
    - Tipo
  - Matriz
    - Lexema
    - Tipo de los elementos
    - Dimensiones

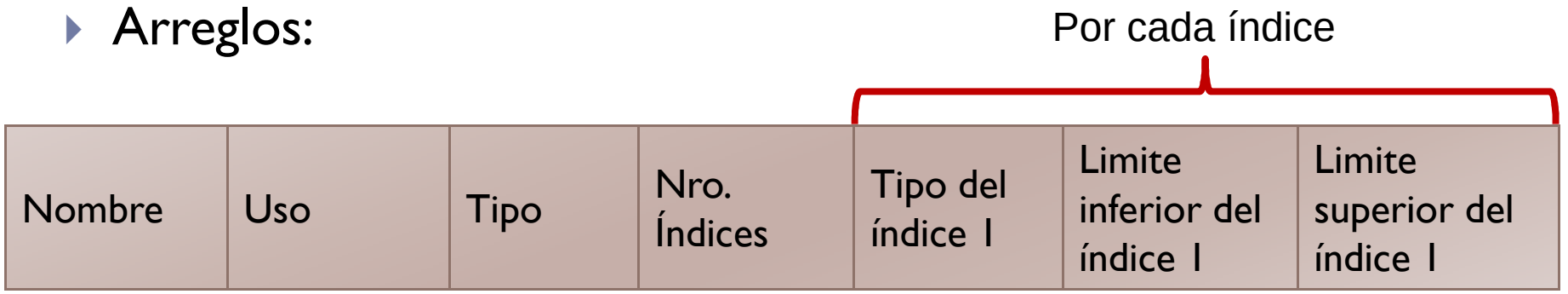
Tabla de Símbolos
Estructura y contenidos

- Según el uso, variarán los atributos:
  - Nombre de Procedimiento
    - Lexema
    - Parámetros formales
  - Nombre de Función
    - Lexema
    - Parámetros formales
    - Tipo devuelto
  - Tipo definido por el usuario
    - Lexema
    - Definición del tipo

Tabla de Símbolos
Estructura y contenidos

Ejemplos:

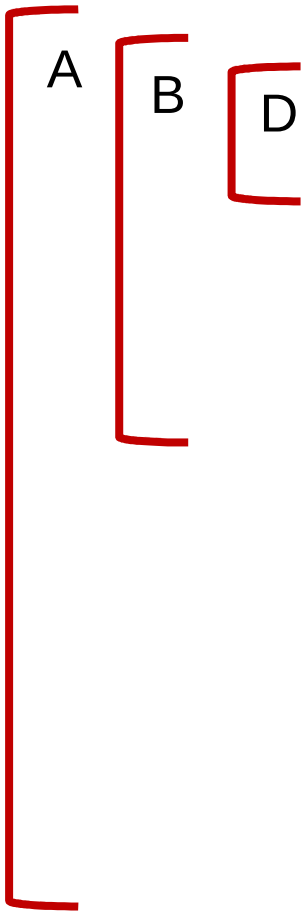
- Arreglos:



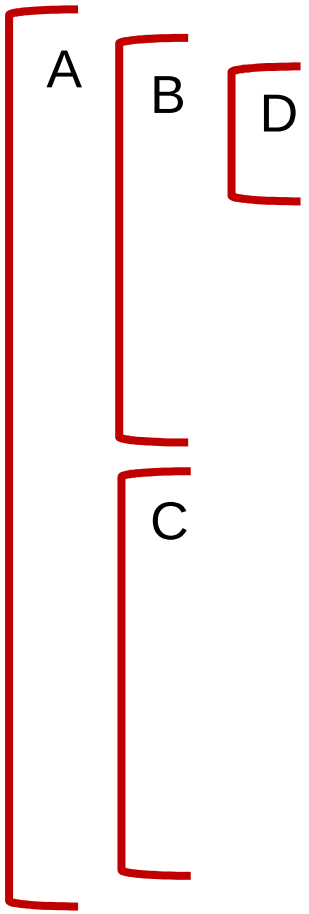
- Funciones:

- etc.

