Alumno:

Juan Cañar

Docente:

Ing. Diego Quisi.

Materia:

Sistemas Expertos

Ciclo:

9no

Fecha:

27/04/2020

Numero primo

```
-----Ver si un numero es o no primo-------
(deffunction numPrimo (?x)
	(bind ?cont 0)
	(bind ?aux (- ?x 1))
	(loop-for-count(?i 2 ?aux)
		(if (=(mod ?x ?i) 0)
		then
			(if (neq ?x 2)
			then
				(bind ?cont 1)
			)
		)
	)
	(if (= ?cont 1)
	then
		(printout t "El numero " ?x " no es primo." crlf)
	else
		(printout t "El numero " ?x " es primo." crlf)
	)
)
( printout t "Respuesta " (numPrimo 4)crlf)
( printout t "Respuesta " (numPrimo 5)crlf)
```

Fibonacci

```
(deffunction
fibo (?a)
                    (printout t ?a "" crlf)
                 (if (or (= ?a 0) (= ?a 1)) then
                           (bind ?a 1)
                           (printout t ?a "" crlf)
                           (bind ?a 0)
                           (printout t ?a "" crlf)
                    else

                           (+ (fibo(- ?a 1)) (fibo(- ?a 2)))))

              (printout t "Fibonacci "(fibo 4)crlf)
              ------------------------------------------------
```

```
(deffunction fi (?n)
        (if (< ?n 2) then
                ?n
        else
            (+ (fi(- ?n 1)) (fi(- ?n 2))))
)
(deffunction fibo (?n)
    (loop-for-count( ?cont 0 (- ?n 1) ) do
        (printout t "" (fi ?cont) crlf)
    )
)
(printout t "" (fibo 5) crlf)
```

Ordenar de Mayor a Menor

```
       (deftemplate MAIN::vect
   (slot index (type INTEGER))
   (slot value (type INTEGER)))
(deffacts MAIN::vector
   (vect (index 1) (value 25))
   (vect (index 2) (value 3))
   (vect (index 3) (value 33))
   (vect (index 6) (value 5))
   (vect (index 7) (value 5))
   (vect (index 8) (value 60))
   (vect (index 9) (value 4)))
(defrule MAIN::sort-bubble
   ?f1 <- (vect (index ?p1) (value ?v1))
   ?f2 <- (vect (index ?p2&:(= ?p2 (+ ?p1 1))) (value ?v2&:(< ?v2 ?v1)))
   =>
   (modify ?f1 (value ?v2))
   (modify ?f2 (value ?v1)))
(defrule MAIN::print-res
   (declare (salience -10))
   ?f1 <- (vect (index ?p1) (value ?v))
   (forall
        (vect (index ?p2))
        (test (<= ?p1 ?p2)))
   =>
   (printout t ?v crlf)
   (retract ?f1))
```

Echos Activa Letras

```
(deffacts MAIN::echosLetras
   (H)
   (K))

(defrule MAIN::Letras-B
   (B)
   =>
   (assert (D))
   (printout t "*Activa -- D --*" crlf))

(defrule MAIN::Letras-H
   (H)
   =>
   (assert (A))
   (printout t "*Activa -- A --*" crlf))

(defrule MAIN::Letras-A
   (A)
   =>
   (assert (E))
   (printout t "*Activa -- E --*" crlf))

(defrule MAIN::Letras-EG
   (E)
   (G)
   =>
   (assert (C))
   (printout t "*Activa -- C --*" crlf))

(defrule MAIN::Letras-EK
   (E)
   (K)
   =>
   (assert (B))
   (printout t "*Activa -- B --*" crlf))

(defrule MAIN::Letras-DEK
   (D)
   (E)
   (K)
```

```
   =>
   (assert (C))
   (printout t "*Activa -- C --*" crlf))

(defrule MAIN::Letras-GKF
   (G)
   (K)
   (F)
   =>
   (assert (A))
   (printout t "*Activa -- A --*" crlf))
```

