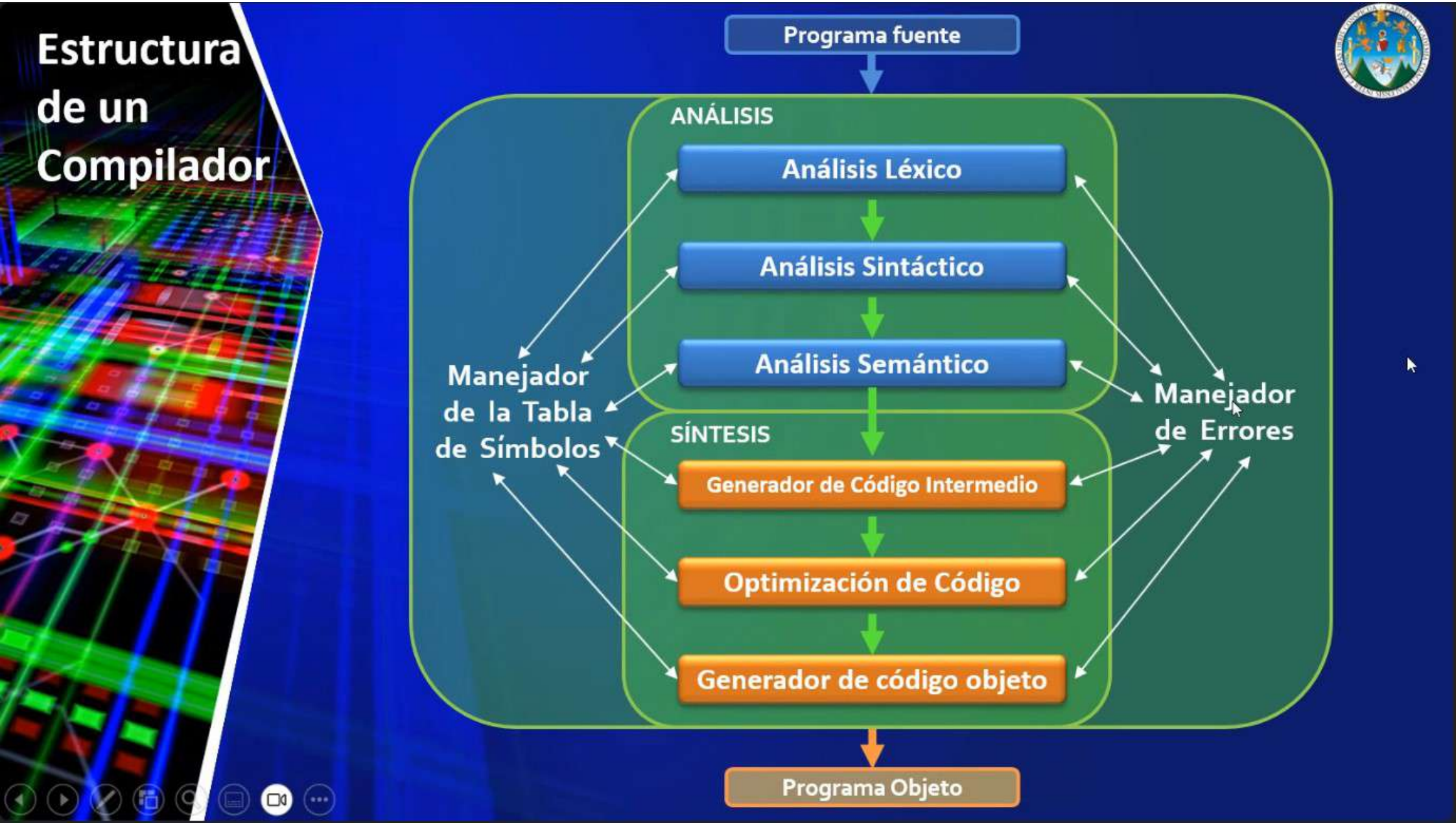
Estructura de un Compilador



Estructura de un Compilador

Análisis Léxico o de exploración (Scanner):

El analizador léxico lee el flujo de caracteres que componen el programa fuente y agrupa los caracteres en secuencias significativas.

Componente Léxico o Token, es una secuencia de caracteres que tienen un significado colectivo.
Al conjunto de cadenas de la entrada para la cual se devuelve el mismo token, se les conoce como **lexemas**, éstos deben cumplir con la regla (**patrón**) asociada al componente léxico.

Por lo tanto, un **lexema** es una secuencia de caracteres en el programa fuente con la que concuerda con el **patrón** de un **componente léxico**.

