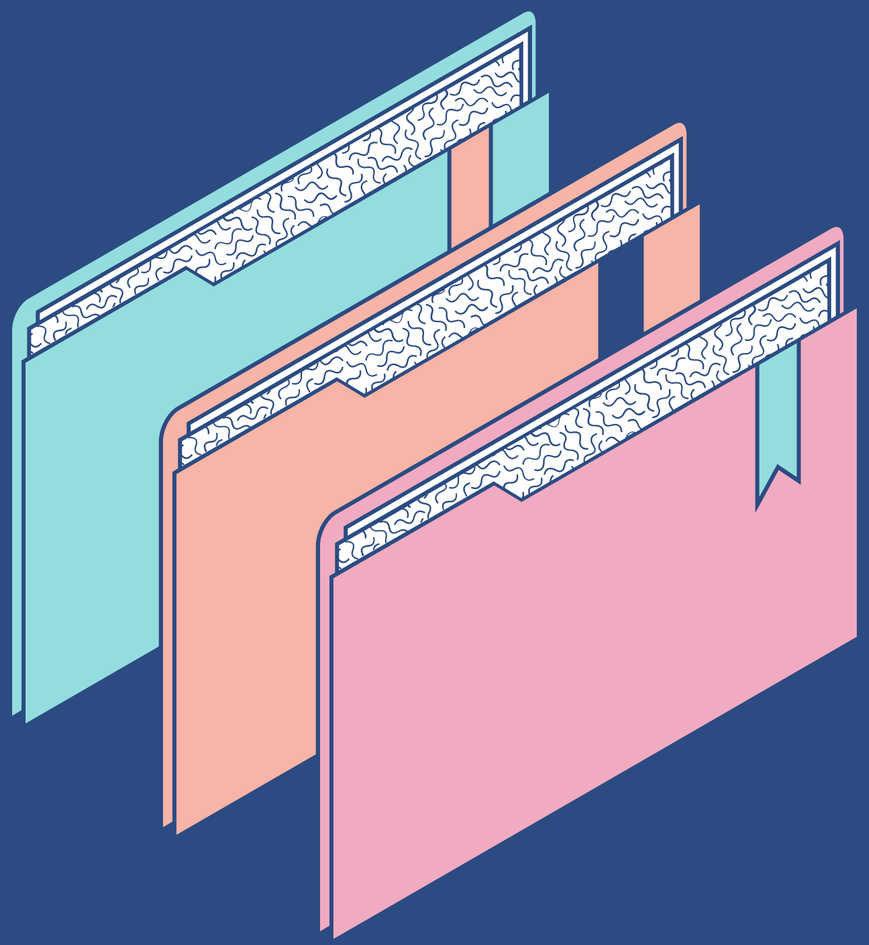
PRIMER SEMESTRE - 2023

Organización de Lenguajes y Compiladores 2

Clase 1 - Repaso



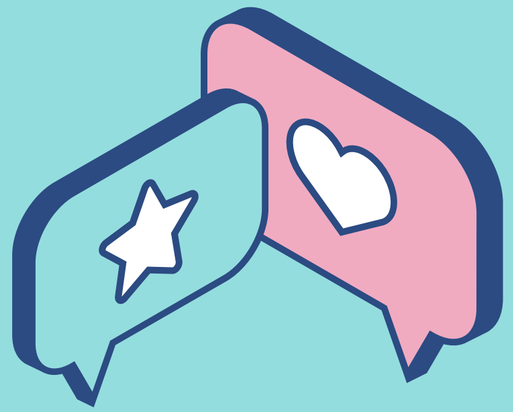
Agenda

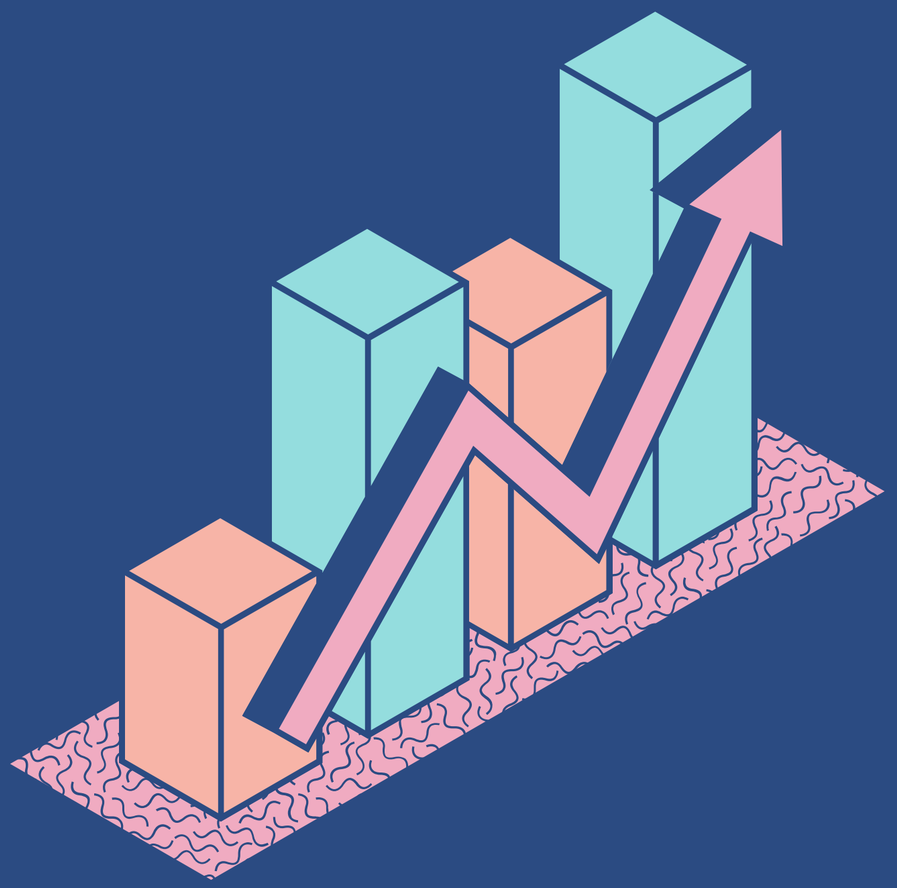
TEMAS CLAVE QUE SE DEBATIRÁN EN ESTA PRESENTACIÓN

