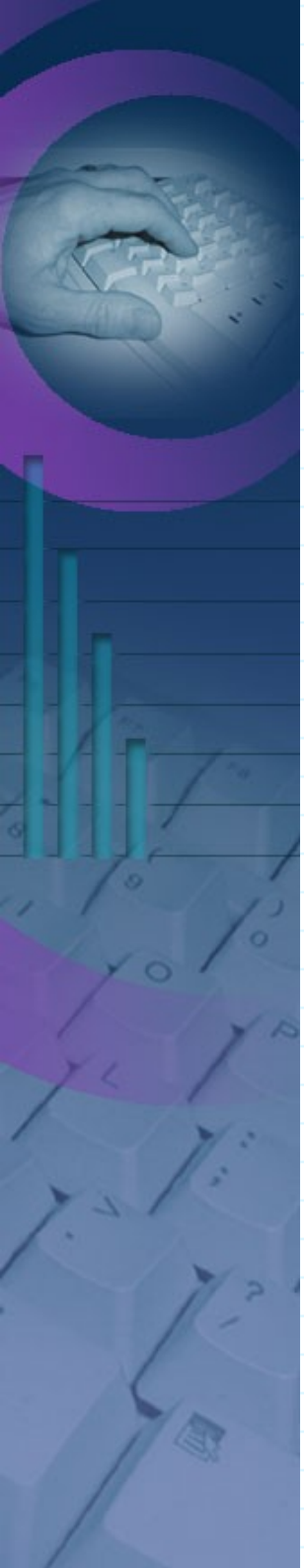


Tecnologías de Programación

Lenguaje de Programación Lógica

ProLog - Sintáxis

Prolog

Elementos del Lenguaje

- Hechos
  - Son declaraciones axiomáticas
  - Para decir que maría es progenitora de juan, escribimos:
    - progenitor(maría, juan).
  - Otros ejemplos:
    - valioso(oro)
    - mas_cercano_al_sol(mercurio, venus).
    - mas_cercano_al_sol(venus, tierra).

Prolog

Elementos del Lenguaje

- Reglas
  - Representan relaciones que se definen en base a otras relaciones, por ejemplo:
    - abuelo(X, Y):-hijo(X, Z), hijo(Z, Y).

    abuelo(X, Y) es la cabecera de la regla

    hijo(X, Z), hijo(Z, Y) es el cuerpo de la regla
  - Para demostrar la veracidad de la cabecera se debe demostrar la veracidad del cuerpo

