



# **INSTITUTO TECNOLÓGICO SUPERIOR DE JEREZ**

**Ingeniería en Sistemas Computacionales**

**7mo Semestre**

**Alumno:**

***Daniel Alejandro de la Rosa Castañeda***

***NC:16070126***

**Materia:**

***Programación lógica y funcional***

**Nombre del trabajo:**

***Mapa Conceptual***

**Docente:**

***ISC Salvador Acevedo***

***Jerez de García Salinas a 3 de febrero del 2020***



## 1. ¿Qué es un paradigma de programación?

Paradigma también se puede denominar como método para resolver algún problema o realizar alguna tarea. El paradigma de programación es un enfoque para resolver problemas usando un lenguaje de programación o también podemos decir que es un método para resolver un problema usando herramientas y técnicas que están disponibles para nosotros siguiendo algún enfoque.

## 2. ¿Qué paradigmas de programación existen?

- Paradigma de programación imperativo
- Paradigma de programación funcional
- Paradigma de programación lógico
- Paradigma de programación orientado a objetos

## 3. ¿Cuáles son las características que define a cada uno de ellos?

- Paradigma de programación imperativo: Se basa en la arquitectura de Von Neumann. Funciona cambiando el estado del programa a través de declaraciones de asignación. Realiza la tarea paso a paso cambiando el estado. El enfoque principal está en cómo lograr la meta. El paradigma consiste en varias declaraciones y después de la ejecución de todo el resultado se almacena.
- Paradigma de programación funcional: El principio clave de este paradigma es la ejecución de una serie de funciones matemáticas. El modelo central para la abstracción es la función destinada a un cálculo específico y no la estructura de datos. Los datos se acoplan libremente a las funciones. La función oculta su implementación. La función se puede reemplazar con sus valores sin cambiar el significado del programa. Algunos de los lenguajes como perl, javascript utilizan principalmente este paradigma.
- Paradigma de programación lógico: se puede denominar como modelo abstracto de computación. Resolvería problemas lógicos como rompecabezas, series, etc. En la programación lógica tenemos una base de conocimiento que conocemos antes y, junto con la pregunta y la base de conocimiento que se da a la máquina, produce resultados. En los lenguajes de programación normales,



dicho concepto de base de conocimiento no está disponible, pero al usar el concepto de inteligencia artificial, aprendizaje automático, tenemos algunos modelos, como el modelo de Percepción, que utiliza el mismo mecanismo. En la programación lógica, el énfasis principal está en la base de conocimiento y el problema.

- Paradigma de programación orientado a objetos: el programa se escribe como una colección de clases y objetos destinados a la comunicación. La entidad más pequeña y básica es el objeto y todo tipo de cálculo se realiza solo en los objetos. Se hace más hincapié en los datos en lugar del procedimiento. Puede manejar casi todo tipo de problemas de la vida real que hoy están en escena.

#### 4. Ejemplos de dichos paradigmas

- Paradigma de programación imperativo

```
int fact(int n) {  
    int f = 1;  
    while (n > 1) {  
        f = f * n;  
        n = n - 1;  
    }  
    return f;  
}
```

- Paradigma de programación funcional

```
Function FindAvg(int x, int y){ return ((x+y)/2);}
```

- Paradigma de programación lógico



Brother(X,Y)	/* X is the brother of Y	*/
:-	/* if there are two people F and M for which*/	
father(F,X),	/* F is the father of X	*/
father(F,Y),	/* and F is the father of Y	*/
mother(M,X),	/* and M is the mother of X	*/
mother(M,Y),	/* and M is the mother of Y	*/
male(X).	/* and X is male	*/

- Paradigma de programación orientado a objetos

```
public class Example{  
    private String name;  
    public void setName(String n){name = n;}}
```

5. Aplicaciones REALES donde se implementan dichos paradigmas

6. Ventajas y desventajas de cada paradigma

- Paradigma de programación imperativo

**Ventaja:**

1. Muy simple de implementar
2. Contiene bucles, variables, etc.

**Desventaja:**

1. El problema complejo no puede resolverse
2. Menos eficiente y menos productivo
3. La programación paralela no es posible.

