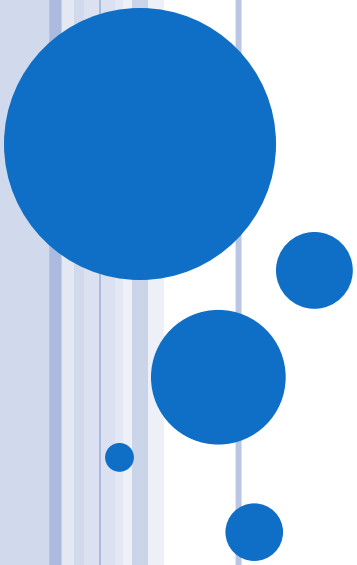
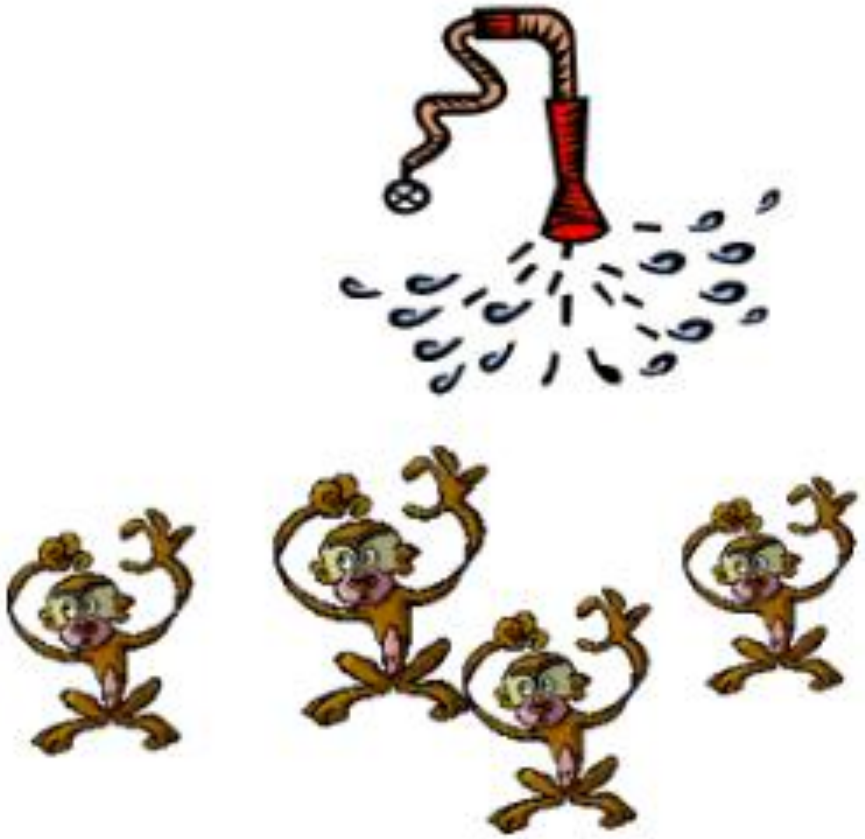


Paradigmas de programación

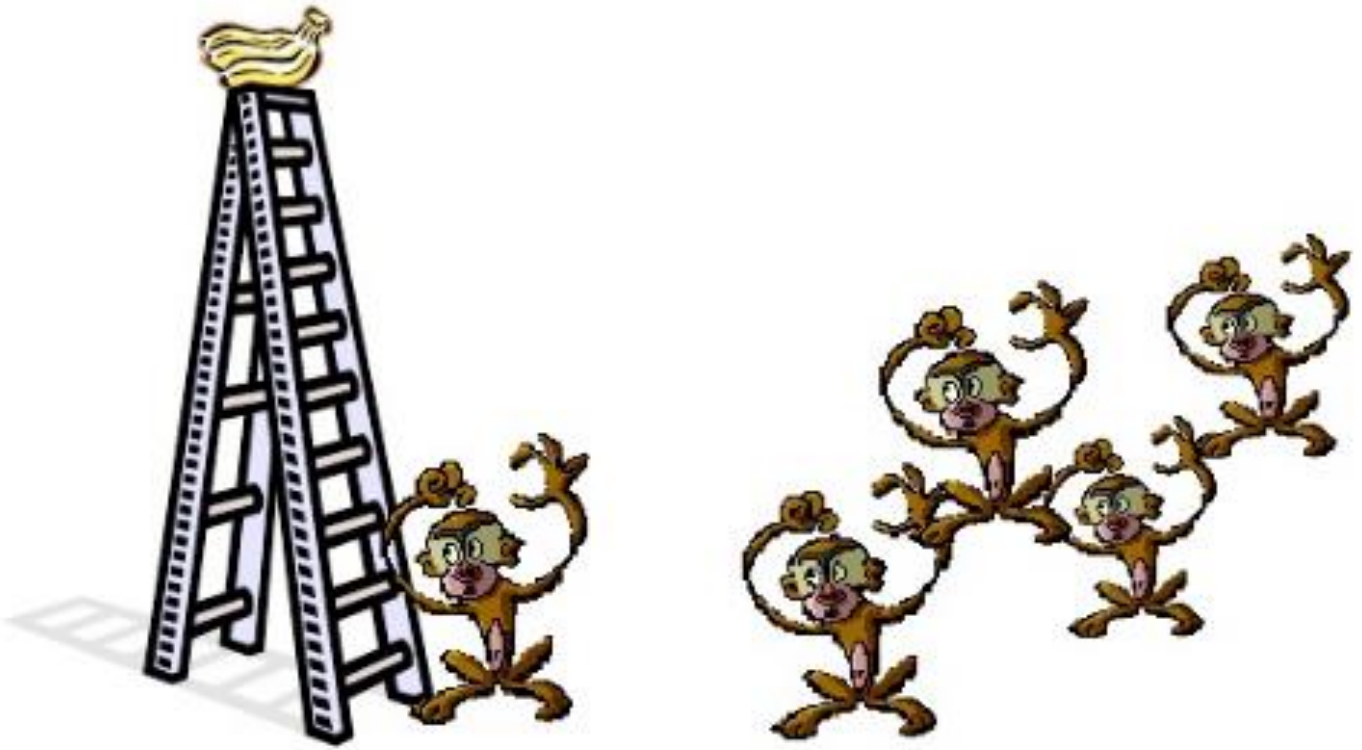
¿Qué es un paradigma?

