ENTREGA 1 PRÁCTICAS INGENIERÍA DEL CONOCIMIENTO

UNIVERSIDAD DE GRANADA

David Muñoz Sánchez

Índice

Práctica 1:

Realiza un sistema en CLIPS que recoja el conocimiento sobre los miembros de una familia (por ejemplo la tuya, o una familia Real o una familia de ficción) y que indique quiénes son los miembros que están relacionados por una relación concreta R con un miembro concreto de la familia M. Por ejemplo, con la familia del ejemplo el sistema tendría que responder cuestiones como:

- ¿Quiénes son los abuelos de Juanito?

- ¿Quiénes son los yernos de Mercedes?

- ¿Quiénes son los hijos de Lidia?

En el caso de que no haya ningún miembro de la familia relacionado mediante R con M, el sistema debe indicarlo.

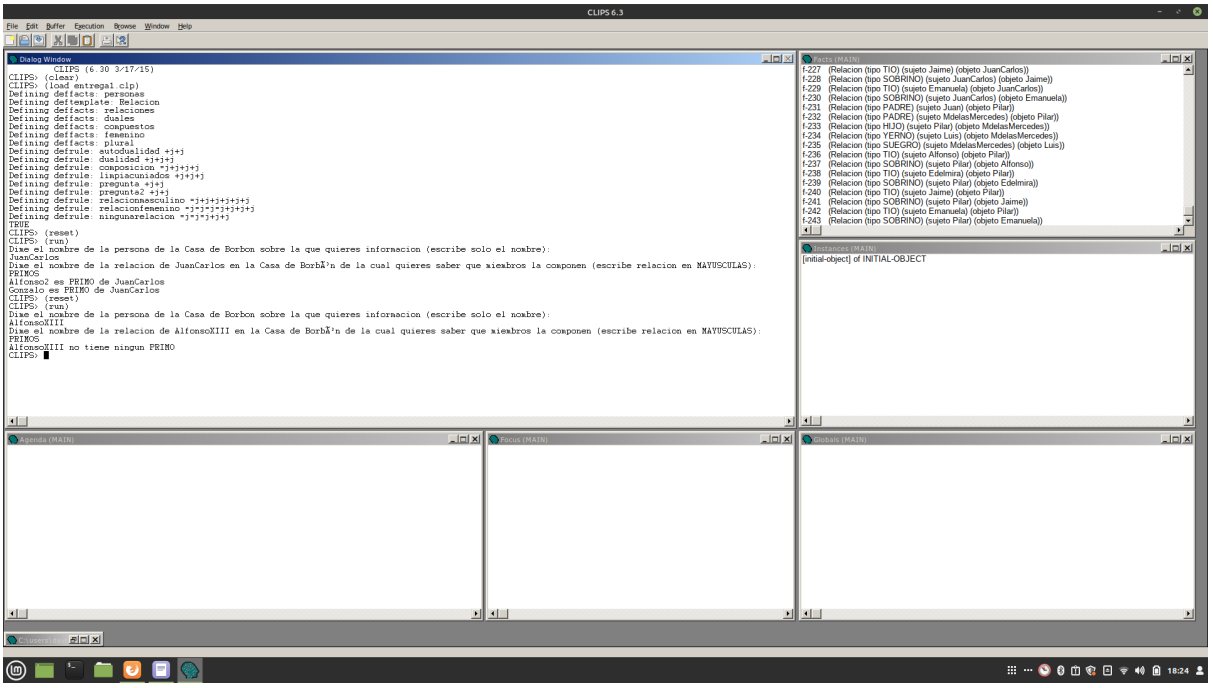
Esta práctica se ha hecho sobre el árbol genealógico de la Casa de Borbón en España, desde Alfonso XIII a Juan Carlos I. Se han introducido solo unos cuantos ejemplos porque bien es sabido que las familias reales son bastante inmensas. A continuación se indican las personas introducidas en CLIPS:

```
(deffacts personas
   (hombre AlfonsoXIII) ; "AlfonsoXIII es un hombre"
   (hombre Alfonso)
   (hombre Jaime)
   (hombre Juan)
   (hombre Alfonso2)
   (hombre Gonzalo)
   (hombre JuanCarlos)
   (hombre Luis)
   (mujer VictoriaEugenia)
   (mujer Edelmira)
   (mujer Emanuela)
   (mujer MdelasMercedes)
   (mujer Carmen)
   (mujer Mercedes)
   (mujer Pilar)
   (mujer Sofia) )
```

```
(deffacts relaciones
   (Relacion (tipo HIJO) (sujeto Alfonso) (objeto AlfonsoXIII)) ;
"Alfonso es HIJO de AlfonsoXIII"
   (Relacion (tipo HIJO) (sujeto Jaime) (objeto AlfonsoXIII))
   (Relacion (tipo HIJO) (sujeto Juan) (objeto AlfonsoXIII))
   (Relacion (tipo HIJO) (sujeto Alfonso2) (objeto Jaime))
   (Relacion (tipo HIJO) (sujeto Gonzalo) (objeto Jaime))
```

```
   (Relacion (tipo HIJO) (sujeto JuanCarlos) (objeto Juan))
   (Relacion (tipo HIJO) (sujeto Pilar) (objeto Juan))
   (Relacion (tipo ESPOSO) (sujeto AlfonsoXIII) (objeto
VictoriaEugenia)) ; "AlfonsoXIII es ESPOSO de VictoriaEugenia"
   (Relacion (tipo ESPOSO) (sujeto Alfonso) (objeto Edelmira))
   (Relacion (tipo ESPOSO) (sujeto Jaime) (objeto Emanuela))
   (Relacion (tipo ESPOSO) (sujeto Juan) (objeto MdelasMercedes))
   (Relacion (tipo ESPOSO) (sujeto Alfonso2) (objeto Carmen))
   (Relacion (tipo ESPOSO) (sujeto Alfonso2) (objeto Carmen))
   (Relacion (tipo ESPOSO) (sujeto Gonzalo) (objeto Mercedes))
   (Relacion (tipo ESPOSO) (sujeto Luis) (objeto Pilar))
   (Relacion (tipo ESPOSO) (sujeto JuanCarlos) (objeto Sofia)))
```