Prolog

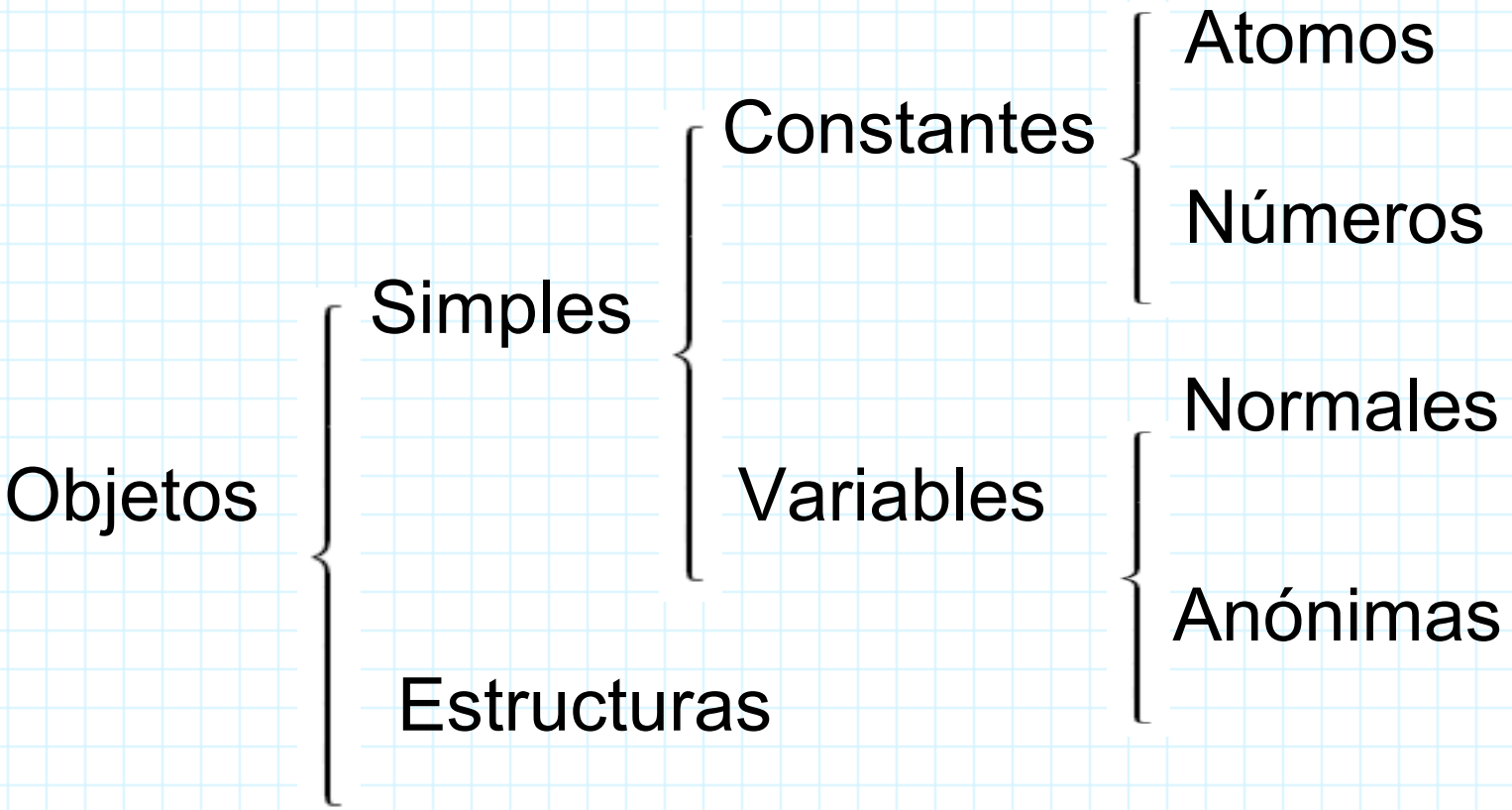
Elementos del Lenguaje

- Objetivos
  - Son declaraciones usadas para indagar la base de conocimientos expresada por el programa.
  - Si quisiéramos saber quienes son los nietos de carlos, podríamos plantear el siguiente objetivo:
    - abuelo(carlos, X).

    Si hay algun hecho o cabecera de regla que unifique con los datos suministrados, la evaluación será exitosa, y las variables libres quedarán ligadas a valores concretos

Prolog

Elementos del Lenguaje



Prolog

Elementos del Lenguaje

- Constantes: Representan objetos del dominio en forma puntual
  - juan, carlos, 2
- Variables: Representan objetos del dominio en forma no puntal
  - X, Persona, _ciudad, _ (variable anónima)

  El alcance de una variable es la cláusula donde aparece.

Prolog

Elementos del Lenguaje

- Estructuras: Representan objetos a partir de sus componentes (functores). La sintaxis es la misma que para los hechos.
  - Por ejemplo:
    - punto(X, Y)
    - segmento(punto(X1, Y1), punto(X2, Y2))
    - triangulo(punto(X1, Y1), punto(X2, Y2), punto(X3, Y3))
  - Lo podríamos usar de la siguiente forma:
    - vertical(segmento(punto(X,Y1), punto(X,Y2))).

Prolog

Sintaxis

- Atomos: comienzan con una letra, pueden contener caracteres especiales y ser nombres entre comillas simple
- Números: son enteros o reales sin una definición explícita de tipos
- Variables: comienzan con mayúscula o subrayado. Se denota con el símbolo del subrayado a la variable anónima
    - es_padre(X) :- progenitor(X, _).

Prolog

Sintaxis

- Estructuras: constan de un functor cuyo primer carácter debe ser una letra minúscula y argumentos que pueden ser constantes, variables u otros functores.
- Hechos: tienen la siguiente sintaxis
  - hecho(c1, c2, ..., cN).
- Reglas: tienen la siguiente sintaxis
  - cabecera(c1, c2, ..., cN):-

    proposición1(p11, p12, ..., p1N), ...,

    proposiciónM(pM1, pM2, ..., pMN).

Prolog

Significado declarativo y procedural

- En un lenguaje declarativo puro, sería de esperar que el orden en el que aparecen los hechos y las reglas en la base fuera independiente de los datos, sin embargo en PROLOG no es así
- Los cálculos y unificaciones se realizan en forma procedural, por lo que el orden de los predicados, y en que se declaran las proposiciones de los mismos, influyen en el desarrollo
  - X=4, Y=2, X>Y. no es lo mismo que escribir X=4, X>Y, Y=2.

  En particular el segundo caso falla, ¿por que?