Un grupo de científicos colocó cinco monos en una jaula, en cuyo centro colocaron una escalera y, sobre ella, un montón de bananas.

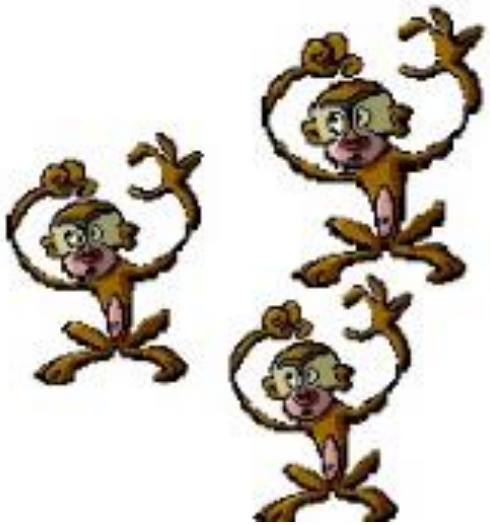
Cuando un mono subía la escalera para agarrar las bananas, los científicos lanzaban un chorro de agua fría sobre los que quedaban en el suelo.

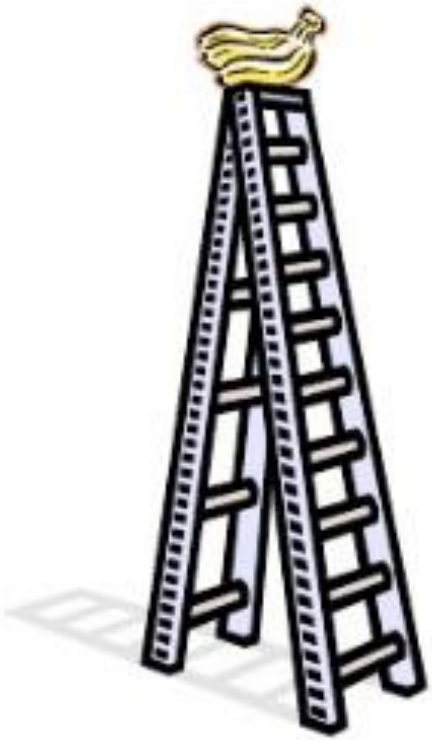


Después de algún tiempo, cuando un mono iba a subir la escalera, los otros lo golpeaban.



Pasado algún tiempo más, ningún mono subía la escalera, a pesar de la tentación de las bananas.