- Presentación y bienvenida
- Encuesta
- Programa del laboratorio
- Repaso
- Actividad del día

Presentación y bienvenida

- ¿Qué esperan del curso de Compi 2?
- ¿Por qué es tan "difícil"?
- ¿Es útil la clase?
- ¿Que necesito para iniciar?

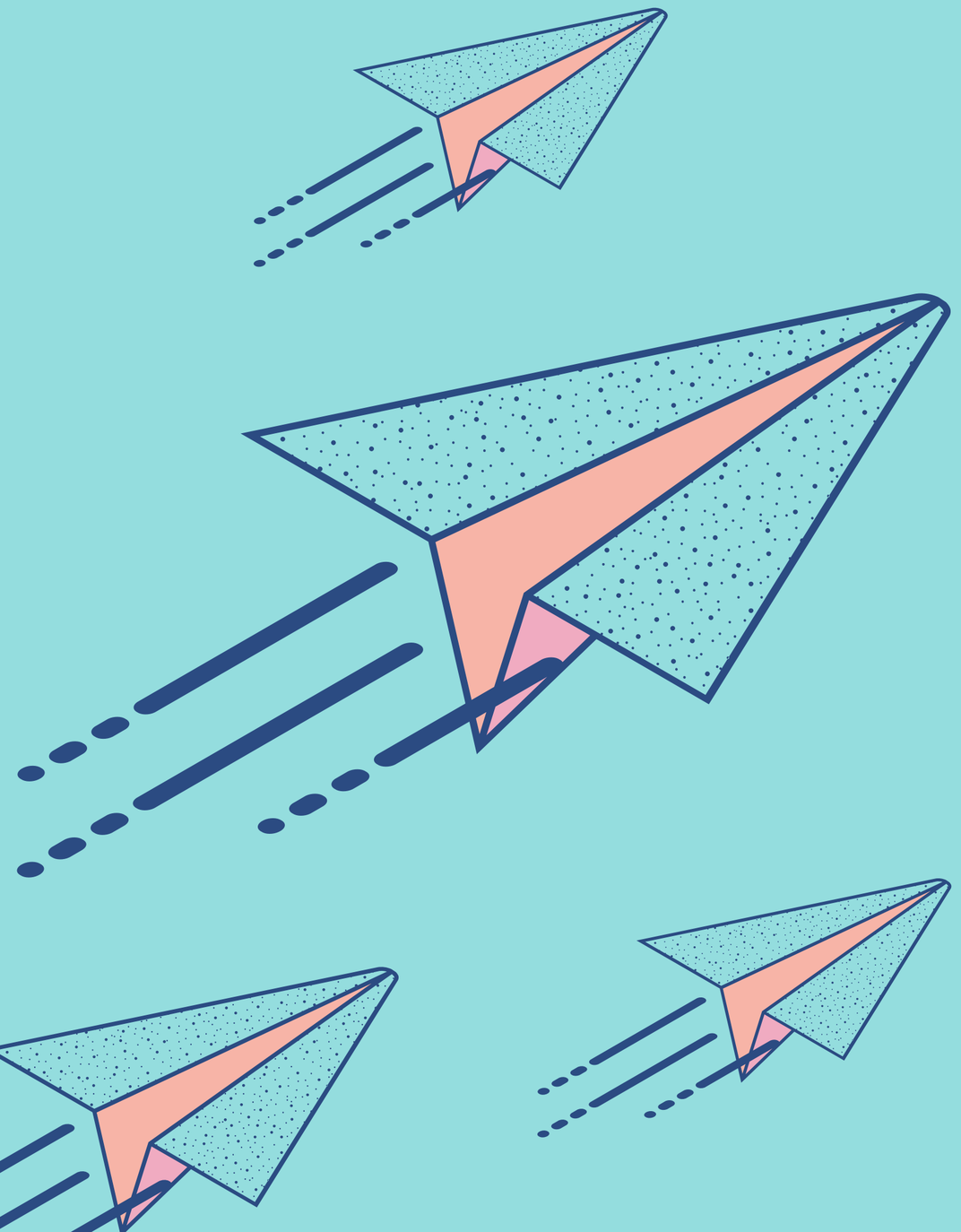


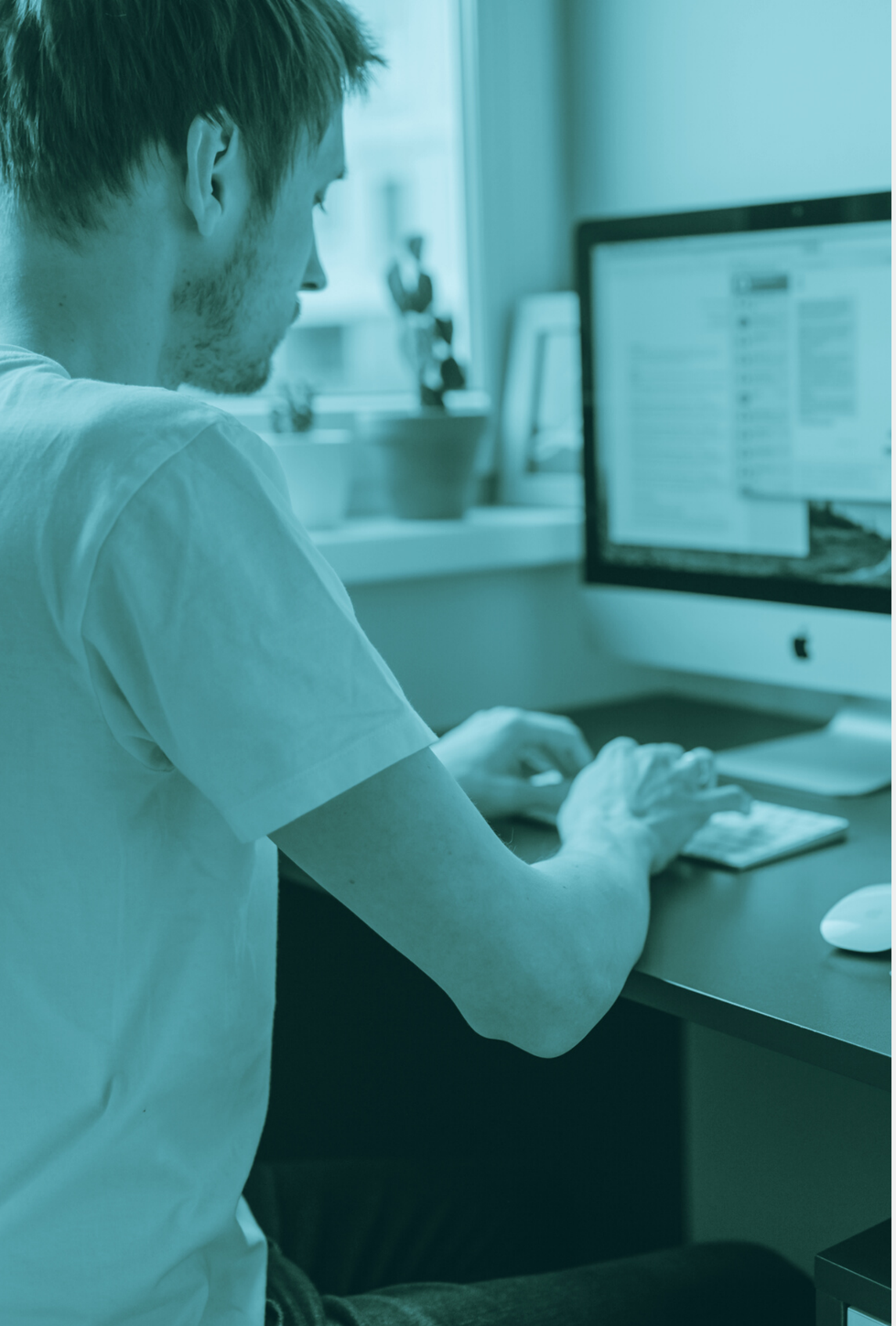


Encuesta

- ¿Primera vez?