Prolog

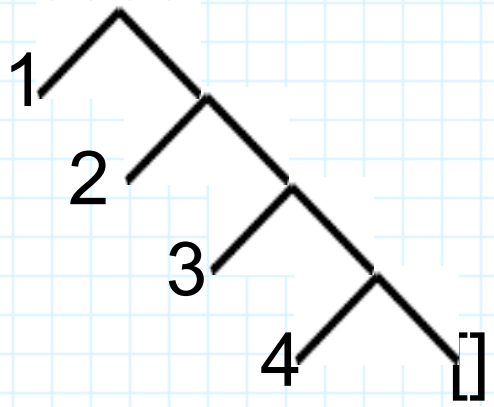
Listas

- Es una estructura de datos simple que denota una secuencia de elementos tales como: 1, 2, 3, 4, 5
- La sintaxis para denotar una lista es encerrar los elementos separados por coma entre corchetes: [1, 2, 3, 4, 5]
- Internamente se representa mediante un árbol binario donde la rama de la derecha representa el primer elemento o cabeza, y la rama de la izquierda representa el resto o cola de la lista

Prolog

Listas

- Así, la lista [1, 2, 3, 4] se representa como:



donde la lista [] es la lista vacía

- Aunque no es necesario, ProLog representa las listas a través del functor “.”
  - Así [1, 2, 3, 4] es equivalente a .(1, .(2, .(3, .(4, []))))

Prolog

Listas

- La cabeza y la cola se pueden separar mediante el símbolo “|”, así las siguientes expresiones se refieren a la misma lista:
  - [a,b,c]
  - [a|[b,c]]
  - [a,b|[c]]
  - [a,b,c|[]]

Prolog

Listas

- Ejemplo de uso:
  - Determinar si un elemento es miembro de una lista:

```
miembro(X,[X|_]).
miembro(X,[_|R]):-miembro(X,R).
```

Prolog

Listas

- Ejemplo de uso:
  - Concatenar dos listas:

```
concatenar ([],L,L).
concatenar ([X|L1],L2,[X|L3]):- concatenar (L1,L2,L3).
```

Prolog

Listas

- Ejemplo de uso:
  - Dado una lista calcular su inversa:

```
inversa([],[]).
inversa([X|L1],L):-
                    inversa(L1,Resto),
                    concatenar(Resto,[X],L).
```

Prolog

Construcción de expresiones aritméticas

- X + Y
- X – Y
- X * Y
- X / Y o X div Y
- X mod Y
- X^Y
- -X
- abs(X)
- acos(X)
- asen(X)
- atan(X)
- cos(X)
- exp(X)
- ln(X)
- log(X)
- sin(X)
- sqrt(X)
- tan(X)
- round(X,N)

Prolog

Comparación de términos

- X<Y /*X es menor que Y*/
- X>Y /*X es mayor que Y*/
- X=<Y /*X es menos o igual que Y*/
- X>=Y /*X es mayor o igual que Y*/
- X=Y /*X es igual que Y*/
- X\=Y /*X es distinto que Y*/

Prolog

Comparación de expresiones

- Una expresión es un conjunto de términos unidos por operadores aritméticos. Los siguientes operadores comparan expresiones en forma sintáctica
  - X==Y /*la expresión X es igual que la expresión Y*/
  - X\==Y /*la expresión X es distinta que la expresión Y*/
  - X@<Y /*la expresión X es menor que la expresión Y*/
  - X@>Y /*la expresión X es mayor que la expresión Y*/
  - X@=<Y /*la expresión X es menor o igual que la expresión Y*/
  - X@>=Y /*la expresión X es mayor o igual que la expresión Y*/

Prolog

Evaluación de expresiones

- El siguiente operador permite asignar la evaluación de una expresión
  - X is Y
  - Ej.: X is (3*5+3)/2 /*PROLOG contestaría X=9*/
- Los siguientes operadores comparan la evaluación de dos expresiones
  - X=:=Y /* El resultado de evaluar la expresión X es igual al resultado de evaluar la expresión Y*/
  - X=\=Y /* El resultado de evaluar la expresión X es distinto al resultado de evaluar la expresión Y*/

Prolog

Ejemplos

- Encontrar la cantidad de números positivos y negativos de una lista:

```
positivo(X) :- X >= 0.
negativo(X) :- X < 0.

evaluar([], 0, 0).
evaluar([X|Lista], Pos, Neg) :-
                        positivo(X),
                        evaluar(Lista, Pos1, Neg),
                        Pos is Pos1 + 1.

evaluar([X|Lista], Pos, Neg) :-
                        negativo(X),
                        evaluar(Lista, Pos, Neg1),
                        Neg is Neg1 + 1.
```