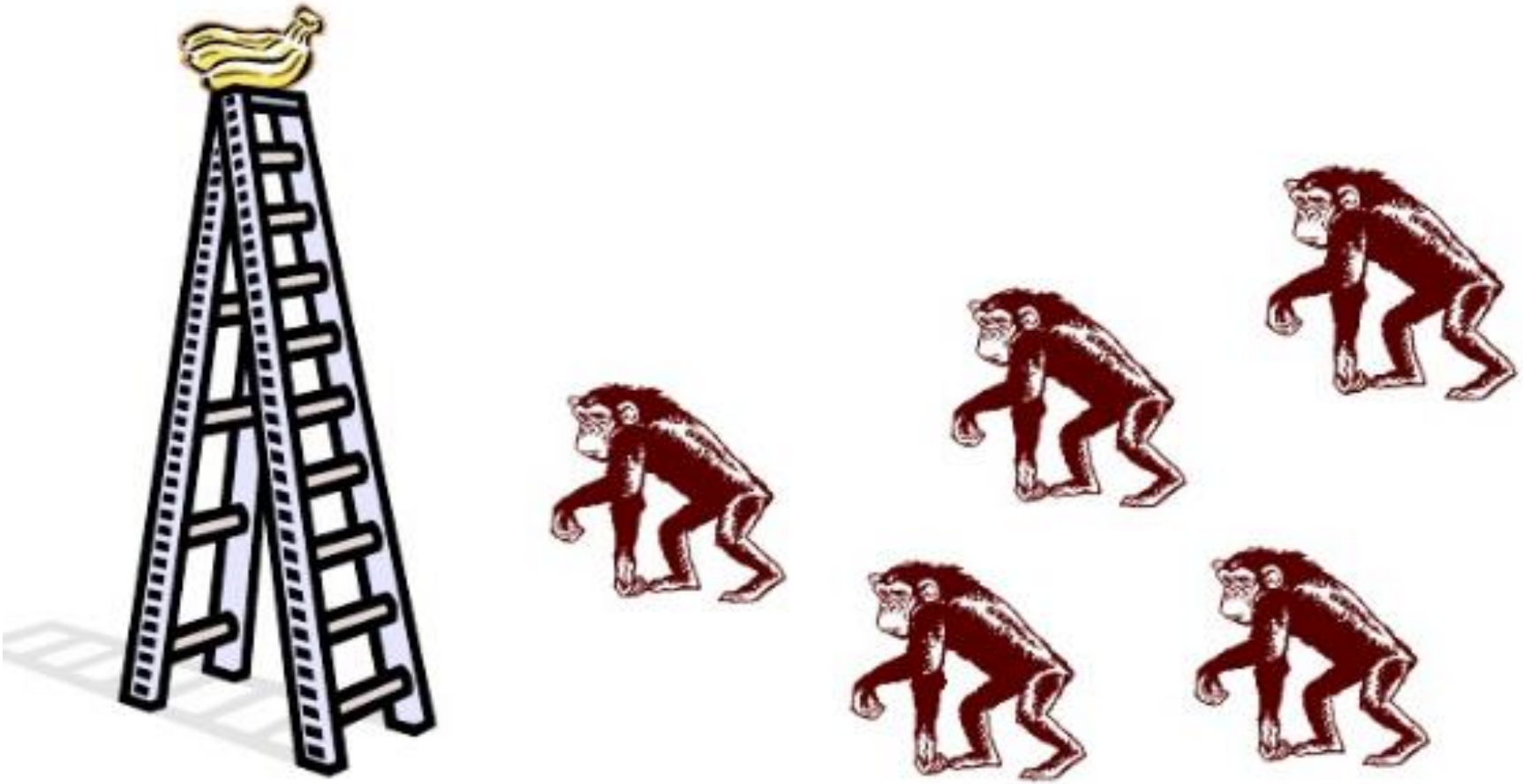




Entonces, los científicos sustituyeron uno de los monos.
La primera cosa que hizo fue subir la escalera,
siendo rápidamente bajado por los otros,
quienes le acomodaron tremenda paliza.

Después de algunas palizas,
el nuevo integrante del grupo
ya no subió más la escalera,
aunque nunca supo el por qué
de tales golpizas.

Un segundo mono fue sustituido, y ocurrió lo mismo. El primer sustituto participó con entusiasmo de la paliza al novato. Un tercero fue cambiado, y se repitió el hecho, lo volvieron a golpear. El cuarto y, finalmente, el quinto de los veteranos fueron sustituidos.



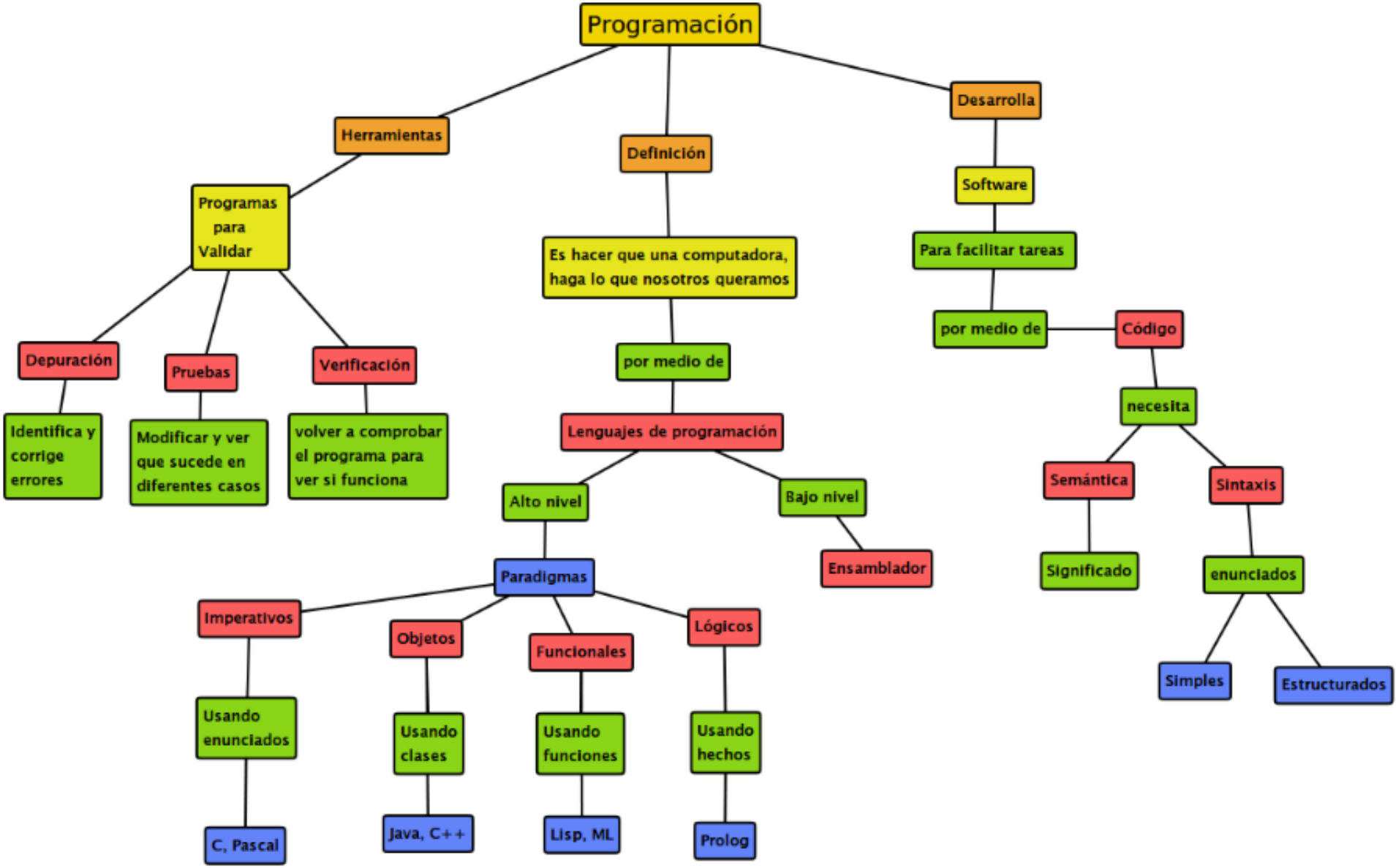
Quedó, entonces, un grupo de cinco monos que, aún cuando nunca recibieron un baño de agua fría, continuaban golpeando a aquel que intentase llegar a los plátanos.

