**Practica 6**

**Ejercicio 1**

Explique brevemente los siguientes conceptos.

* Parámetro
* Parámetro real
* Parámetro formal
* Ligadura posicional
* Ligadura por palabra clave o nombre

**Parámetro**: entidad local a una rutina usada para representar la *recepción* de datos en el momento de la invocación. Variable para obtener valores de entrada en una rutina, subrutina o método.

Según la lista de parámetros, puede ser:

**Parámetro real**: es la entidad que da valor a un parámetro en una invocación a una rutina. Es un parámetro que se codifica en la invocación del subprograma. Puede ser local en la unidad llamadora, o ser una vez en un rol formal de ella o un dato no local pero visible en dicha unidad o una expresión.

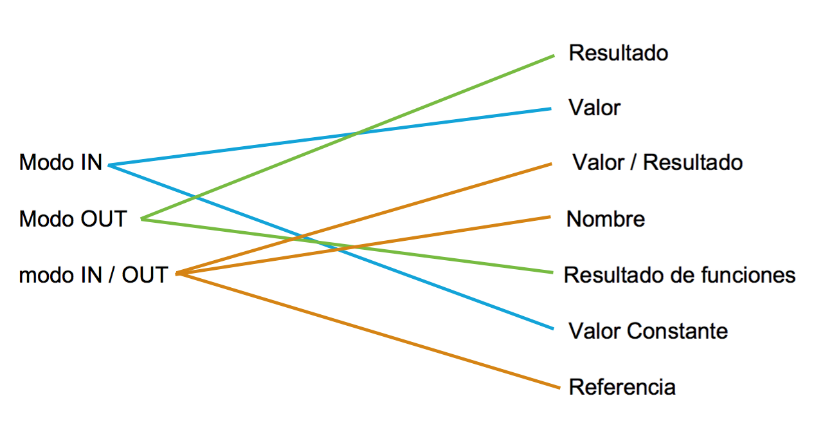
**Parámetro formal**: es la entidad local a la rutina que representa al parámetro. Es el parámetro declarado en la especificación del subprograma. Contiene los nombres y los tipos de los datos compartidos. En general son similares a las variables locales.

Según la ligadura con el parámetro formal, el parámetro puede ser:

**Ligadura posicional**: la ligadura entre la función formal y la realidad se lleva a cabo en el mismo lugar. Se corresponde con la posición que ocupan en la lista.

**Ligadura por palabra clave o nombre**: la ligadura entre el parámetro formal y el real se efectúa indicando el nombre de los mismos. Se corresponde con el nombre por lo que tanto podemos estar colocados indistintamente en la lista.

## Ejercicio 2



* Modo EN
  + Por valor
    - El valor del parámetro real se usa para inicializar el correspondiente parámetro real al invocar la unidad.
    - Se transfiere el dato real.
    - En este caso, la función formal como una variable local de la unidad llamada.
    - **Desventaja**
      * Consumir el tiempo para hacer la copia y el almacenamiento para duplicar el dato.
    - **Ventaja**
      * Protege los datos de la unidad llamadora, el parámetro real no se modifica.
  + Por valor constante
    - La información no se ha modificado.
    - Ejemplo: parámetros EN ADA.   
      **Desventaja**: requiere realizar más trabajo para implementar los controles.   
      **Ventaja**: protege los datos de la unidad llamadora, el parámetro real no se modifica.
* Modo OUT
  + Por Resultado
    - El valor del parámetro formal se copia al parámetro real al final de ejecutarse la unidad llamada.
    - El parámetro formal es una variable local, sin valor inicial.
    - **Desventajas**
      * Consumir tiempo y espacio
      * Si se repiten los parámetros reales, los resultados pueden ser diferentes.
      * Se debe tener en cuenta el momento en que se evalúa el parámetro real
    - **Ventaja**
      * Protege los datos de la unidad llamadora, el parámetro real no se modifica en la ejecución de la unidad llamada
  + Por resultado de funciones
    - El resultado de una función puede ser devolverse con el retorno como en Python, C, etc., o como en Pascal en el nombre de la función que se considera como una variable local.
    - Este resultado reemplaza la invocación en la expresión que contiene el llamado.
* Modo IN / OUT
  + Por Valor / Resultado
    - Copia a la entrada ya la salida de la activación de la unidad llamadora.
    - El parámetro formal es una variable local que se recibe una copia a la entrada del contenido real y la salida el parámetro real se recibe una copia de lo que tiene el parámetro formal.
    - Cada referencia al parámetro formal es una referencia local.
    - **Tiene las desventajas y las ventajas de ambos.**
  + Por Referencia
    - Se transfiere la dirección del parámetro real al parámetro formal. El parámetro formal será una variable local a la unidad llamadora que contiene la dirección en el ambiente no local.
    - Cada uno de ellos tiene un carácter formal. Esto significa que cualquier cambio que se realice en el parámetro formal dentro del cuerpo del subprograma quedará registrado en el parámetro real.
    - El parámetro real es compartido por la unidad llamada.
    - **Desventajas**
      * El acceso al dato es más lento por la indirección
      * Se pueden modificar el parámetro real inadvertidamente.
      * Se pueden generar alias y estos también la legibilidad y por lo tanto la confiabilidad.
    - **Ventaja**
      * Eficiente en tiempo y espacio. No se realizan copias del dato
* Por nombre:
  + El parámetro formal es el texto textual por el parámetro real.
  + Es decir, se establece la ligadura entre la formalidad y la parametrización real en el momento de la invocación pero la ligadura del valor se difiere hasta el momento en que se utiliza.
  + El objetivo es otorgar mayor flexibilidad a través de esta evolución de valor diferente.
  + Si el dato a compartir es:
    - Un único valor se comporta exactamente igual que el pasaje por referencia.
    - Si es una constante es equivalente a por valor.
    - Si es un elemento de un arreglo puede cambiar el suscripto entre las distintas referencias.
    - Si es una expresión se evalúa cada vez.

## Ejercicio 3

| **Tipo de pasaje de parámetros** | **Lenguaje** |
| --- | --- |
| valor, resultado, valor-resultado, referencia | ADA |
| valor, valor constante | Do |
| valor / referencia? | Rubí |
| valor / referencia | Java |
| valor / referencia? | Phyton |

1. Ada es más seguro que Pascal, respecto al pasaje de los parámetros en las funciones. Explique por qué.

En las funciones no sólo se cumplen los parámetros de valor, lo que impide que estos produzcan **efectos laterales** .

1. Explique cómo maneja Ada los tipos de parámetros in-out de acuerdo to type of dato.
   * Tipos primitivos: pasaje de parámetros por valor-resultado
   * Tipos no primitivos / compuestos: pasaje de parámetros por referencia.

# Ejercicio 6

Indique, con un ejemplo, el comportamiento del parámetro por nombre para los siguientes casos de parámetros reales:

* Un único valor se comporta exactamente igual que el pasaje por referencia.
* Si es una constante es equivalente a por valor.
* Si es un elemento de un arreglo puede cambiar el suscripto entre las distintas referencias
* Si es una expresión se evalúa cada vez.

# Ejercicio 8

1. Indique las diferencias entre el pasaje de subprogramas como parámetros deep y shallow.

* La ligadura superficial o superficial no es apropiada para lenguajes con estructuras de bloques que se puede acceder a un ambiente que está prácticamente no visible.
* Para más información se puede utilizar la ligadura profunda y se necesita que en el momento de la invocación a la unidad con el subprograma. cadena estática correspondiente.