Tabla de Símbolos. Ámbitos



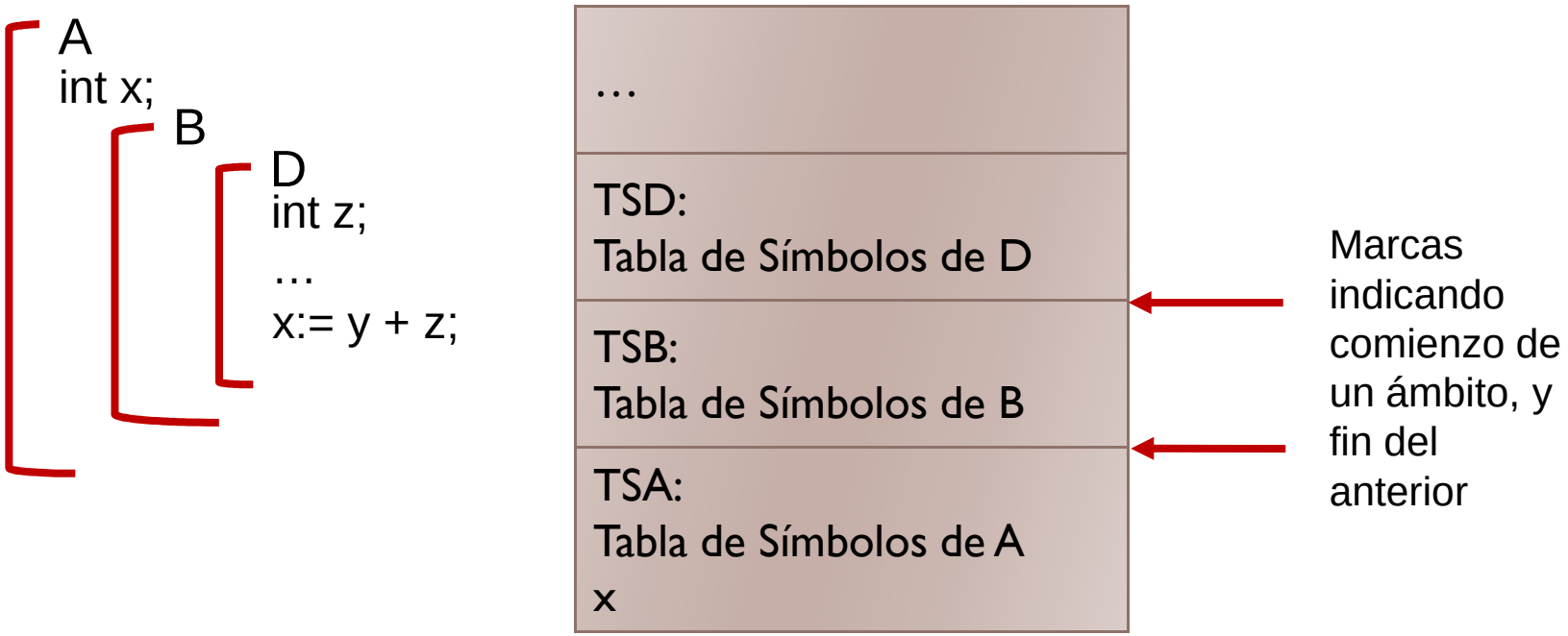
Situación 1: Ámbitos anidados



Situación 2: Ámbitos hermanos

Tabla de Símbolos. Ámbitos

Situación 1: Ámbitos anidados



Buscar x en TSD,

No está → Contador = 1

→ Buscar en TSB

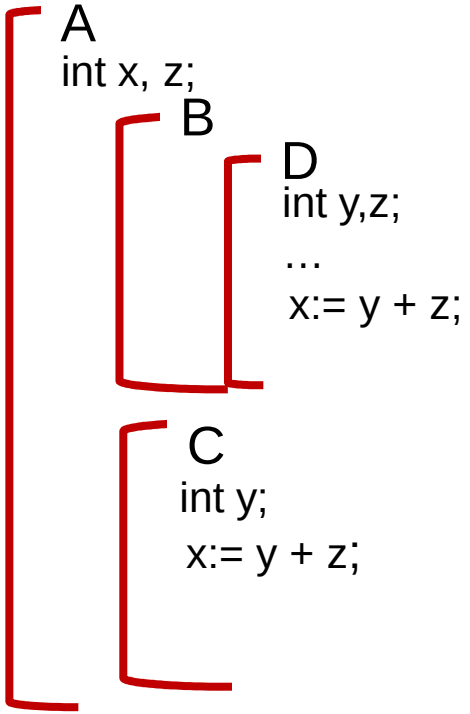
No está → Contador = 2

→ Buscar en TSA

- No se necesita construir el árbol de anidamiento.
- El valor en el contador (número de límites que se cruzaron) indica la posición en el árbol de anidamiento.
- Se pueden borrar las tablas cuando ya no se necesitan.

