Ejercicio 1

1. Que son lenguajes imperativos y declarativos?

El **imperativo** es el modo verbal que sirve para dar órdenes, ruegos, consejos e incluso deseos de forma directa. Dado su uso, en imperativo no existe la primera persona o yo, ya que no nos podemos dar órdenes directas a nosotros mismos. Las personas del imperativo son entonces aquellas a las que nos podemos dirigir: Tú, vosotros, usted y ustedes. En todas las demás personas se utiliza el modo subjuntivo.

Los Tipos más utilizados son:

- Estructurados
- OOP
- Procedimental

En general, un lenguaje imperativo ofrece al programador conceptos que se traducen de forma natural al modelo de la máquina. Los lenguajes imperativos más destacados de la historia han sido:

- FORTRAN
- Algol
- Pascal
- C
- Modula-2
- Ada

La **declarativa** es que siempre se describe el resultado final deseado, en lugar de mostrar todos los pasos de trabajo. Para alcanzar el objetivo, en la programación declarativa se determina automáticamente la vía de solución. Esto funciona siempre y cuando las especificaciones del estado final se definan claramente y exista un procedimiento de ejecución adecuado. Si se dan las dos condiciones, la programación declarativa es muy eficiente.

Como la programación declarativa no determina el “cómo”, sino que funciona a un nivel de abstracción muy alto, este paradigma deja margen para la optimización. Si se ha desarrollado un procedimiento de ejecución mejor, el algoritmo integrado lo encuentra y lo aplica. En este sentido, el paradigma está muy preparado para el futuro porque, al escribir el código, no es necesario determinar el procedimiento según el cual se alcanza el resultado.

Los lenguajes además de ser declarativos de subdividen en diferentes tipos:

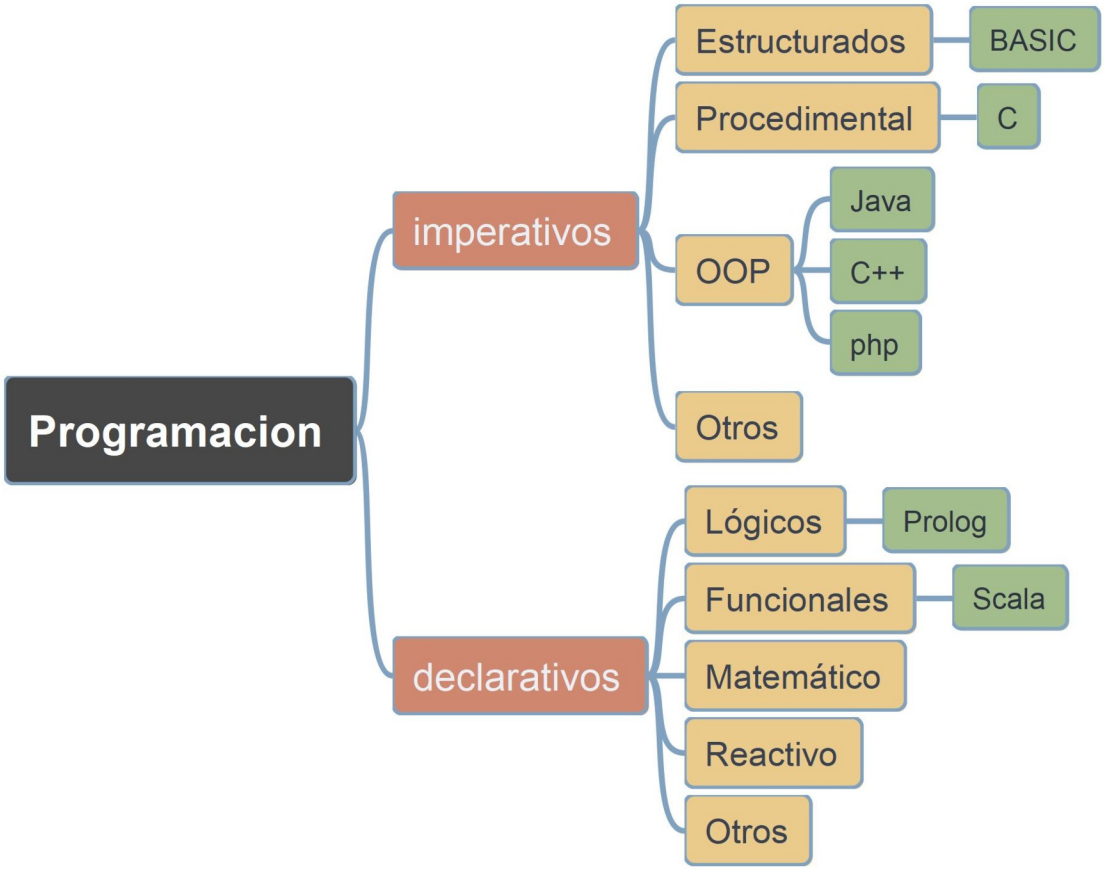
- Lógicos
- Funcionales
- Matemático
- Reactivo

Cada tipo de programación declarativa es a su vez un paradigma. La diferencia entre ellos depende del enfoque y sobre qué elementos se basan para construir sus algoritmos.

Los lenguajes de programación declarativa más conocidos son:

- Prolog
- Lisp
- Haskell
- Miranda
- Erlang
- SQL (en un sentido amplio)

2. Mapa mental con freeplane, imperativos y declarativos



3. Que es un sistema experto?

Los sistemas expertos (SE) son programas informáticos que tienen el objetivo de solucionar un problema concreto y utilizan la Inteligencia Artificial (IA) para simular el razonamiento de un ser humano. Se denominan sistemas expertos porque estos programas imitan la toma de decisiones de un profesional en la materia.