Si fuese posible preguntar a algunos de ellos por qué
le pegaban a quien intentaba subir la escalera,
con certeza la respuesta sería:
"No sé, aquí las cosas siempre se han hecho así."

¡¿Te suena este "*razonamiento*"?!

Paradigma de programación

- Paradigma de programación es una propuesta tecnológica adoptada por una comunidad de programadores cuyo núcleo central es incuestionable en cuanto a que unívocamente trata de resolver uno o varios problemas claramente delimitados.
- La resolución de estos problemas debe suponer consecuentemente un avance significativo en al menos un parámetro que afecte a la ingeniería de Software.
- Los lenguajes de programación suelen implementar, a menudo de forma parcial, varios paradigmas


Programación
Herramientas
Definición
Desarrolla
Programas para Validar
Depuración
Pruebas
Verificación
Identifica y corrige errores
Modificar y ver que sucede en diferentes casos
volver a comprobar el programa para ver si funciona
Es hacer que una computadora, haga lo que nosotros queramos
por medio de
Lenguajes de programación
Alto nivel
Bajo nivel
Ensamblador
Paradigmas
Imperativos
Objetos
Funcionales
Lógicos
Usando enunciados
Usando clases
Usando funciones
Usando hechos
C, Pascal
Java, C++
Lisp, ML
Prolog
Software
Para facilitar tareas
por medio de
Código
necesita
Semántica
Sintaxis
Significado
enunciados
Simples
Estructurados

PROGRAMACIÓN DECLARATIVA

En la programación declarativa **se describe la lógica de computación para resolver un problema sin describir un flujo de control** de ningún tipo. En la programación declarativa **no es necesario definir algoritmos** puesto que se detalla la solución del problema en lugar de como llegar a esa solución.

Describe que se debe calcular, sin explicitar el cómo.

- *Programación funcional*
- *Programación lógica*

PROGRAMACIÓN FUNCIONAL

El paradigma funcional considera al **programa como una función matemática**, donde el dominio representaría el conjunto de todas las entradas posibles (inputs) y el rango sería el conjunto de todas las salidas posibles (outputs).

- Ventajas:

  Más fáciles de escribir y depurar. Uso de **arquitecturas paralelas**.

- Desventajas:

  Carecen de librerías, interfaces con otros lenguajes y herramientas de depuración.-Al no utilizar sentencias, no existen asignaciones, por lo que una vez que **las variables** asumen un valor, **no cambian durante la ejecución**. **Modelo alejado de Von Neumann**

- Lenguajes de programación:

  *Haskell
  Scala
  Miranda
  Scheme
  Lisp
  SAP
  Standard ML

PROGRAMACIÓN LOGICA

La Programación Lógica estudia el **uso de la lógica** para el planteamiento de problemas y el control sobre las reglas de inferencia para alcanzar la solución automática. **Basado en la lógica de predicados**.

- VENTAJAS

  Permite **visualizar gráficamente el camino que sigue la solución a un problema**.- No se necesitan muchos conocimientos técnicos para utilizar esta técnica.

- DESVENTAJAS

  Dado que los flujos (representados con flechas) **pueden ir de cualquier lugar a cualquier lugar** da espacio para que el diagrama llegue a ser casi inentendible.- Los ciclos deben ser reinterpretados para poder ser diagramados en esta técnica

  Algunas veces la analogía entre el diagrama y la codificación en el Lenguaje de Programación resulta ser compleja.

- Lenguajes de programación:

  *Prolog
  ALF
  Gödel
  Mercury

PROGRAMACIÓN ESTRUCTURADA O IMPERATIVA

La programación estructurada es una forma de escribir programas de computadora utilizando ciertas instrucciones de control (bucles y condicionales), se describe paso a paso un conjunto de instrucciones que deben ejecutarse para variar el estado del programa y hallar la solución, es decir, **se emplea un algoritmo que describe los pasos necesarios para solucionar el problema**.