Tabla de Símbolos. Ámbitos

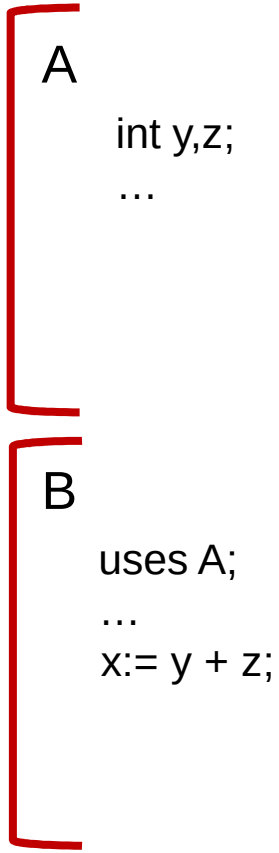
Situación 2: Ámbitos hermanos



…
TSD: Tabla de Símbolos de D y z
TSB: Tabla de Símbolos de B
TSA: Tabla de Símbolos de A x z

…
TSC: Tabla de Símbolos de C y
TSA: Tabla de Símbolos de A x z

Tabla de Símbolos. Ámbitos



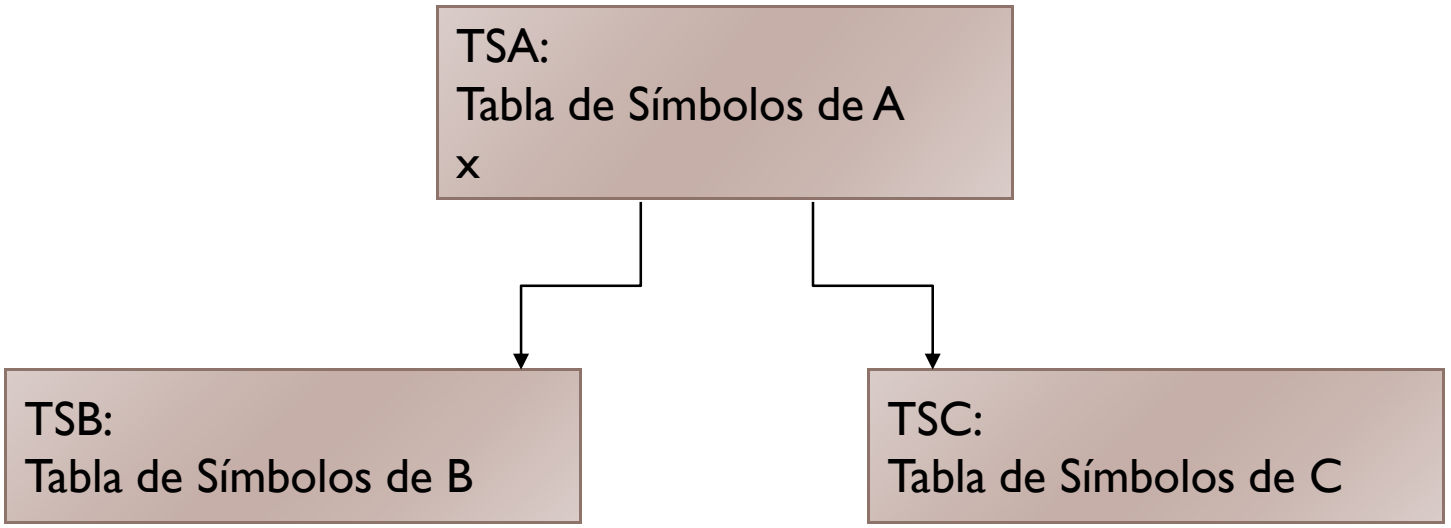
Situación 3: Las declaraciones están en una unidad diferente

Tabla de Símbolos. Ámbitos

Situación 3

```
A
int x;

B
uses A;

x:= y + z;

C
uses A;

x:= 5;
```

- Se utilizan Tablas de Símbolos enlazadas.



- La búsqueda se hace desde las hojas a la raíz.
- No se pueden borrar las TdeS.

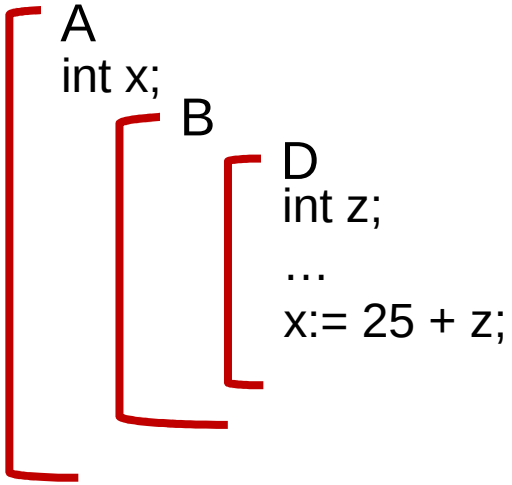
Tabla de Símbolos. Ámbitos

- Problema: En la vinculación, la estructura es secuencial, no es posible manejar árboles.

- Solución: Name Mangling (Name decoration)
  - Se renombran los símbolos, incluyendo información de ámbitos.