- ¿Sé qué es un analizador Léxico?
- ¿Sé qué es un analizador Sintáctico?
- ¿Sé qué es un analizador Semántico?
- ¿Sé qué es C3D?
- ¿He utilizado Jison?
- ¿He utilizado Flex/Cup?
- ¿He utilizado Irony?
- ¿He utilizado Antlr?
- ¿He programado en R?

Programa del Laboratorio



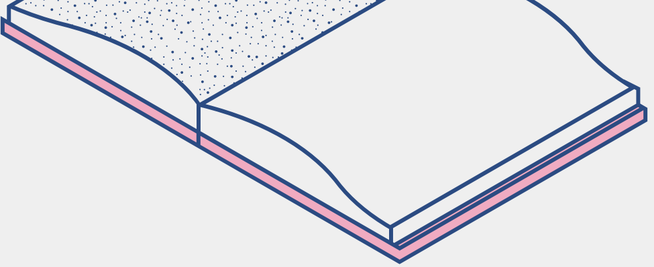


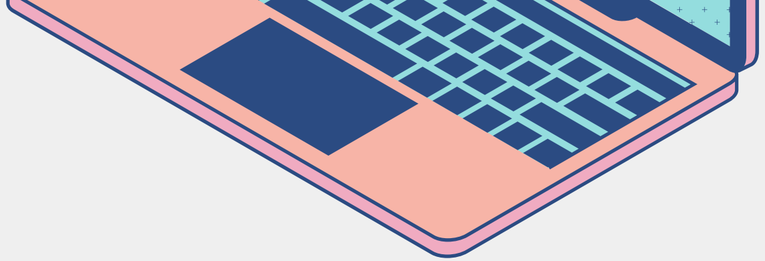
Repaso...

