Unidades

1. ¿Qué es una unidad?

Es una entidad que cumple una función abstracta. También son llamadas *rutinas*. Hay de varios tipos, las mas comunes son las funciones y los procedimientos. Como toda entidad posee atributos, que son los siguientes:

- **Nombre**: string de caracteres que se usa para invocar a la rutina(identificador). El nombre de la rutina se introduce en su declaración y es lo que se usa para invocarla.
- **Alcance**: Rango de instrucciones donde se conoce su nombre. El alcance se extiende desde el punto de su declaración hasta algún constructor de cierre. Según el lenguaje, puede ser estático o dinámico. Para activar una rutina, la llamada a ésta debe estar solo dentro del alcance de la rutina.
- **Tipo**: el encabezado de la rutina define el tipo de los parámetros y el tipo del valor de retorno(si lo hubiera). Un llamado a una rutina es correcto si esta de acuerdo al tipo de la rutina(no podes asignar el resultado de un procedimiento a una variable porque el procedimiento no devuelve nada). La conformidad requiere la correspondencia de tipos entre parámetros formales y reales(en otras palabras, coherencia con los valores que se pasan cuando se llama a la rutina-parámetro real- y con lo que se espera recibir en la declaración -parámetro formal-).
- **l-value**: es el lugar de memoria en el que se almacena el cuerpo de la rutina.
- **r-value**: la llamada a la rutina causa la ejecución su código, eso constituye su r-valor.

2. Definición vs declaración

Muchos lenguajes(C, C++, ADA, etc) hacen distinción entre definir y declarar una rutina. Tip para no confundirse los conceptos:

- Declarar una rutina implica especificar su tipo, los parametros que recibe y su nombre.
- Definir una rutina implica declararla y darle un cuerpo(comportamiento).

3. Ejecución de una rutina

Cuando se invoca una rutina, se ejecuta una instancia del proceso con los particulares valores de los parámetros. Las rutinas poseen un segmento de código, que se almacena en la zona de código, y un registro de activación donde se coloca la información de datos locales que se almacena en la zona de datos. Dependiendo del esquema de ejecución que utilice el lenguaje, el registro de activación se almacena en la memoria desde un comienzo(esquema estático), o se almacenan tantos registros de activación de esa rutina como llamados se hagan, en caso de un esquema basado en pila.
El segmento de código son las instrucciones de la unidad que son almacenadas en la memoria de instrucción. Éste contenido es fijo. El registro de activación son los datos locales de la unidad que se almacenan en la memoria de datos. Éste contenido es cambiante.
Otros elementos:

- **Punto de retorno**: es una pieza cambiante de información que debe ser salvada en el registro de activación de la unidad llamada.
- **Ambiente de referencia**: puede ser local o no local. Si es local, las variables son locales, ligadas a los objetos almacenados en su registro de activación. Si es no local, las variables no son locales, ligadas a objetos almacenados en los registros de activación de otras unidades

4. ¿Cuál es la estructura de ejecución de un lenguaje de programación?

Hay tres:

- **Estático(espacio fijo)**: el espacio necesario para la ejecución se deduce del código. Todos los requerimientos de memoria necesarios se conocen antes de la ejecución por lo que la alocación puede hacerse estáticamente. Por esa misma razón tampoco puede haber recursión. Todas las variables son estáticas, sensibles a la historia.
- **Basado en pila(espacio predecible)**: el espacio se deduce del código. Este esquema sirve para programas más potentes cuyos requerimientos de memoria no pueden calcularse en traducción. La memoria a utilizarse es predecible y sigue una disciplina last-in-first-out. Las variables se alocan automáticamente y se desalocan cuando el alcance se termina. Se utiliza una estructura de pila para modelizar la estructura.
- **Dinámico(espacio impredecible)**: los datos son alocados dinámicamente solo cuando se los necesita durante la ejecución. No pueden modelizarse con una pila, el programador puede crear objetos de dato en cualquier punto arbitrario durante la ejecución del programa. Los datos se alocan en la zona de memoria heap.

5. ¿Qué es una unidad genérica?

Una unidad genérica es una unidad que puede instanciarse con parámetros formales de distinto tipo. Como todas las unidades instanciadas tienen el mismo nombre, da la sensación que una unidad pudiera ejecutarse con parámetros de distinto tipo. Por ejemplo, una unidad genérica que ordena elementos de un arreglo, podrá instanciarse para elementos enteros, flotantes, etc.