Tabla de Símbolos. Ámbitos

Situación 1: Ámbitos anidados



x ⟹ x:A

z ⟹ z:A:B:D

Buscar x:A:B:D

No está → Buscar x:A:B

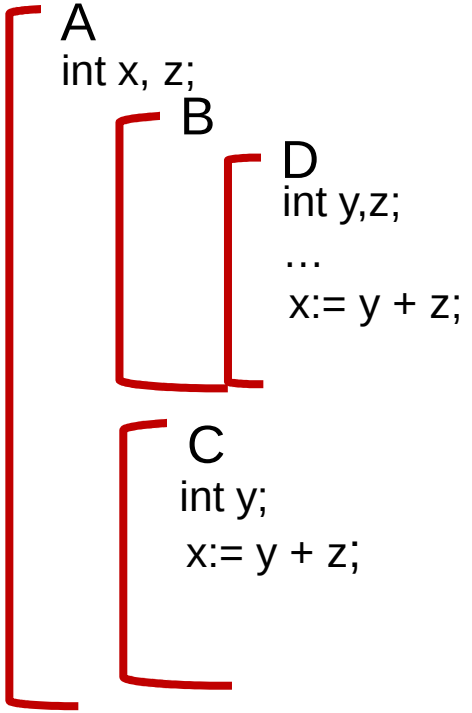
No está → Buscar x:A ✓

Buscar z:A:B:D ✓

- Habrá una única Tabla de Símbolos
- Se requiere una variable global que indica el ámbito actual
  - Al entrar a un ámbito, se concatena el nuevo ámbito
  - Al salir de un ámbito, se elimina de la variable de ámbito.

Tabla de Símbolos. Ámbitos

Situación 2: Ámbitos hermanos



x → x:A

z → z:A

y → y:A:B:D

z → z:A:B:D

y → y:A:B:C

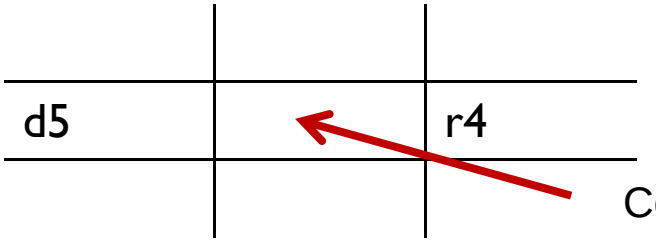
Tratamiento de errores

Manejo de Errores

- Cada una de las etapas del Compilador puede detectar errores que son informados al programador.
- Un buen compilador no debería terminar su ejecución al detectar un error, sino que debería recuperarse y continuar con la compilación.

Errores

- Detectados durante el Analizador Léxico
  - Carácter no válido
  - Constante fuera de rango
- Detectados durante el Analizador Sintáctico
  - Falta operando
  - Dos operadores juntos
  - Falta paréntesis de cierre



Celda vacía
→ Error

- Detectados durante la Generación de Codigo
  - Variable no declarada
  - Variable redeclarada
  - Incompatibilidad de tipos en una expresión

Qué hacer ante un error?

- La compilación no debe terminar ante un error.

- Técnicas de recuperación
  - Pánico o Resincronización
  - Inserción / Borrado / Reemplazo
  - Gramática de error

Técnicas de Recuperación

- Pánico o resincronización
  - No se conoce la causa del error
  - No se conoce solución

→ Descartar elementos hasta un carácter de sincronización

Técnicas de Recuperación

- Inserción / Borrado / Reemplazo
  - Ejemplo: x := y + * z;
    - Inserción: Agregar un I entre el + y el *
    - Borrado: Borrar el *
    - Reemplazo: Reemplazar el * por un blanco
  - Se trata de un "parche", que no necesariamente es lo correcto

Técnicas de Recuperación

- Gramática de error
  - Predice posibles errores

  Ejemplo:

  1. T → T * F
  2. T → T * / F // Error 1
  3. T → T * * F // Error 2
  4. T → T F // Error 3

  - La cantidad de combinaciones permitidas es mucho menor que la de combinaciones prohibidas.
  - No se pueden considerar todos los posibles errores.
  - Puede generar conflictos

    → ***Usar sólo para errores típicos***

  - Se pueden combinar con Inserción / Borrado / Reemplazo

Técnicas de Recuperación

	Análisis Léxico	**Anáilisis Sintáctico**	**Generación de Código**
I / B / R	Mucho	-	Mucho
Pánico / Resincronización	-	Mucho	-
Gramática de Error	Algo	Algo	-

Técnicas de Recuperación
Consecuencias

- Pánico
  - Se informa como Error o Error Fatal
  - Se suspende la generación de código
- Inserción / Borrado / Reemplazo
  - Se informa como "Warning"
  - El compilador entiende que el "parche" es correcto
  - Se puede generar código
- Gramática de error
  - Dependiendo del tratamiento, se puede tratar como
    - Error
    - Warning

Nota: En los compiladores actuales, el uso de Warnings se ha extendido a la información de desprolijidades en el código, como:

"Variable declarada y no usada"

¿PREGUNTAS?