Tema: Paradigmas de Programación:

* En general, es una forma de ver/entender/modelar el mundo.
* En el ámbito del software, es un estilo fundamental de programación. Determina cómo el programa “ve” el mundo.
* Determina como debe ser usado el lenguaje por el programador.
* Algunos lenguajes son multiparadigma.
* No se hace una selección explícita en el código.
* Hay 4 paradigmas principales:
* Funcional
* Lógico
* Imperativo
* Orientado a Objetos
* *Paradigma Lógico:*
* El mundo se modela mediante predicados lógicos.
* Se aplica directamente principios de matemática discreta.
* Para utilizar en aplicaciones comerciales.
* Prolog es el principal lenguaje.
* *Paradigma Funcional:*
* El mundo se modela como funciones matemáticas.
* Lenguajes 100% funcionales:

🡪 Lisp

🡪 Scheme

* Permite declarar funciones, pasar funciones por parámetro, retornar funciones, entre otros.
* Muchos lenguajes modernos, han incorporado este paradigma por conveniencia.
* Expresiones lambda.
* *Paradigma Imperativo:*
* El mundo se modela por instrucciones, pasos, procedimientos.
* Ampliamente utilizados en aplicaciones comerciales.
* Tienden a ser muy eficientes (generalmente compilan en lenguaje máquina).
* Gramaticalmente simples.
* Se consideran como de nivel de abstracción bajo.
* Ejemplos: C/ C++.
* *Paradigma Orientado a Objetos:*
* El mundo se modela como objetos del mundo real junto con su interacción.
* Muy naturales para el ser humano.
* Facilita la reutilización del código.
* Mayoritariamente, no generan lenguaje máquina.
* Posee muchas estructuras sintácticas.
* Java/ C++/ Python/ Javascript.

Tipos de Datos:

Primitivos:

Se identifican al ser palabras reservadas del lenguaje de programación, representarn tipos de datos básicos como son:

* int
* float
* short
* byte
* large

Referencias:

Utiliza la memoria para referenciar otras direcciones de memoria, que contienen los datos que voy a utilizar.

* Si se cambia el valor de la referencia, todas las variables que referenciaban al valor del dato toma el nuevo valor.
* String c = new String(“casa”);
* String d = new String(c);

|  |  |  |
| --- | --- | --- |
| Posición en Memoria | Referencia en Memoria | Dato en Memoria |
| 1 |  | “casa” |
| 2 | C | 1 |
| 3 |  | “casa” |
| 4 | d | 3 |

* Una manera de cambiar el valor de un String sin cambiar el valor de otro es usando la sentencia *new*.

If (e.equals(d)){

}else{

}

* Forma de comprar Strings utilizando métodos.