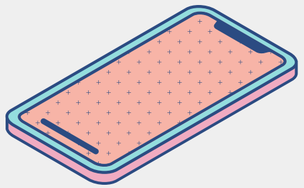
PROCESADORES DE LENGUAJE

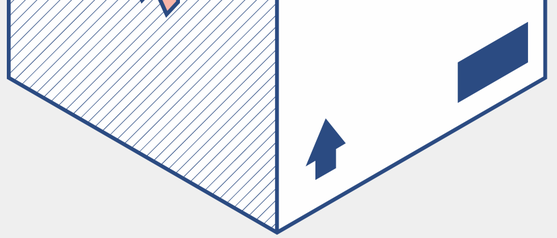
Los procesadores de lenguaje son programas que cuentan con un lenguaje fuente como entrada, dicha entrada será procesada para dar una acción como salida, las acciones van a depender del tipo de procesador.

Existen dos tipos principales de procesadores.









Compilador

Es un traductor que recibe como entrada un lenguaje y genera un ejecutable.

Intérprete

Reconoce un programa de entrada y lo ejecuta linea por linea. Su resultado es lógico.

Traductor

Cumple la misma función que un compilador pero sin funciones complejas de **análisis y sintesis.**

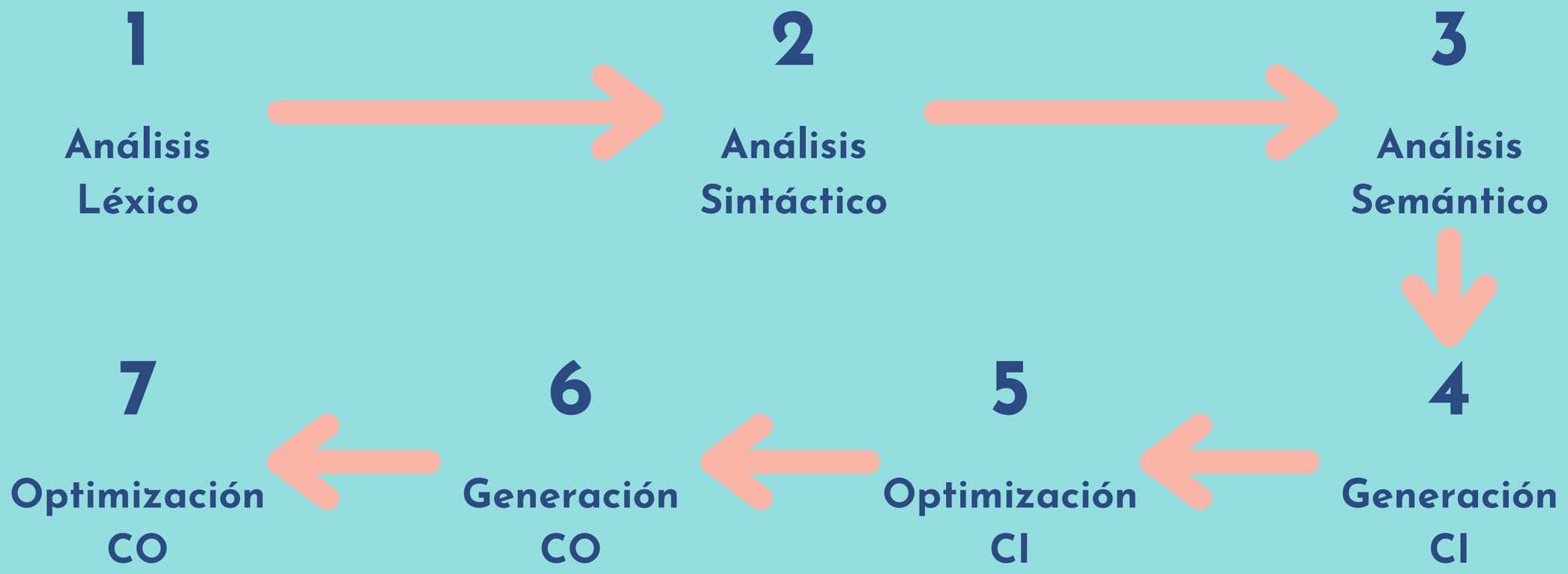
Compilador Híbrido

Combina la compilación y la interpretación.