- Paradigma de programación funcional

Ventajas

1. Debido a que las funciones puras no cambian ningún estado y dependen completamente de la entrada, son fáciles de entender. El valor de retorno dado por tales funciones es el mismo que el resultado producido por



ellas. Los argumentos y el tipo de retorno de las funciones puras se dan por su firma de función.

2. Debido a la naturaleza de las funciones puras para evitar el cambio de variables o cualquier dato externo, la implementación de la concurrencia se vuelve eficaz
3. La programación funcional admite el concepto de evaluación diferida, lo que significa que el valor se evalúa y almacena solo cuando es necesario.
4. Las funciones puras toman argumentos una vez y producen una salida inmutable. Por lo tanto, no producen ningún resultado oculto. Utilizan valores inmutables, lo que facilita la depuración y las pruebas.
5. El estilo de programación funcional trata las funciones como valores y pasa las mismas a otras funciones como parámetros. Mejora la comprensión y la legibilidad del código.

## Desventajas

1. Los valores inmutables combinados con la recursividad pueden conducir a una reducción en el rendimiento.
2. En algunos casos, escribir funciones puras provoca una reducción en la legibilidad del código
3. Aunque escribir funciones puras es fácil, combinar las mismas con el resto de la aplicación y las operaciones de E / S es difícil
4. Escribir programas en estilo recursivo en lugar de usar bucles para el mismo puede ser una tarea desalentadora

- **Paradigma de programación lógico**

## Ventajas

1. Descripciones independientes de la implementación (unificación semántica).
2. Puede mejorarse la eficiencia modificando el componente de control sin tener que modificar la lógica del algoritmo.
3. Base de conocimiento fácilmente escalable.
4. Relaciones multipropósito.
5. Expresión simple y precisa de los problemas.
6. Generación rápida de prototipos e ideas complejas.



7. Sencillez en la implementación de estructuras complejas.
8. Potencia.

#### Desventajas

1. Dependiendo del problema a solucionar, la implementación y el motor de inferencia, puede llegar a ser extremadamente ineficiente.
  2. Pocas y muy específicas áreas de aplicación.
  3. Existen muy pocas herramientas de depuración, en su mayoría poco efectivas.
  4. En problemas reales, es poco utilizado.
  5. Si el programa no contiene suficiente información para responder una consulta la respuesta puede ser una que se preste para malentendidos.
  6. Inferencia limitada por su base de conocimiento.
- **Paradigma de programación orientado a objetos**

#### Ventajas:

1. Seguridad de datos
2. Herencia
3. Código de reutilización
4. Flexible y abstracción también está presente

#### 7. ¿Qué es una función (desde el punto de vista matemático)?

Una definición técnica de una función es: una relación de un conjunto de entradas a un conjunto de posibles salidas donde cada entrada está relacionada con exactamente una salida.

#### 8. ¿Qué es la programación funcional?

La programación funcional es un paradigma o estilo que valora la inmutabilidad, las funciones de primera clase, la transparencia referencial y las funciones puras. Si ninguna de esas palabras tiene sentido para ti, ¡no te preocupes! Vamos a desglosar toda esta terminología en este artículo.

La programación funcional evolucionó a partir del cálculo lambda, un sistema matemático construido en torno a la abstracción y generalización de funciones. Como resultado, muchos lenguajes de programación funcionales parecen muy matemáticos.



### 9. ¿Qué es una expresión matemática?

En matemáticas, una expresión o expresión matemática es una combinación finita de símbolos que está bien formada de acuerdo con las reglas que dependen del contexto. Los símbolos matemáticos pueden designar números (constantes), variables, operaciones, funciones, corchetes, puntuación y agrupación para ayudar a determinar el orden de las operaciones y otros aspectos de la sintaxis lógica.

### 10. ¿Qué es la reducción (simplificación o evaluación) de expresiones matemáticas?

La reducción de términos semejantes es un método que se emplea para simplificar expresiones algebraicas. En una expresión algebraica, los términos semejantes, son aquellos que tienen la misma variable; es decir, tienen las mismas incógnitas representadas por una letra, y estas tienen los mismos exponentes.

