Si el código que programamos simplemente se ejecutara linealmente siempre de la misma manera, no conseguiríamos más que preparar algunos calculos matemáticos, teniendo que modificar el programa para realizar cualquier ajuste. Es de importancia poder conseguir que el programa realice diferentes cosas en función de diferentes opciones.

Para realizar pruebas de control que determinen en qué dirección avanzará el programa, realizaremos operaciones lógicas teniendo en cuenta sus precedencias, y utilizando paréntesis para modificarlas, con las siguientes particularidades

- Si existen dos expresiones separadas por || (OR), y la primera es verdadera, no se evaluará la segunda y se devuelve directamente verdadero
- Si existen dos expresiones separadas por && (AND), y la primera es falsa, no se evaluará la segunda, y se devuelve directamente falso
- La comparación igual ==, y diferente !=, no tiene en cuenta los tipos de datos entre los elementos, de forma que el entero 5 y la cadena “5” pueden considerarse iguales en una comparación.
- La comparación igual estricto =, y diferente estricto ! si tiene en cuenta los tipos de datos de los elementos comparados || (OR), && (AND), ==, =, !=, ! , <, >
- Al hacer echo de un valor verdadero, se envía el valor 1
- Al hacer echo de un valor falso, no se envía nada (vacío)

Para controlar el flujo de ejecución del programa, podemos utilizar las siguientes estructuras de control:

- `if ( prueba ) { código } else { alternativa }`
  - Ejecuta un código si una prueba es verdadera. Opcionalmente ejecuta una alternativa si es falso
- `while ( prueba ) { código }`
  - Ejecuta un código repetidamente mientras la prueba sea verdadera, comprobandola antes de la primera ejecución
- `do { código } while ( prueba )`
  - Ejecuta un código repetidamente mientras la prueba sea verdadera, comprobandola después de la primera ejecución
- `for( valor inicial; incremento; prueba límite) { código }`
  - Ejecuta un código repetidas veces, en función del valor incial de una variable, el incremento en cada ejecución, y la prueba para permanencer en el bucle
- `foreach($miarray as $clave => $valor) { codigo }`
  - Recorre cada elemento de un array, estableciendo el valor del elemento actual en $valor, y opcionalmente su clave en $clave
- `switch( $variable_prueba) { case 'valor1': ..código.. break; case 'valor2': ..código.. break; default: código}`
  - Realiza una prueba sobre una variable, especificando el código a ejecutar según cada uno de los posibles valores especificados, y opcionalmente un código por defecto en default para el resto de los valores no especificados.

La referencia completa se incluye en:http://php.net/manual/es/language.control-structures.php

Dada su importancia, el uso correcto de las estructuras de control será desarrollado de manera extensa en las clases prácticas.