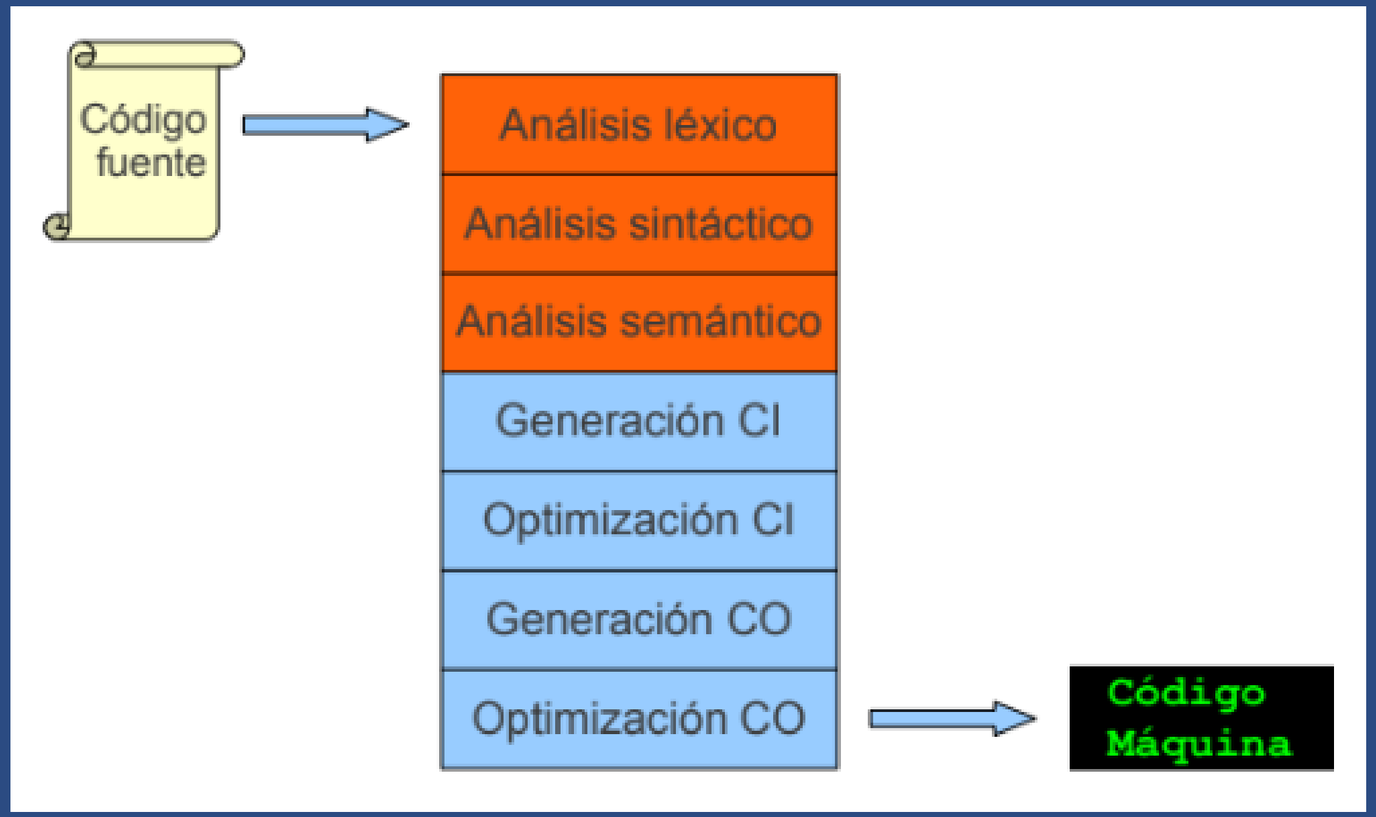
Compilador vs Intérprete

	Ventajas	Desventajas
Compilador	• No nos tenemos que preocupar mucho por los recursos del equipo. • Es mas rápido.	• Suele ser difícil la implementación. • Depende del SO y la arquitectura.
Intérprete	• Puede funcionar casi en cualquier lugar (multiplataforma). • Es "fácil" de implementar.	• Es mas lento. • Hay que contemplar los recursos y su disponibilidad.

Las fases de un Compilador

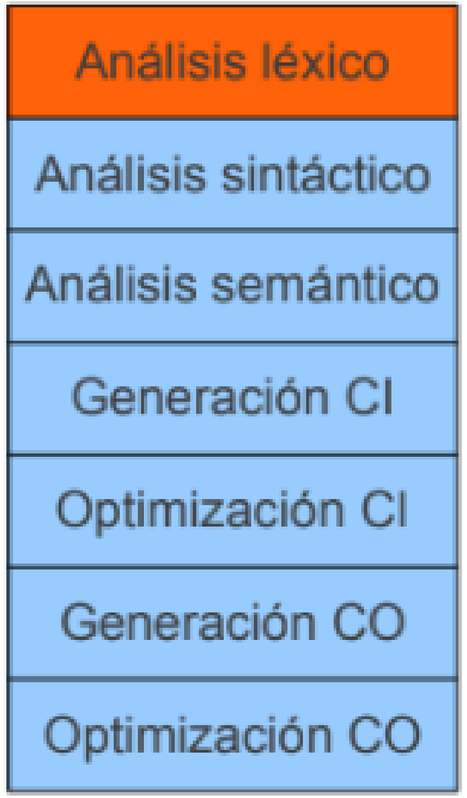


Estructura de un Compilador



Ejemplo...

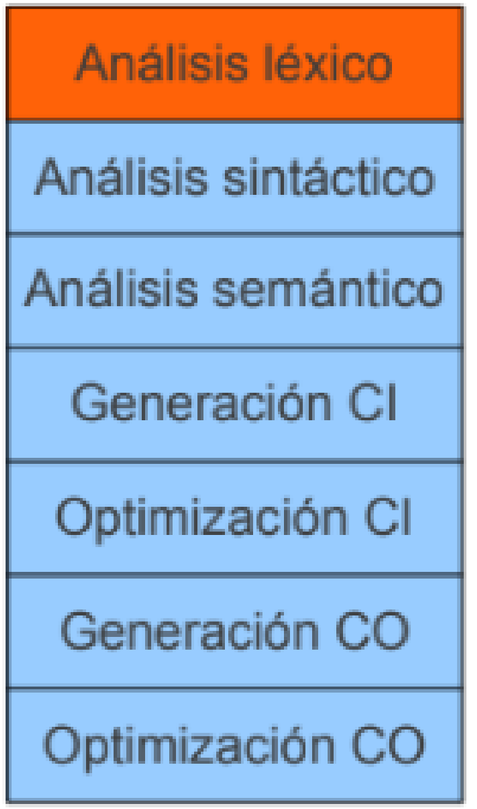
```
while (y < z) {
    int x = a + b;
    y += x;
}
```



Ejemplo...