- VENTAJAS

  La programación estructurada se caracteriza por tener **lenguajes sencillos** de comprender para una persona con basto conocimiento ya que se puede leer de secuencia por ello se considera semi-natural la lógica que se emplea en este tipo de programación es mas visible lo que facilita las **pruebas de escritorio** y la corrección de errores.

- DESVENTAJAS

  El principal inconveniente de este método de programación, es que se obtiene **un único bloque** de programa, **que cuando se hace demasiado grande puede resultar problemático su manejo**

- LENGUAJES

  ALGOL
  PL/I
  Ada
  BASIC
  C
  Fortran
  Pascal
  Perl
  PHP
  Java
  Python

Paradigmas

- Programación declarativa
- Programación imperativa
- Programación lógica
- Programación funcional

PARADIGMA ORIENTADO A EVENTOS

La programación dirigida por eventos es un paradigma de la programación en el que tanto **la estructura como la ejecución de los programas van determinados por los sucesos que ocurran en el sistema**, definidos por el usuario o que ellos mismos provoquen.

- VENTAJAS

  Uno de los gran enfoques de la programación orientada a eventos es la **calidad en la interfaz grafica de usuario** la cual es importante para que el administrador de el evento le facilite el uso del software.

- DESVENTAJAS

  La mayoría de herramientas de creación de interfaces gráficas de usuario se distribuyen como frameworks.

  Los **frameworks** implementan el bucle de eventos y la cola de eventos para que no tengamos que implementarlos nosotros mismos. pero el mecanismo queda encerrado dentro del framework lo cual **dificulta su programación y ejecución**.

- LENGUAJES DE PROGRAMACIÓN

  Visual Basic

  Java script

  Visual C++

  Visual C#

  Lexico

PROGRAMACIÓN ORIENTADA A OBJETOS

El comportamiento del programa es llevado a cabo por objetos, entidades que representan elementos del problema a resolver y tienen atributos y comportamiento.

- Ventajas:
  - La **facilidad de re-utilización de código en diferentes proyectos**.
  - La **facilidad de añadir, suprimir o modificar nuevos objetos** nos permite hacer modificaciones de una forma muy sencilla.
  - Debido a las sencillez para abstraer el problema, los programas orientados a objetos son más **sencillos de leer y comprender**
- Desventajas
  - **Complejidad para adaptarse**
  - La **necesidad de utilizar bibliotecas** de clases obliga a su aprendizaje y entrenamiento.
- Lenguajes de programación:
  - SIMULA
  - **SMALLTALK**
  - ACTOR
  - EIFFEL
  - JAVA
  - C++
  - PYTHON
  - RUBY

PROGRAMACIÓN ORIENTADA A ASPECTOS

Este es un paradigma de programación relativamente nuevo, que incluye como característica el concepto de "Aspecto" que pretende dar una determinada funcionalidad al sistema; **separando esta funcionalidad del sistemas**, con el fin de **aumentar la modularidad**.

- VENTAJAS

  Permite una complementación modularizada reduciendo el acoplamiento entre sus partes.-**El código es más limpio, menos duplicado, más fácil de entender y de mantener**. -

  Mayor re utilización, los aspectos tienen mayores probabilidades de ser **reutilizados en otros sistemas** con requerimientos similares.

- Lenguaje

  AspectJ

Lenguajes de programación:

- **Haskell (Programación funcional)**
- **ML**(Programación funcional)
- **Lisp**(Programación funcional)
- **Prolog**(Programación Lógica)
- **F-Prolog**(Programación Lógica Difusa)
- **Curry**(Programación Lógico-Funcional)
- **SQL** (Base de datos)
- **Scala**: Imperativo, orientado a objetos, funcional, genérico y concurrente
- **Erlang**: Funcional, concurrente y distribuido
- **Perl**: Imperativo, orientado a objetos y funcional
- **PHP**: Imperativo, orientado a objetos, funcional y reflexivo
- **JavaScript**: Imperativo, orientado a objetos (prototipos) y funcional
- **Java**: Imperativo, orientado a objetos, reflexivo y genérico
- **Python y Ruby**: Imperativo, orientado a objetos, reflexivo y funcional
- **C++**: Imperativo, orientado a objetos, funcional y genérico
- **C#**: Imperativo, orientado a objetos, funcional (lambda), reflexivo y genérico