Estructura de un Compilador

Análisis Léxico o de exploración (Scanner):

Por ejemplo:

Precio: ent
costo: dec
Precio = costo + 35.4

Token	Lexema	Patrón (informal)
Identificador	Precio, costo	Letra seguida de letras o dígitos
Asignación	=	=
Operador	+	+ o – o * o /
Decimal	35.4	Uno o más dígitos, punto, uno o más dígitos
DosPtos	:	:
TipoDec	dec	dec
TipoEnt	ent	ent

Estructura de un Compilador

Manejador de la Tabla de Símbolos:

Registra los nombres de las variables que se usan en el programa fuente y recopila la información de sus atributos.

Estos atributos pueden proporcionar información sobre el almacenamiento, nombre, tipo, alcance, etc. También almacena los procedimientos o funciones, con sus respectivos datos.

La tabla de símbolos es una estructura de datos que contiene un registro para cada variable; diseñada para que el compilador encuentre los datos rápidamente

Precio: ent
costo: dec
Precio = costo + 35.4

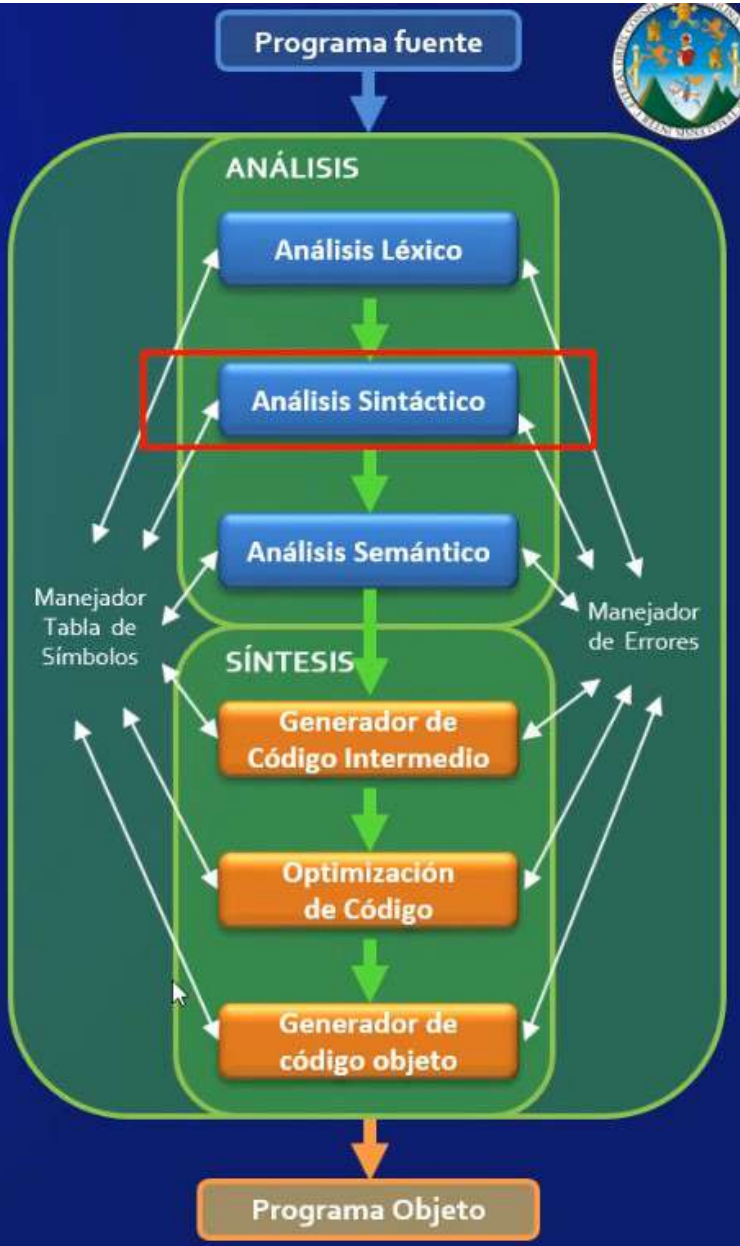
#	Token	Variable	Tipo	Valor	...
1	Identificador	Precio	ent	0	
2	Identificador	costo	dec	0	

Estructura de un Compilador

Análisis Sintáctico o Jerárquico (Parser):

Es donde se agrupan los componentes léxicos de forma jerárquica en colecciones anidadas con un significado colectivo. Es decir, es la fase donde se revisa en qué orden deben venir los componentes léxicos.

Precio = costo + 35.4



Estructura de un Compilador

Análisis Semántico:

Se realizan ciertas revisiones para asegurar que los componentes de un programa se ajustan de un modo lógico significativo.

Precio: ent
costo: dec
Precio = costo + 35.4

➔ Error Semántico: precio es entero y la asignación genera un valor decimal.