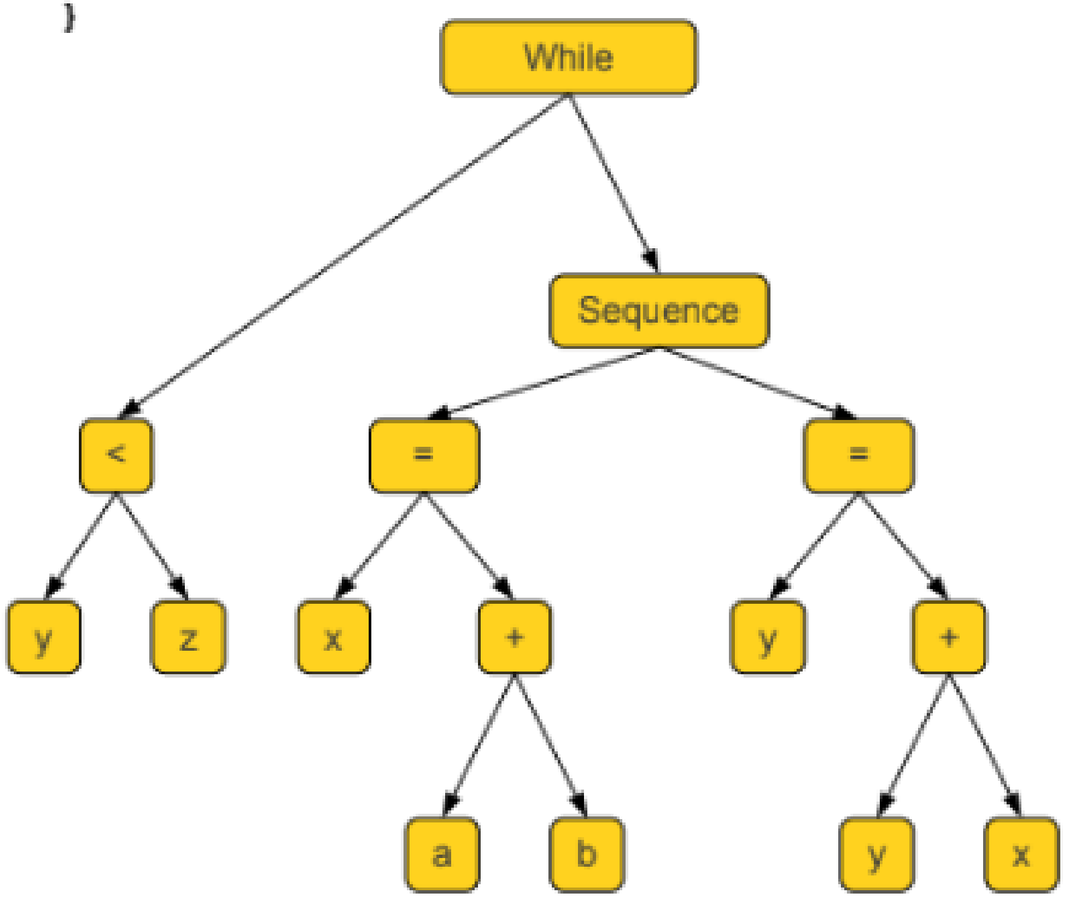
```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
T_While
T_LeftParen
T_Identifier y
T_Less
T_Identifier z
T_RightParen
T_OpenBrace
T_Int
T_Identifier x
T_Assign
T_Identifier a
T_Plus
T_Identifier b
T_Semicolon
T_Identifier y
T_PlusAssign
T_Identifier x
T_Semicolon
T_CloseBrace
```



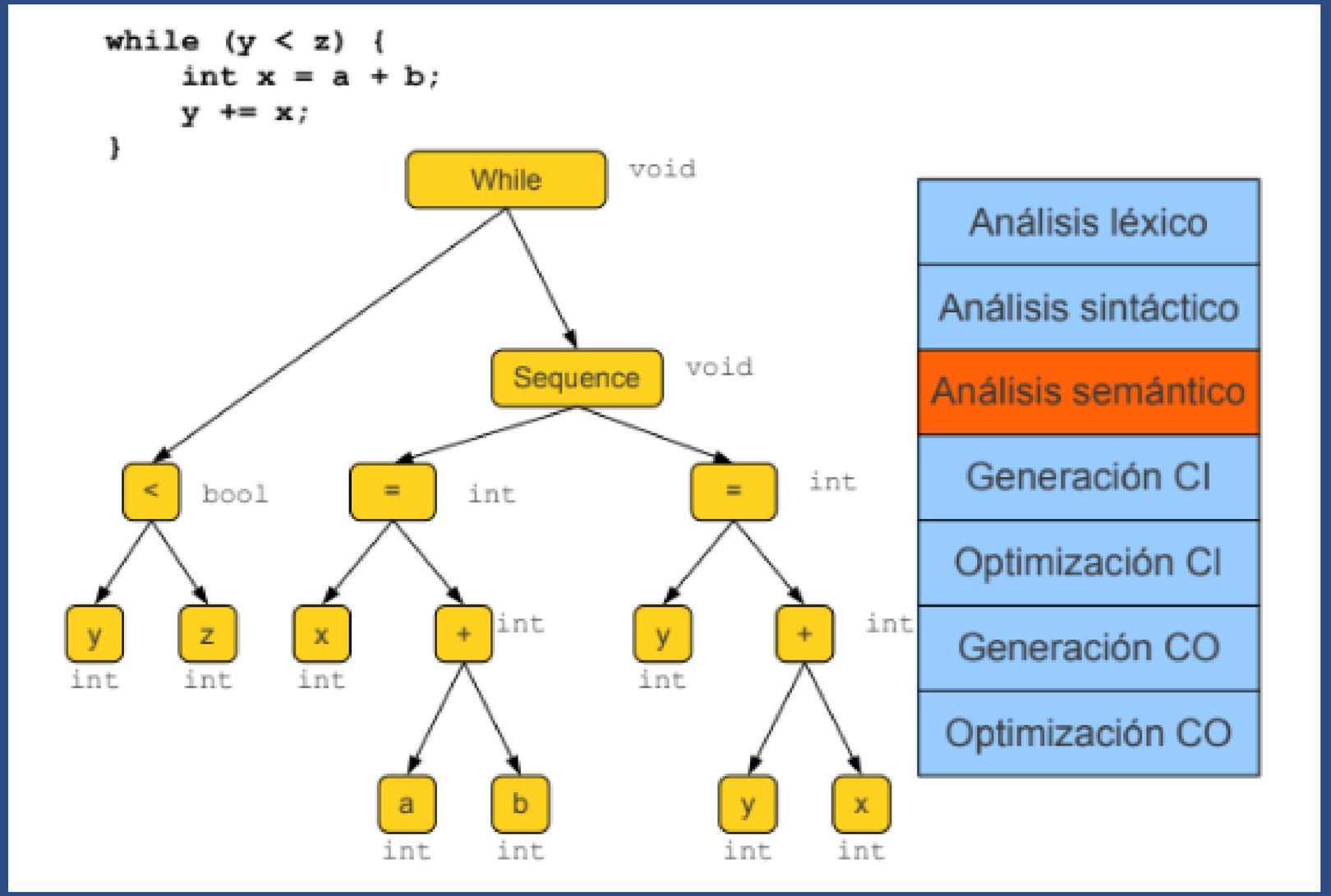
Ejemplo...

```
while (y < z) {
    int x = a + b;
    y += x;
}
```