PARADIGMAS DE PROGRAMACIÓN

PROGRAMACAION LOGICA

- Hechos
- Sentencias
- Reglas
- Logica
- Control

PROGRAMACION ORIENTADA A OBJETOS

- Metodos
- Clases
- Atributos
- Objetos
- Modularizacion
- Herencia

PROGRAMACIÓN FUNCIONAL

- Concepto Matematico
- Dominio
- Rango
- Funciones

PROGRAMACION ORIENTADA A EVENTOS

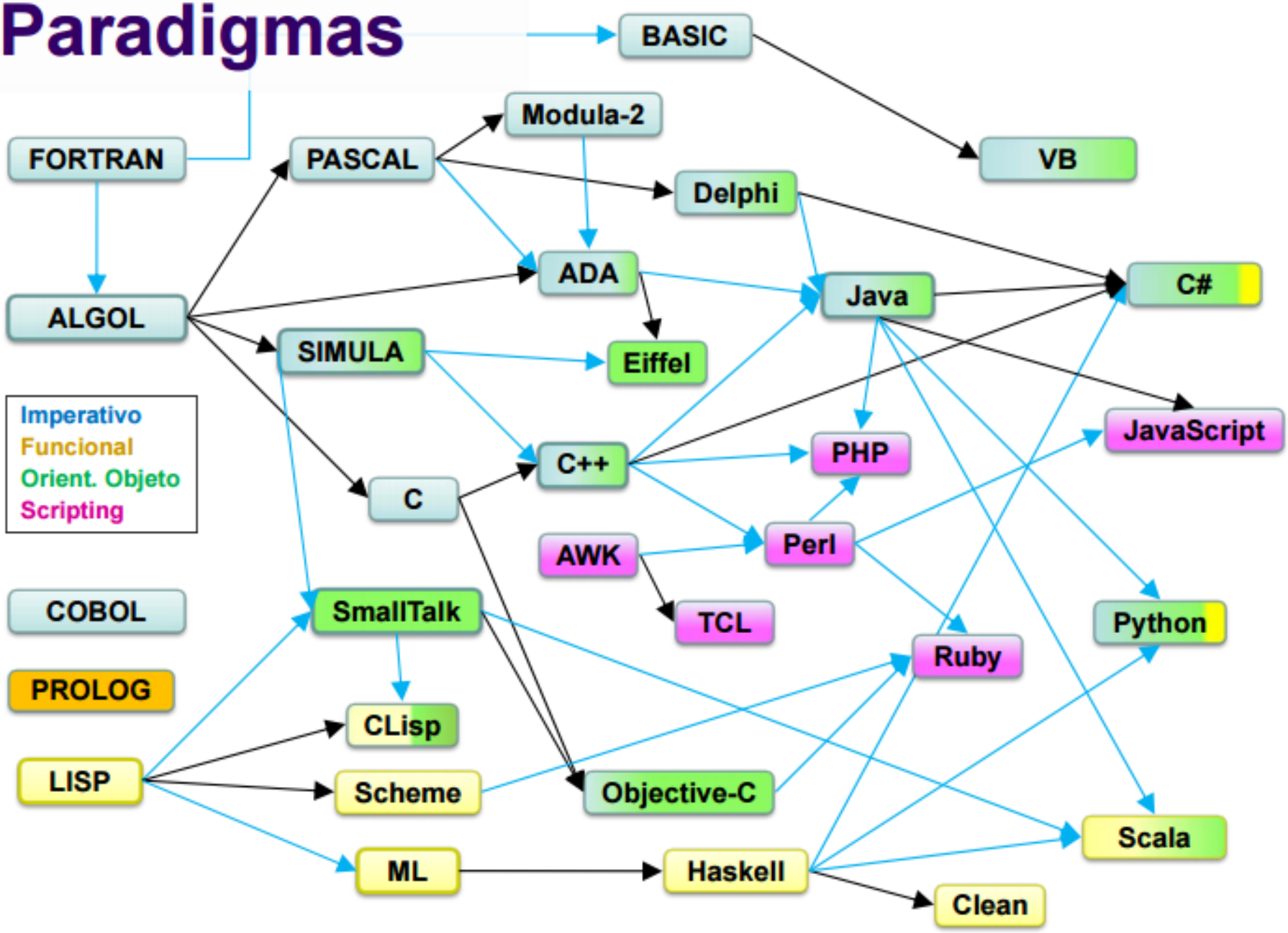
- Sucesos
- Administrador del Evento
- Usuario

PROGRAMACIÓN ESTRUCTURADA

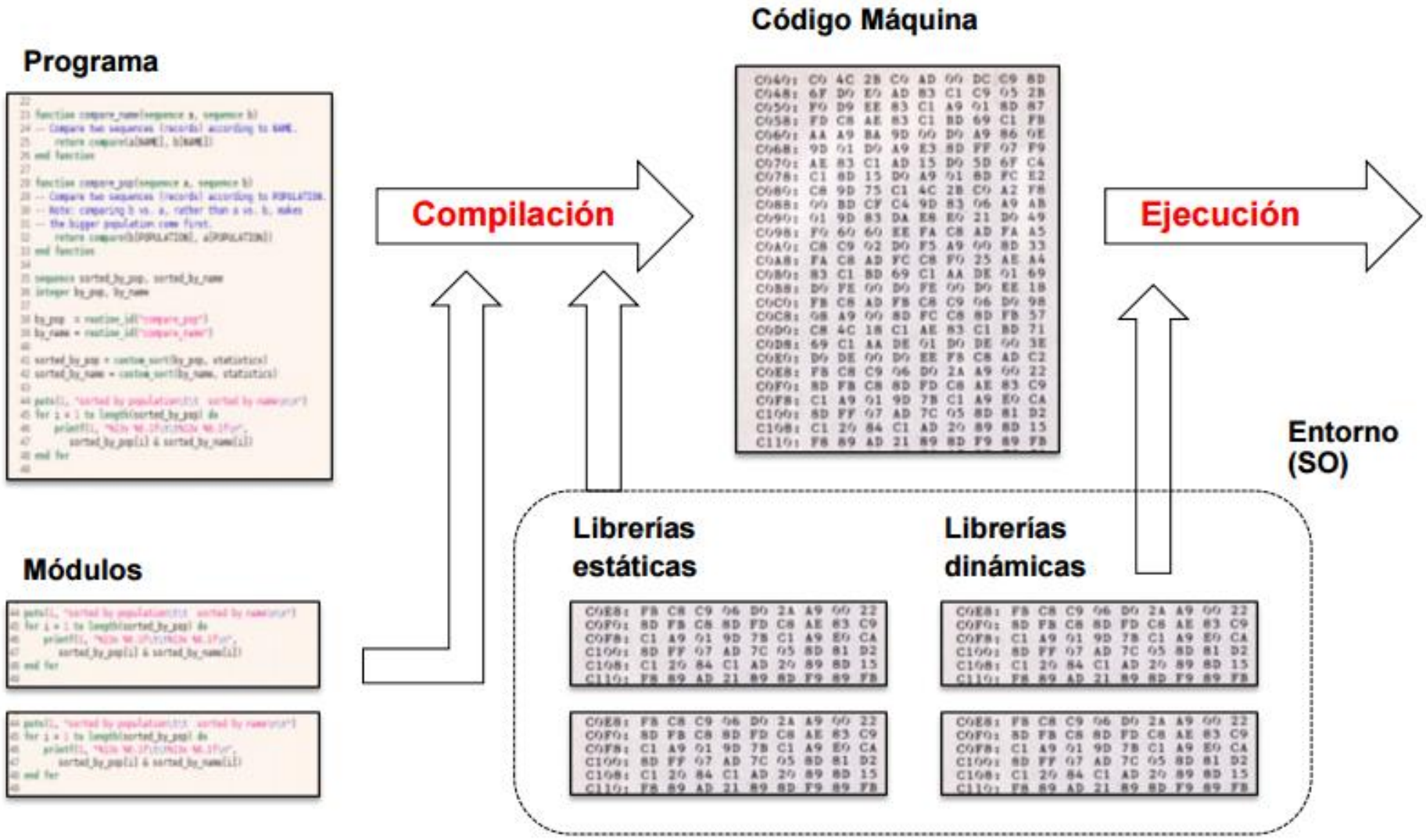
- Secuencia
  - Pasos
- Seleccion
  - Condicion
- iteracion
  - Instruccion

