Las funciones pueden ir en dos lados, “aplicar función primitiva” o en “evaluar”.

En “aplicar función primitiva” las funciones van ahí porque ya están los argumentos evaluados. Se necesita que todos los elementos sean correctos para su funcionamiento optimo, sino tirará error.

En cambio, en “evaluar” no es necesario que se evalúen todos los argumentos. Ya que por ejemplo en un or si encuentro un #T ya estaría dando true el resultado, lo mismo en un and y el #F. En cambio si tengo algo como por ejemplo (max 1 (+ 1 2)) va en aplicar porque si te importa tener los argumentos evaluados para usarlos

*¿Que es un interprete?*

Es una forma para la implementación de lenguajes, en la cual no se genera ningún código objeto. En su lugar, un intérprete lee el código fuente y lo “ejecuta” inmediatamente. Por lo tanto, si el intérprete encuentra la instrucción a=b+1, analiza los caracteres en el código fuente para determinar que la entrada es una instrucción de asignación, extrae de sus propias estructuras de datos el valor de b, le suma uno a ese valor y almacena el resultado en su propio registro de a.

A diferencia de los programas escritos en lenguaje de alto nivel mediante un compilador, estos tienen un proceso de dos etapas. En la primera el código fuente se traduce a código maquina y luego el hardware lo ejecuta para producir los resultados. Claramente con un interprete es mucho mas inmediato.

*Partes de un interprete*

ANÁLISIS LÉXICO

Es la primera transformación que tiene un código fuente para ser interpretado. La transformación la lleva a cabo un analizador léxico, que lee los caracteres de código fuente y los agrupa en una secuencia de tokens. Cada token es una estructura de datos pequeña que puede ser categorizable que representa las componentes básicas del lenguaje de programación que estamos queriendo procesar. Estos tokens pueden ser números, palabras reservadas, operadores, etc.

ANÁLISIS SINTÁCTICO

A continuación, los tokens se le pasan al analizador sintáctico o parser. Esto es es un programa informático que analiza una cadena de símbolos de acuerdo a las reglas de una gramática formal... El análisis sintáctico convierte el texto de entrada en otras estructuras (comúnmente árboles), que son más útiles para el posterior análisis. Por ejemplo javascript.

ANÁLISIS SEMÁNTICO

Aquí se atraviesa el árbol, insertando y comprobando la información de los tipos. Sin evaluación, una expresión como 1 + 2 es solo una serie de caracteres, tokens o una estructura de árbol que representa esta expresión. No significa nada. Evaluada, por supuesto, 1 + 2 se convierte en 3, 5 > 1 se convierte en true, 5 < 1 se convierte en false, y puts("Hola mundo!") se convierte en el mensaje amistoso que todo el mundo conoce

El llamado modelo de evaluación basado en ambientes consta de dos partes básicas:

1. Para *evaluar* una combinación (una expresión compuesta), se evalúan las subexpresiones y luego se le aplica el valor de la subexpresión operador a los valores de las subexpresiones operandos.
2. Para *aplicar* un procedimiento compuesto a un conjunto de argumentos, se evalúa el cuerpo del procedimiento en un nuevo ambiente.