Universidad Andrés Bello

Lisp y Prolog

...

Jean Contreras

Sistemas Inteligentes

Instalación en Linux

Universidad Andrés Bello

LISP:

```
$ sudo apt-get install clisp
```

PROLOG:

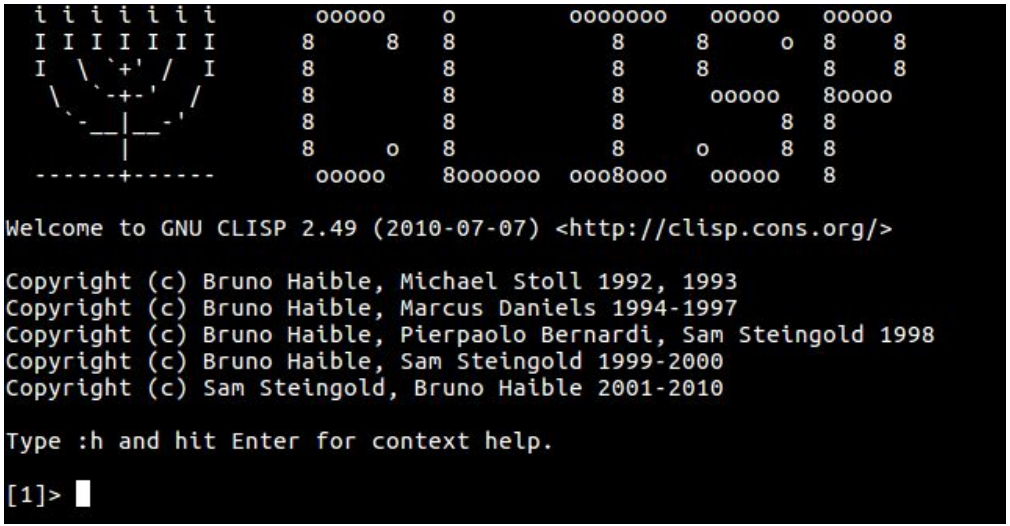
```
$ sudo add-apt-repository ppa:swi-prolog/stable
$ sudo apt-get install swi-prolog
```

Lisp

Universidad Andrés Bello

Iniciar lisp:

$ clisp



Lisp

Universidad Andrés Bello

Cargar programa lisp:

> (load “cuadrado.lisp”)

```
[1]> (load "cuadrado.lisp")
;; Loading file cuadrado.lisp ...
;; Loaded file cuadrado.lisp
T
```

***Ejecutar programa:**

> (cuad 5)

```
[2]> (cuad 5)
25
```

Salir:

> (quit)

```
Break 1 [2]> (quit)
Adiós.
```

*Se llama a la función principal del programa, no al nombre del archivo.

Lisp

Universidad Andrés Bello

Hola Mundo:

```
; hello world lisp program.
(print "Hello World")
```

```
[1]> (load "hello.lisp")
;; Loading file hello.lisp ...
"Hello World"
;; Loaded file hello.lisp
T
```

Cuadrado de un número:

```
(DEFUN CUAD (N)
  (* N N))
```

```
[1]> (load "cuadrado.lisp")
;; Loading file cuadrado.lisp ...
;; Loaded file cuadrado.lisp
T
[2]> (cuad 5)
25
```

Lisp

Universidad Andrés Bello

Mover la cabeza de una lista al final:

ABCD -> BCDA

```
(DEFUN ROTAR-IZQ (L)
 (APPEND (CDR L) (LIST (CAR L))))
```

```
[1]> (load "rotarIzq.lisp")
;; Loading file rotarIzq.lisp ...
;; Loaded file rotarIzq.lisp
T
[2]> (rotar-izq '(A B C D))
(B C D A)
```

Las funciones CAR y CDR devuelven la cabeza y la cola de la lista, respectivamente.

APPEND une dos listas.

Lisp

Universidad Andrés Bello

Factorial:

```
(DEFUN FACT (N)
 (COND ((= N 1) 1)
      (T (* N (FACT (- N 1)))) ))
```

```
[1]> (load "factorial.lisp")
;; Loading file factorial.lisp ...
;; Loaded file factorial.lisp
T
[2]> (fact 5)
120
```

COND: Condición de término, retorna 1.

T: “Else”.

Lisp

Universidad Andrés Bello

Lista:

```
(DEFUN SEGUNDO (L)
 (CAR (CDR L)))

(DEFUN TERCERO (L)
 (CAR (CDR (CDR L))))

(DEFUN PENULTIMO (L)
  (NTH (- (LENGTH L) 2) L))
```

ABCD -> BCD -> B

ABCD -> BCD -> CD -> C

LENGTH = 4
NTH(4-2) = NTH(2) = C

NTH devuelve el nth elemento de una lista partiendo desde cero.

Lisp

Universidad Andrés Bello

Lista:

```
(DEFUN SEGUNDO (L)
 (CAR (CDR L)))

(DEFUN TERCERO (L)
 (CAR (CDR (CDR L))))

(DEFUN PENULTIMO (L)
  (NTH (- (LENGTH L) 2) L))
```

```
[1]> (load "listas.lisp")
;; Loading file listas.lisp ...
;; Loaded file listas.lisp
T
[2]> (segundo '(A B C D))
B
[3]> (tercero '(A B C D))
C
[4]> (penultimo '(A B C D))
C
```

