El entorno de ejecución en los compiladores

Entorno de Ejecución

Los entornos de ejecución en tiempo de ejecución son sistemas o conjunto de servicios y estructuras necesarios para que un programa compilado pueda ejecutarse correctamente. Estos entornos permiten una mayor flexibilidad y adaptabilidad en la ejecución de programas y su función principal es coordinar y administrar varios aspectos clave para el funcionamiento del software

Organización de la memoria en tiempo de ejecución:

la organización de la memoria en tiempo de ejecución se refiere a cómo se gestiona y organiza la memoria mientras un programa se está ejecutando en un sistema y cómo se accede a las variables en diferentes contextos.

La memoria en tiempo de ejecución se organiza en diferentes secciones:

- Segmento de código (text): Contiene el código ejecutable del programa.
- Segmento de datos estáticos (data): Almacena variables globales y estáticas inicializadas.
- Segmento de bss: Almacena variables globales y estáticas no inicializadas.
- Segmento de pila (stack): Se utiliza para almacenar información sobre las funciones en ejecución, como variables locales y direcciones de retorno.
- Segmento de montón (heap): Se utiliza para almacenar datos dinámicos o variables cuyo tamaño no se conoce en tiempo de compilación.

Estrategias de asignación de memoria:

Asignación estática: Se asigna memoria durante la compilación para variables globales y estáticas.

Asignación dinámica: Se asigna memoria durante la ejecución, como en el uso de funciones como malloc() en C o new en C++ para asignar memoria en el montón.

Acceso a variables locales, no locales y globales:

Variables locales: Son aquellas definidas dentro de una función y solo son accesibles dentro de esa función.

Variables no locales: Son variables definidas fuera de la función actual pero accesibles dentro de ella debido a la estructura de anidamiento, como en funciones anidadas.

Variables globales: Son accesibles desde cualquier parte del programa y se definen fuera de cualquier función. Pueden ser accedidas y modificadas desde cualquier parte del código.

Paso de parámetros:

Paso por valor: Se envía una copia del valor del argumento a la función, lo que significa que los cambios realizados dentro de la función no afectan el valor original.

Paso por referencia: Se envía la dirección de memoria o referencia del argumento a la función, lo que permite a la función acceder y modificar directamente el valor original.

Paso por nombre: Se pasan expresiones o nombres en lugar de valores, lo que permite que se evalúen dentro de la función.