| **Modo** | **Mecanismo** | **¿Qué se pasa?** | **¿Cuándo se evalúa? / ¿Cuándo afecta a la variable original?** | **Notas** |
| --- | --- | --- | --- | --- |
| **IN** El PF recibe el dato desde el PR. Conexión al inicio (invocación). Después se corta la vinculación | Por valor | Copia del valor | Se evalúa antes de entrar al subprograma. No afecta a la variable original | Mecanismo clásico |
|  | Por valor constante | Copia del valor (no modificable) | Igual que por valor, pero el parámetro se vuelve de solo lectura dentro del subprograma | Protege contra cambios accidentales |
| **OUT** El valor del PF se copia al parámetro real al terminar de ejecutarse la unidad que fue llamada | Por resultado | Dirección de la variable destino (sin leer el valor inicial) | Se ignora el valor de entrada. Al final de la unidad se copia al PR el valor del PF. | PF es una variable local a la rutina, sin valor inicial (debe ser inicializada si el lenguaje no lo hace por defecto) |
|  | Por resultado de funciones | Igual que por resultado | Igual que anterior, pero usado al retornar valores desde funciones | El retorno funciona como un OUT implícito |
| **IN/OUT** El PF recibe el dato del PR y el PF le envía el dato al PR al final la rutina. Vinculación al inicio y al final, no en el medio | Por valor/resultado | Copia del valor + dirección de la variable | Se pasa una copia, se trabaja sobre ella, y al final el PR recibe una copia de lo que tiene el PF | No se modifica el PR durante la ejecución de la rutina, se trabaja sobre el PF. Cada referencia al PF es una referencia local |
|  | Por referencia | Dirección de la variable | Se trabaja directamente sobre el PR (PF actúa como un puntero a PR) que es bidericcional (compartido por unidad llamadora y llamada) | Cada referencia al PF será a un ambiente no local🡪 cambios instantáneos. Se extiende el alcance de la rutina (aliasing situation) |
|  | Por nombre | Expresión textual / thunk | Ligadura entre PF y PR en momento de invocación pero ligadura de valor se pospone hasta que tenga que usarse (diferencia semántica con el anterior) por medio de un thunk🡪 Cada vez que necesito el valor del parámetro que pase, llamo s thunk que resulve el valor actual de la variable en ese momento | Evaluación diferida y múltiples efectos posibles (todos los de IN y los de OUT) |