- Análisis léxico
- **Análisis sintáctico**
- Análisis semántico
- Generación CI
- Optimización CI
- Generación CO
- Optimización CO

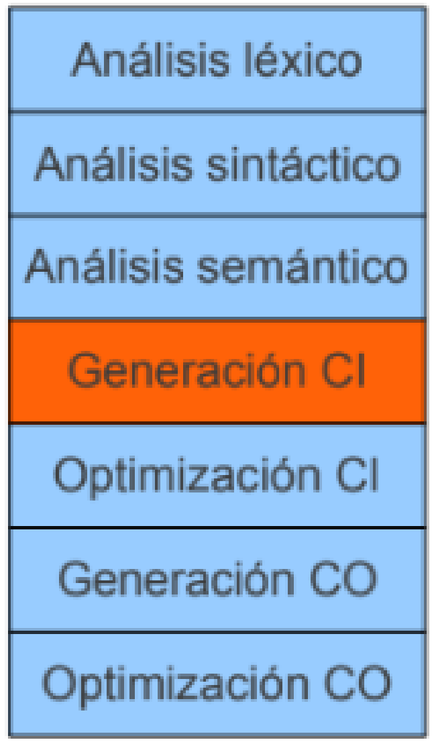
Ejemplo...


while (y < z) {
int x = a + b;
y += x;
}
While
void
Sequence
void
<
bool
=
int
=
int
y
int
z
int
x
int
+
int
y
int
+
int
a
int
b
int
y
int
x
int
Análisis léxico
Análisis sintáctico
Análisis semántico
Generación CI
Optimización CI
Generación CO
Optimización CO

Ejemplo...

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

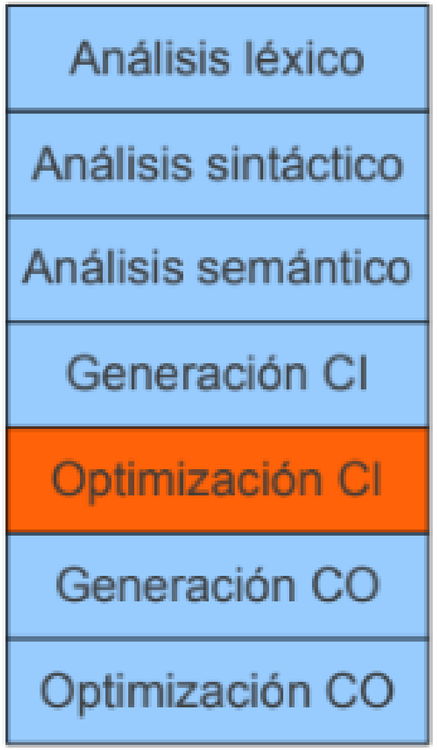
```
Loop: x = a + b
      y = x + y
      _t1 = y < z
      if _t1 goto Loop
```



Ejemplo...