Línea del tiempo de los paradigmas de programación:

Paradigmas


BASIC
FORTRAN
PASCAL
Modula-2
VB
Delphi
ADA
Java
C#
ALGOL
SIMULA
Eiffel
Imperativo
Funcional
Orient. Objeto
Scripting
JavaScript
C++
PHP
C
AWK
Perl
COBOL
SmallTalk
TCL
Python
Ruby
PROLOG
CLisp
LISP
Scheme
Objective-C
Scala
ML
Haskell
Clean

Código compilado:



Aplicación a paradigmas de programación:

Funcional
Lógica
Estructurada
Orientada a objetos
Orientada a eventos
Orientada a aspectos

Paradigma de programación funcional

- Sólo existen valores y expresiones matemáticas que devuelven nuevos valores a partir de los declarados.
- El calculo de lambda.
- Aplicaciones:
  - Inteligencia artificial
  - Procesamiento de lenguaje natural
  - Reconocimiento de voz
  - Rápido desarrollo de prototipos
  - Programación de sistemas de telecomunicaciones y telefonía (Erlang lenguaje diseñado por Ericsson)
  - Asignación de tripulaciones a vuelos(Haskell)
  - Base de datos para e-commerce

Paradigma de programación lógica

- *"El conjunto de enunciados lógicos que son asumidos como axiomas constituyen el programa lógico."* Los enunciados que deben ser derivados, que pueden ser vistos como entradas que desencadenan el cálculo son las demandas o metas.
- los sistemas de programación lógica son llamadas bases de datos deductivas
- Permite formalizar hechos, por ejemplo:
  - las aves vuelan
  - los pingüinos no vuelan
  - "Cleo" es un ave
  - "Doky" es un perro
  - "Alegría" es un ave

  Permite agregar restricciones:
  - una mascota vuela si es un ave y no es un pingüino

  Es posible establecer hipótesis:
  - ¿ "Cleo" vuela ?
  - ¿ qué mascotas vuelan ?....

  Verificar la hipótesis y responder a las incógnitas:
  - Es cierto que "Cleo" vuela.
  - "Cleo" y "Alegría" vuelan.

Aplicaciones:

- Inteligencia artificial
- Procesamiento de lenguaje natural
- IBM desarrolló EI – IA para **asignación de aviones en líneas aéreas israelíes en tiempo real**
- **Sistema experto para diseño de anteojos a la medida del cliente por medio de una fotografía**, el diseño se basa en el color de piel, cabello y preferencias.
- **CVE; desarrollado por Siemens** para **verificar** de forma automática, **la salida de un circuito digital, garantiza 100% seguridad** ya que contempla todos los posibles patrones de entrada verificando las salidas de la simulación.

ESTRUCTURADA

```
1   #include<stdio.h>
2   void sumaUno(int a, int b, int& r);
3   void sumaDos(int a, int b, int* r);
4
5   main()
6   {
7       int x,y;
8       sumaUno(7, 5, x);
9       printf ("7 + 5 = %d", x);
10      sumaDos(3, 6, y);
11      printf ("3 + 6 = %d", y);
12  return 0;
13  }
14
15  void sumaUno(int a, int b, int& r)
16  {
17      r = a + b;
18  }
19
20  void sumaDos(int a, int b, int* r)
21  {
22      *r = a + b;
23  }
24
```

ORIENTADA A OBJETOS

- No es difícil, pero es una manera especial de pensar.