Tabla de Símbolos:

#	Token	Variable	Tipo	Valor	...
1	Identificador	Precio	ent	0	
2	Identificador	costo	dec	0	

Estructura de un Compilador

Generador de código intermedio:

Depende del compilador, puede tener una o más representaciones intermedias. Debe ser fácil de producir y fácil de traducir a la máquina destino.

Código fuente:

Precio: dec
costo: dec
Precio = costo + 35.4

Código intermedio:

```
T1 = decimal (35.4)
T2 = Id2 + T1
Id1 = T2
```

Estructura de un Compilador

Optimización de código:

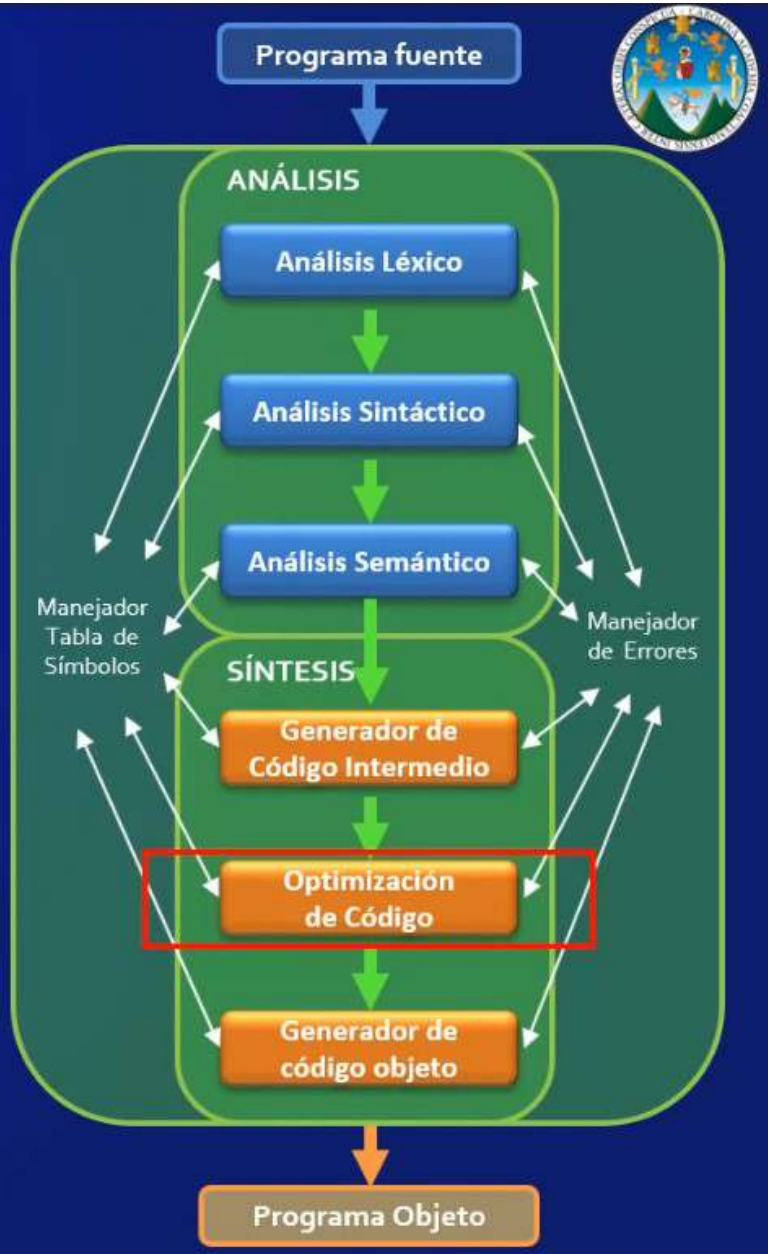
En esta fase, se busca mejorar el código, puede ser que tenga diferente objetivo, dependiendo del compilador; por ejemplo: que se ejecute más rápido, que el código sea más corto, que consuma menos energía, que se omita código que nunca se usa o que se omitan los comentarios.

Código intermedio:

```
T1 = decimal (35.4)
T2 = Id2 + T1
Id1 = T2
```

Código optimizado:

```
T1 = Id2 + 35.4
Id1 = T1
```



Estructura de un Compilador

Generador de código objeto:

Es la fase donde finalmente se genera el código en el lenguaje de bajo nivel que entienda la máquina. En este ejemplo se genera en Lenguaje Ensamblador.

Código optimizado:

```
T1 = Id2 + 35.4
Id1 = T1
```

Código Objeto:

```
LDF R1, id2
ADDF R1, R1, #35.4
STF Id1, R1
```