Actualmente, se consideran dentro del global de la Inteligencia Artificial. Se crearon durante la década de los 60 (aunque alcanzaron su mayor popularidad en los años posteriores) y fueron uno de los primeros sistemas de Inteligencia Artificial utilizados con éxito.

Tipos de sistemas expertos

- RBO (Rule Based Reasoning)
- CBR (Case Based Reasoning)
- Basados en Redes de Bayes

De forma más concreta, se pueden mencionar algunos sistemas que hicieron historia en el desarrollo de la Inteligencia Artificial. Es el caso de Dendral, un sistema experto desarrollado por Edward Feigenbaum y otros programadores en la Universidad de Stanford a mediados de la década de 1960, cuyo fin era interpretar estructuras moleculares y su desarrollo duró diez años. Posteriormente, se desarrolló el sistema Mycin, que estaba orientado a la investigación y determinación de diagnósticos de enfermedades infecciosas de la sangre.

Como anécdota, es conocido mundialmente el sistema experto Deep Blue, que fue desarrollado por IBM y que en 1997 logró derrotar en una partida de ajedrez a Garri Kasparov. Este fue uno de los primeros acercamientos del público general a las funcionalidades de un sistema experto y su aplicación en el mundo real.

4. Qué es CLIPS?

Desarrollado en el Centro Espacial Johnson de la NASA de 1985 a 1996, el Sistema de Producción Integrada en Lenguaje C (CLIPS) es un lenguaje de programación basado en reglas útil para crear sistemas expertos y otros programas donde una solución heurística es más fácil de implementar y mantener que una solución algorítmica. Escrito en C para mayor portabilidad, CLIPS se puede instalar y utilizar en una amplia variedad de plataformas. Desde 1996, CLIPS está disponible como software de dominio público.

CLIPS probablemente es el sistema experto más ampliamente usado debido a que es rápido, eficiente y gratuito. Aunque ahora es de dominio público, aún es actualizado y mantenido por su autor original, Gary Riley.

Los siguientes cambios se introdujeron en la versión 6.4.1 de CLIPS.

Nuevas funciones y comandos: se han agregado varias funciones y comandos nuevos. Ellos son:

- union$
    - La función union$ crea un valor multicampo que contiene una única aparición de cada valor contenido dentro de cada uno de sus argumentos multicampo.
- intersection$
    - La función intersection$ crea un valor multicampo que contiene una única aparición de cada valor contenido en todos sus argumentos multicampo.
- difference$ (ver sección 12.2.18)
    - La función difference$ crea un valor multicampo que contiene una única aparición de cada valor contenido en su primer argumento, pero ninguno de sus argumentos multicampo posteriores.

5. ¿Cual es el futuro de los sistemas expertos con el auge de la inteligencia artificial, es decir como afectará la IA a los SE?

Este capítulo contiene algunas reflexiones sobre inteligencia artificial (IA). En primer lugar, se explica la distinción entre la IA fuerte y la débil, así como los conceptos relacionados de IA general y específica, dejando claro que todas las manifestaciones existentes de IA son débiles y específicas. Se describen brevemente los principales modelos, insistiendo en la importancia de la corporalidad como aspecto clave para conseguir una IA de naturaleza general. A continuación se aborda la necesidad de proporcionar a las máquinas conocimientos de sentido común que hagan posible avanzar hacia el ambicioso objetivo de construir IA de tipo general. También se comentan las últimas tendencias en IA basadas en el análisis de grandes cantidades de datos que han hecho posibles progresos espectaculares en épocas muy recientes, con una alusión a las dificultades presentes hoy en los enfoques de la IA. Por último, se comentan otras cuestiones que son y continuarán siendo clave en la IA, antes de cerrar con una breve reflexión sobre los riesgos de la inteligencia artificial.

El camino hacia la IA realmente inteligente seguirá siendo largo y difícil, al fin y al cabo la IA tiene apenas sesenta años y, como diría Carl Sagan, sesenta años son un brevísimo momento en la escala cósmica del tiempo; o, como muy poéticamente dijo Gabriel García Márquez: «Desde la aparición de vida visible en la Tierra debieron transcurrir 380 millones de años para que una mariposa aprendiera a volar, otros 180 millones de años para fabricar una rosa sin otro compromiso que el de ser hermosa, y cuatro eras geológicas para que los seres humanos fueran capaces de cantar mejor que los pájaros y morirse de amor».

6. En el ejemplo en rojo que puedes encontrar mas abajo, haz que el código imprima "jirafa", ¿cómo lo has hecho?

```
(deffacts hechos-iniciales
(tiene-pelos)
(tiene-pezugnas)
(tiene-cuello-largo)) ; añadir esta línea

(defrule mamifero-1
(tiene-pelos)
=>
(assert (es-mamifero)))

(defrule mamifero-2
(da-leche)
=>
(assert (es-mamifero)))

(defrule ungulado-1
(es-mamifero)
(tiene-pezugnas)
=>
(assert (es-ungulado)))

(defrule ungulado-2
(es-mamifero)
(rumia)
=>
(assert (es-ungulado)))

(defrule jirafa
```

```
(es-ungulado)
(tiene-cuello-largo)
=>
(printout t "Es una jirafa" crlf))

(defrule cebra
(es-ungulado)
(tiene-rayas-negras)
=>
(printout t "Es una cebra" crlf))
```