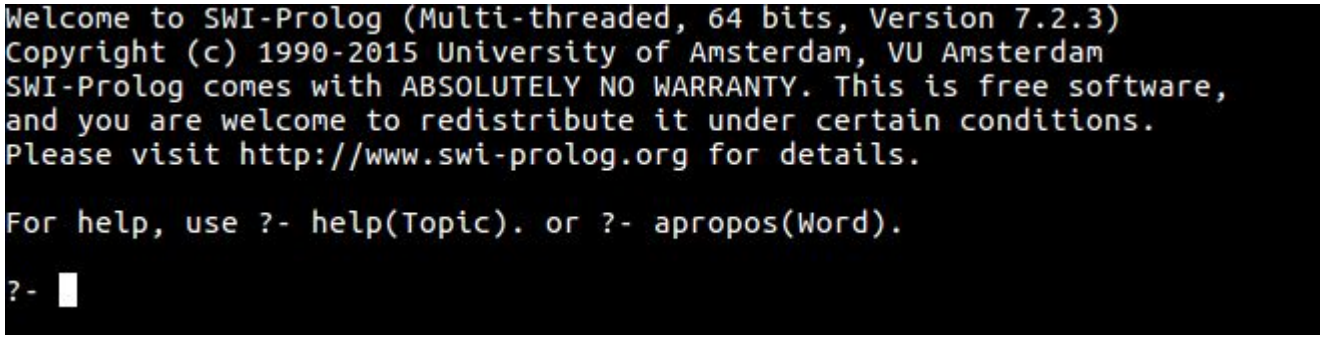
NTH devuelve el nth elemento de una lista partiendo desde cero.

Prolog

Universidad Andrés Bello

Iniciar prolog:

$ prolog



Prolog

Universidad Andrés Bello

Cargar programa prolog:

?- consult("parentezco").

```
?- consult("parentezco").
true.
```

Ejecutar programa:

?- progenitor(pedro,jose).

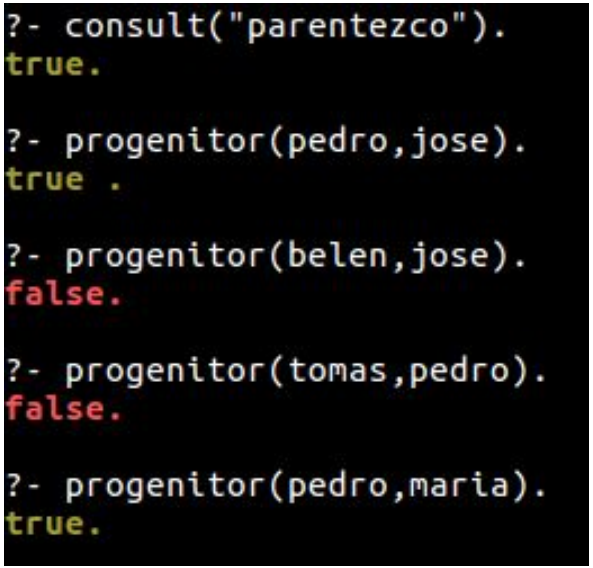
```
?- progenitor(pedro,jose).
true .
```

Prolog

Universidad Andrés Bello

Pertenencia de un valor en una lista:

```
progenitor(pilar,belen).
progenitor(tomas,belen).
progenitor(belen,pedro).
progenitor(pedro,jose).
progenitor(pedro,maria).
```



Prolog

Universidad Andrés Bello

Longitud de una lista:

```
longitud([],0).

longitud([_|L],N):-longitud(L,N1), N is N1 + 1.
```

Hecho, condición de término.

Regla, :- (si ...)

[_|L] es una Lista, separado en cabeza y cola.

```
?- consult("longitud").
true.

?- longitud([a,b,c],N).
N = 3.

?- longitud([a,b,c,d],N).
N = 4.

?- longitud([1,2,3,4,6],N).
N = 5.
```

Prolog

Universidad Andrés Bello

Pertenencia de un valor en una lista:

```
pertenece(X,[X|_]).

pertenece(X,[_|L]):- pertenece(X,L).
```

X: variable a buscar.
pertenece(X,[X|_]). : Objetivo (True or False).

```
?- consult("pertenece").
true.

?- pertenece(1, [1,2,3,4,6]).
true .

?- pertenece(7, [1,2,3,4,6]).
false.

?- pertenece(a, [a,b,c]).
true .

?- pertenece(d, [a,b,c]).
false.
```

Prolog

Universidad Andrés Bello

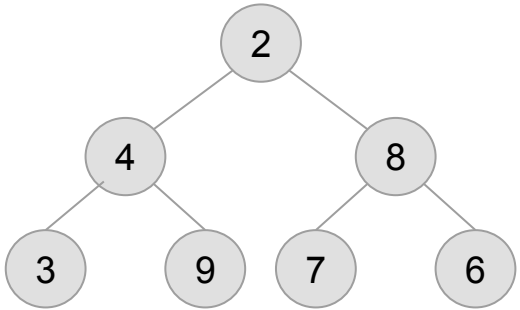
Máximo valor de un árbol:

```
maximo(nil,0).

maximo(t(I,R,D),M):- maximo(I,MI),maximo(D,MD),M1 is max(MI,MD), M is max(R,M1).
```

nil = Árbol vacío

t(t(t(nil,3,nil), 4, t(nil,9,nil)), 2, t(t(nil,7,nil), 8, t(nil,6,nil)))



Prolog

Universidad Andrés Bello

Máximo valor de un árbol:

```
maximo(nil,0).

maximo(t(I,R,D),M):- maximo(I,MI),maximo(D,MD),M1 is max(MI,MD), M is max(R,M1).
```

```
?- consult("maximo_arbol").
true.

?- maximo(t( t( t(nil,3,nil), 4, t(nil,9,nil)), 2, t( t(nil,7,nil), 8, t(nil,6,nil))),M).
M = 9.
```