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
      x = a + b
Loop: y = x + y
      _t1 = y < z
      if _t1 goto Loop
```



Ejemplo...

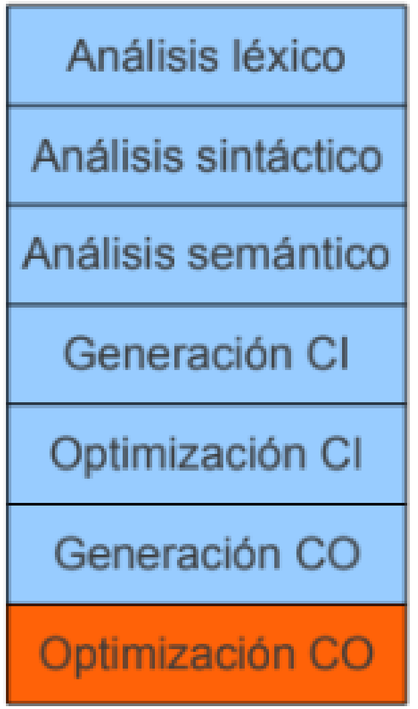
```
while (y < z) {
    int x = a + b;
    y += x;
}
```

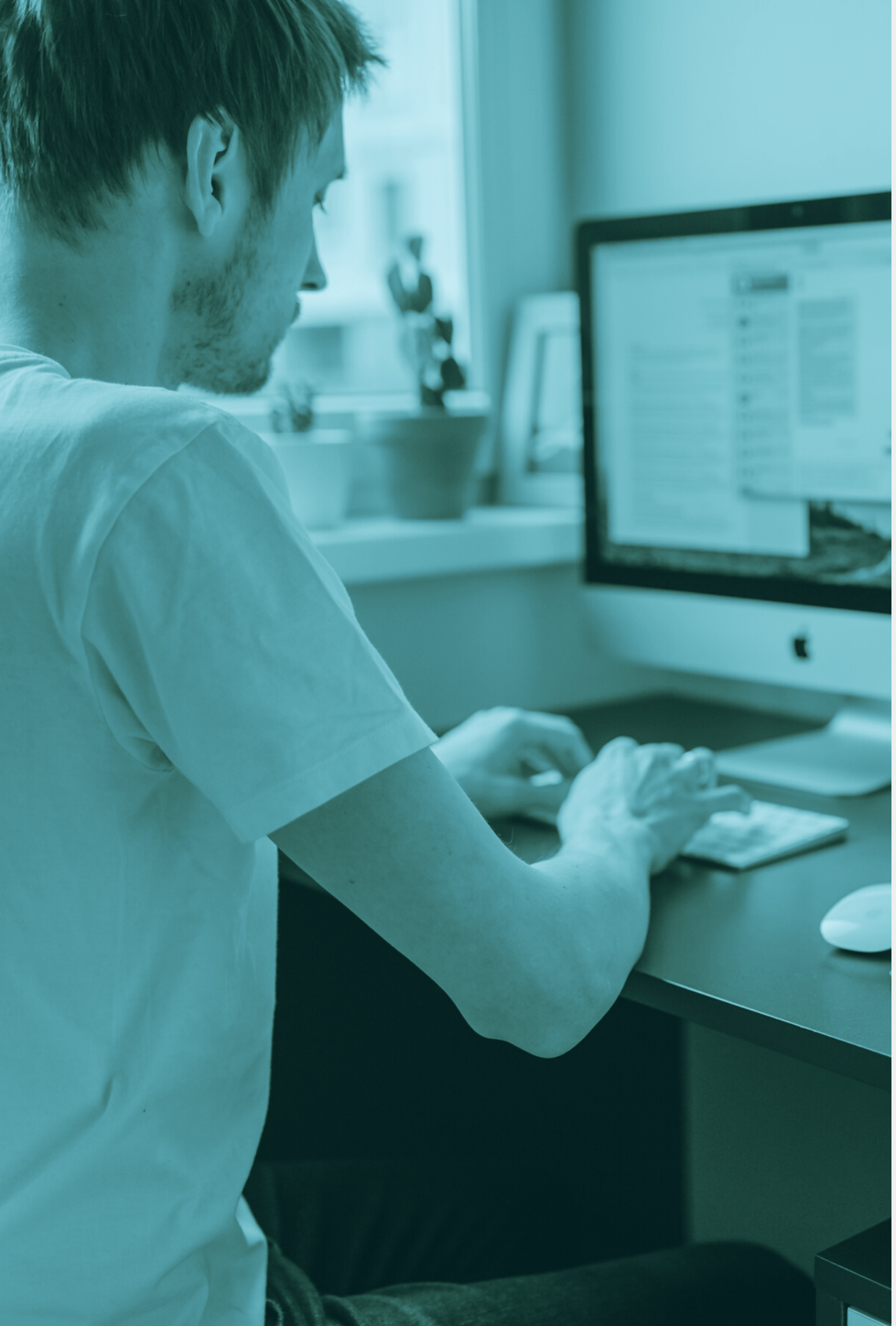
```
      add $1, $2, $3
Loop: add $4, $1, $4
      slt $6, $1, $5
      beq $6, loop
```

Ejemplo...

```
while (y < z) {
    int x = a + b;
    y += x;
}
```

```
      add $1, $2, $3
Loop: add $4, $1, $4
      blt $1, $5, loop
```





Conceptos de programación

PASO DE PARÁMETROS

Valor: enteros, strings, float, etc...
Referencia: arreglos, objetos, etc...

Tabla de Símbolos

Por lo general estas tablas son objetos utilizados por los compiladores para guardar información sobre las estructuras y composiciones del programa fuente.

Identificador	Tipo	Dimensión	Parámetro	Entorno	Función
ackerman	Float	0	valor	main	si
x	arr	1	referencia	main	no
var	String	0	valor	ackerman	no
num	Int	0	valor	ackerman	no

Entornos

Es un término asignado al conjunto de tablas de símbolos que son utilizables en un punto dado del programa.

```
1   // Entorno global
2   let x = [1,2,3,4,5];
3   function ackerman() {
4       // Entorno ackerman
5       let num = 2;
6       let x = 'ackerman';
7       if(num !== 0){
8           // Entorno If
9           console.log(x);
10      }
11  }
12
```

Herramientas Comunes

Irony

Jison

Jlex/Cup

Yacc

Goyacc

PLY

PRIMER SEMESTRE - 2023

Gracias por su atención..

Clase 1 - Repaso