|  |  |  |  |
| --- | --- | --- | --- |
| **PARADIGMAS** | **DEFINICION** | **EJEMPLOS** | **SE USAN** |
| **POGRAMACION**  **ESTRUCTURADA** | Mejora la claridad, calidad y tiempo de desarrollo de un programa de computadora recurriendo únicamente a subrutinas y tres estructuras básicas: secuencia, selección (if y switch) e iteración (bucles for y while) | Do-while | Sirve para realizar una instrucción de forma automática siempre que se dé una determinada condición |
| **ROGRAMACION PROCEDIMENTAL** | Amplía el enfoque imperativo con la posibilidad de desglosar algoritmos en porciones manejables asi el  código de programa es más claro y evita las repeticiones innecesarias de código | procedimiento div(numero) {  out(numero/2);  out(numero/3);  out(numero/4); | si queremos dividir al número por 2,3,4,etc en otro pseudocódigo: |
| **PROGRAMACION MODULAR** | consiste en dividir un programa en módulos o subprogramas con el fin de hacerlo más legible y manejable | Double calcularnotapromedio(doublé[]notas){double sum=0;  For(int i=0;i<notas.length:i++)//suma todas las notas  Sum = sum + notas [i];  Return(sum/notas.length);//calcula el promedio } | Se usa para calcular una nota promedio, utilizando una matriz de tipo de datos double y retorna el promedio calculado |
| **PROGRAMACION LOGICA** | Consiste en la aplicación del corpus de conocimiento sobre lógica para el diseño de lenguajes de programación | F={ f, g, h, ….} | Donde cada elemento f es un símbolo funcional n-ario (n ≥ 0), por ejemplo, sucesor(x). Si n = 0 el símbolo de función se denomina símbolo de constante. Las constantes mas empleadas son verdadero y falso. |
| **PROGRAMACION FUNCIONAL** | se centra en las funciones. En un programa funcional, todos los elementos pueden entenderse como funciones y el código puede ejecutarse mediante llamadas de función secuenciales | Java, PHP, Ruby, Python, Elixir, Kotling, Haskell, Erlang. | Se usan segun la necesidad del programador |