|  |  |  |  |
| --- | --- | --- | --- |
| Paradigma | Descripción | Donde se usa | Ejemplos |
| [Paradigma Funcional](https://wiki.uqbar.org/wiki/articles/paradigma-funcional.html) | Implementaciones de comportamiento que reciben un conjunto de datos de entrada y devuelven un valor de salida. | * LISP. * ML. * Haskell. * OCaml. * F# * Erlang | * Aplicaciones técnicas y matemáticas * Inteligencia Artificial (IA) * Compiladores y analizadores * Algoritmos |
| [Paradigma Lógico](https://wiki.uqbar.org/wiki/articles/paradigma-logico.html) | se basa en la lógica matemática. En lugar de una sucesión de instrucciones, un software programado según este principio contiene un **conjunto de principios**que se pueden entender como una recopilación de **hechos y suposiciones.** |  Prolog   Mercury   CLP (FD)   CSP (Constraint Satisfaction Problem)   Lambda Prolog | * [Desarrollo de aplicaciones de inteligencia artificial.](https://ferestrepoca.github.io/paradigmas-de-programacion/proglogica/logica_teoria/aplicaciones.html#1) * [Reconocimiento de lenguaje natural](https://ferestrepoca.github.io/paradigmas-de-programacion/proglogica/logica_teoria/aplicaciones.html#4) |
| Paradigma declarativo | Opuesto al imperativo. Los programas describen los resultados esperados sin listar explícitamente los pasos a llevar a cabo para alcanzarlos. | * Prolog * Lisp * Haskell * Miranda * Erlang * SQL | Para las empresas que quieren aplicar el mantenimiento y desarrollo de las aplicaciones con independencia de la persona que las desarrolló |
| [Paradigma de Objetos](https://wiki.uqbar.org/wiki/articles/paradigma-de-objetos.html) | Conjunto de objetos que se conocen entre sí a través de referencias y se envían mensajes en un ambiente. | * [Python](https://es.wikipedia.org/wiki/Python) * [Ruby](https://es.wikipedia.org/wiki/Ruby) * [Self](https://es.wikipedia.org/wiki/Self_(lenguaje_de_programaci%C3%B3n)) * [Smalltalk](https://es.wikipedia.org/wiki/Smalltalk)[17](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos#cite_note-17) | Este es muy usado en programas que requieran actualizaciones constantes |
| Paradigma imperativo | Los programas se componen de un conjunto de sentencias que cambian su estado. Son secuencias de comandos que ordenan acciones a la computadora. | * C * C# * C++ * Python * Ruby | Los lenguajes **imperativos** pueden resolver prácticamente cualquier problema en cualquier área: desde simples hasta complejos cálculos matemáticos |

|  |  |  |  |
| --- | --- | --- | --- |
| **Programación Orientada a Aspectos** | que permite una adecuada modularización de las aplicaciones y posibilita una mejor separación de responsabilidades | [AspectC++](https://en.wikipedia.org/wiki/AspectC%2B%2B)  [Aspecto](https://es.wikipedia.org/wiki/AspectJ) | Provee una fuerte herramienta para modularizar programas sin importar lo extensos y complicados que estos sean.   * Vuelve más limpio el código fuente. |

**BIBLIOGRAFIA**

<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/paradigmas-de-programacion/>

<https://wiki.uqbar.org/wiki/articles/paradigma-de-programacion.html#:~:text=Un%20paradigma%20de%20programaci%C3%B3n%20es,Las%20formas%20v%C3%A1lidas%20de%20combinarlas>.

<https://www.4rsoluciones.com/blog/que-son-los-paradigmas-de-programacion-2/>

<https://www.infor.uva.es/~cvaca/asigs/docpar/intro.pdf>