Clase coche

Objeto

Propiedades/atributos:

*Color
*Modelo

Métodos

*Ponerse en marcha
*Parar

Clase fracción

Objeto:
fracción

Propiedades/atributos :
*Numerador
*Denominador

Métodos:
*Simplificarse
*Sumarse
*Restarse

```java
class Ciclo {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        ciclo1: while (true) {
            System.out.println("Estoy en Ciclo1");

            ciclo2: while (true) {
                System.out.println("Estoy en Ciclo2");

                ciclo3: while (true) {
                    System.out.println("Estoy en Ciclo3");

                    cicloLectura: while (true) {
                        System.out.println("Estoy en CicloLectura");
                        System.out.print("Dame un numero: ");

                        //Leemos de la entrada estandar
                        int num = in.nextInt();

                        switch (num) {
                        case 1:
                            break ciclo1;
                        case 2:
                            break ciclo2;
                        case 3:
                            break ciclo3;
                        case 4:
                            break cicloLectura;
                        default:
                            System.out.println("Solo numeros del 1 al 4");
                            break;
                        }
                    }
                    System.out.println("Sali de CicloLectura");
                }
                System.out.println("Sali de Ciclo3");
            }
            System.out.println("Sali de Ciclo2");
        }
        System.out.println("Sali de Ciclo1");
        System.out.println("Fin");

    }
}
```

- A la acción de crear un objeto a partir de una clase de le conoce como ***instanciar,creando un objeto***:

  miCoche = new Coche()

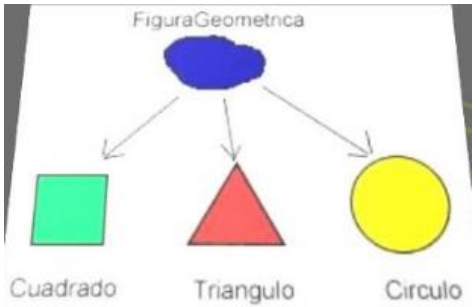
- Cuando tenemos un ***objeto sus propiedades toman valores***. El ***valor*** concreto ***de una propiedad de un objeto se llama estado***.Para acceder a un estado de un objeto para ver su valor o cambiarlo se utiliza el operador punto.

  miCoche.color = rojo

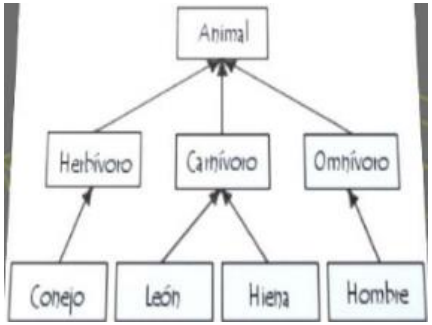
- **Mensajes en objetos**
  Un mensaje en un objeto es ***la acción de efectuar una llamada a un método***. Para mandar mensajes a los objetos utilizamos el operador punto, seguido del método que deseamos invocar.
- miCoche.ponerseEnMarcha()

Características:

- Encapsulamiento: Clases publicas o privadas
- Abstracción: Son las características esenciales de un objeto que lo distinguen de los demás.



- Herencia: Propiedad que permite que los objetos sean creados a partir de otros.



- Polimorfismo: Acceder a un variado rango de funciones a través de las misma interfaz

Orientada a ASPECTOS

- Su objetivo es lograr la **separación entre los requerimientos funcionales de los no funcionales** para obtener un mejor entendimiento de los conceptos, eliminando la dispersión del código y haciendo que las implementaciones resulten más comprensibles, adaptables y reutilizables.

- Aplicaciones:
  - Filtros de imágenes.

ORIENTADA A EVENTOS

- Aplicaciones con una Interface Gráfica de Usuario son casos típicos de Sistemas Dirigidos por Eventos.
- Java proporciona un marco de aplicación que facilita el manejo de eventos y la creación de GUI.

Bibliografía:

- http://www.tiki-toki.com/timeline/entry/406010/PARADIGMAS-DE-PROGRAMACION/
- http://ldp-roberto.blogspot.mx/2010/08/haz-clic-en-la-imagen-para-agrandar.html
- https://jarroba.com/paradigmas-de-programacion/
- https://www.emaze.com/@AOCTQIFZ/Paradigmas-de-Programaci%C3%B3n
- https://www.genbetadev.com/paradigmas-de-programacion/diferencias-entre-paradigmas-de-programacion
- https://www.infor.uva.es/~cvaca/asigs/docpar/intro.pdf
- https://erickcion.wordpress.com/2010/08/25/etiquetas-para-identificar-ciclos-en-java/

Bibliografía

- https://es.slideshare.net/psfracchia/programacin-funcional
- https://es.slideshare.net/GabyNarvaez/aplicaciones-desarrolladas-con-prolog
- https://www.desarrolloweb.com/articulos/499.php
- https://es.slideshare.net/KillexFghijk/la-poo
- https://es.slideshare.net/wfranck/programacin-orientada-a-aspectos-poa
- http://www.manchoneria.es/colaboracion/tipo/1677/la-programacion-orientada-a-aspectos-poa
- http://iie.fing.edu.uy/~josej/docs/Programacion%20Orientada%20Aspectos%20-%20Jose%20Joskowicz.pdf
- http://lagosjuan777.blogspot.com/2010/09/programacion-orientada-eventos.html