Triangulo

```
(defrule tipo-triangulo
(initial-fact)
=>
(printout t "-----------------------------------------------" crlf)
(printout t "Quieres saber el tipo de Triangulo (si/no)? " crlf
(printout t "-----------------------------------------------" crlf))
(assert (entrada (read)))
)

(defrule ingrese-tipo
(entrada si)
=>
(printout t "Ingrese numero de lados? " (printout t "*Recuerda debe tener 3!!*"
crlf))
(assert (lados(read)))
)

;-- ~ es igual que  "&", "|" (lados ?l&~:(integerp ?l)|:(> ?l 3))
(defrule valida-triangulo
(or(lados ?l &:(integerp ?l)&:(> ?l 3))(lados ?l &:(integerp ?l)&:(<= ?l 2)))
=>
(printout t "_____________________________________________" crlf)
(printout t "No es un triangulo" crlf)
(printout t "RECUERDA QUE SOLO TIENEN TRES LADOS!!!" crlf)
(printout t "_____________________________________________" crlf)
)
```

```
(defrule tipo
(lados 3)
=>
(printout t "1.Segun sus lados 2.Segun sus angulos" crlf)
(assert (ve(read))))

(defrule r1
(and(lados 3)(ve 1))
=>
(printout t "Tiene los 3 lados iguales" crlf)
(printout t "si/no" crlf)
(assert (res1(read))))

(defrule r2
(res1 si)
=>
(printout t "Todos los ángulos son iguales y suman 180º? " crlf
(printout t "α=60º ?  β=60º ? γ=60º ? "crlf))
(printout t "si/no" crlf)
(assert (res2(read))))

(defrule r3
(res2 si)
=>
(printout t "**--TRIANGULO EQUILATERO--**: tiene todos sus lados y los tres
angulos iguales." crlf
(printout t "α=60º -- β=60º -- γ=60º "crlf))
(printout t "ok" crlf)
(assert (res(read))))

(defrule r4
(res1 no)
=>
(printout t "Tiene dos lados iguales y ? " crlf
(printout t "Dos de sus ángulos también son iguales? "crlf))
(printout t "si/no" crlf)
(assert (res3(read))))

(defrule r5
(res3 si)
=>
(printout t "**--TRIANGULO iSOSCELES--**" crlf
```

```
(printout t "Tiene dos lados iguales dos de sus ángulos también son iguales
"crlf))
(printout t "ok" crlf)
(assert (res(read))))

(defrule r6
(or(res3 no) (res2 no))
=>
(printout t "Los tres lados son desiguales y." crlf
(printout t "Los tres ángulos también son diferentes. "crlf))
(printout t "si/no" crlf)
(assert (res4(read))))

(defrule r7
(res4 si)
=>
(printout t "**--TRIANGULO ESCALENO--**" crlf
(printout t "Los tres lados son desiguales."crlf))
(printout t "por lo que los tres ángulos también son diferentes " crlf)
(printout t "ok" crlf)
(assert (res(read))))

;-- and
(defrule r8
(or(res4 no)(ve 2))
=>
(printout t "**-----------------------------------------------------------------------
----------------------**" crlf)
(printout t "**--TRIANGULO RECTANGULO--**" crlf
(printout t "Uno de sus ángulos es de 90º " crlf
(printout t "Los otros dos son agudos menores de 90º "crlf)))
(printout t "-----------------------------------------------------------------------
-------------------------" crlf)
(printout t
"_____________________________________________________________________________________
_____________" crlf)
(printout t
"_____________________________________________________________________________________
_____________" crlf)
(printout t "-----------------------------------------------------------------------
-------------------------" crlf)
(printout t "**--TRIANGULO OBLICUANGULO--**"  crlf
(printout t " No tiene ningun ángulo recto de 90°" crlf
```

```
(printout t "---->Triángulo acutangulo: Los tres angulos son agudos , menores de
90º" crlf
(printout t "---->Triángulo obtusangulo: Uno de sus angulos es mayor a 90º. Los
otros dos son agudos -menores de 90º" crlf))))
(printout t "-------------------------------------------------------------------
--------------------------" crlf)
(printout t "Press ok" crlf)
(assert (res(read))))

(defrule r11
(or(res4 no)(res5 no))
=>
(printout t "**--No existe triangulo--**" crlf
(printout t "Prueba de nuevo."crlf))
(printout t "ok" crlf)
(assert (res(read))))
```