Aquí solo hemos indicado hijos y matrimonios, pero los demás parentescos los deducirá automáticamente CLIPS. A continuación, se muestra una ejecución preguntando por los primos de Juan Carlos, lo cual nos tiene que devolver Alfonso2 y Gonzalo. Además, si preguntamos por los primos de AlfonsoXIII, nos debería devolver que no tiene (todo en base a lo que hemos metido).



Se han añadido reglas para preguntar por la persona y por el parentesco (ambas con prioridad positiva, siendo mayor la primera que la segunda para asegurar que siempre pregunte antes de hacer nada más). Se ha añadido además, hechos para los plurales de los parentescos y tenemos ya los femeninos y masculinos. Por último, si no hay ninguna relación, lo indica, y esto siempre se ejecuta al final del todo por si se nos escapara alguna relación (por eso tiene prioridad negativa).

Práctica 2:

Considerar el juego del 4 en raya o conecta 4 (https://www.ludoteka.com/clasika/4-en-raya.html). En Prado podéis encontrar un CLP con un sistema basado en reglas de CLIPS que juega al cuatro en raya contra un humano. En este fichero encontraréis reglas para que el sistema juegue en una columna aleatoria. Esta actividad consiste en que añadáis hechos y reglas para que el sistema razone y juegue como lo haríais vosotros, de forma que pueda ser utilizado por un niño para aprender a jugar, razonando las jugadas.

Lo primero antes de proporcionar reglas y hechos para que la máquina juegue con sentido ha sido implementar los ejercicios propuestos en Prado.

En el primero, añado reglas que añaden hechos para calcular la posición siguiente y anterior a todas las posiciones del tablero. El hecho tiene la forma **siguiente/anterior fila1 col1 orientación fila2 columna2.** Se ha hecho en todas las orientaciones posibles, vertical, horizontal, diagonal y diagonal al revés.

```
;;;;;;EJERCICIO1: aniado reglas para deducir la posicion siguiente y
anterior a una posicion

;;;siguiente posicion h
(defrule siguienteh
(Tablero ?t ?f ?c ?j)
(test(< ?c 7))
=>
(assert(siguiente ?f ?c h ?f (+ ?c 1)))
)

;;;anterior posicion h
(defrule anteriorh
(Tablero ?t ?f ?c ?j)
(test(> ?c 1))
=>
(assert(anterior ?f ?c h ?f (- ?c 1)))
)

;;;siguiente posicion v
(defrule siguientev
(Tablero ?t ?f ?c ?j)
(test(> ?f 1))
=>
(assert(siguiente ?f ?c v (- ?f 1) ?c))
)
```

```
;;;anterior posicion v
(defrule anteriorv
(Tablero ?t ?f ?c ?j)
(test(< ?f 6))
=>
(assert(anterior ?f ?c v (+ ?f 1) ?c))
)

;;;siguiente posicion diagonalmente
(defrule siguiented
(Tablero ?t ?f ?c ?j)
(test(< ?c 7))
(test(> ?f 1))
=>
(assert(siguiente ?f ?c d (- ?f 1) (+ ?c 1)))
)

;;;anterior posicion diagonalmente
(defrule anteriord
(Tablero ?t ?f ?c ?j)
(test(> ?c 1))
(test(< ?f 6))
=>
(assert(anterior ?f ?c d (+ ?f 1) (- ?c 1)))
)

;;;siguiente posicion diagonal inversa
(defrule siguientedi
(Tablero ?t ?f ?c ?j)
(test(< ?c 7))
(test(< ?f 6))
=>
(assert(siguiente ?f ?c di (+ ?f 1) (+ ?c 1)))
)

;;;anterior posicion diagonal inversa
(defrule anteriordi
(Tablero ?t ?f ?c ?j)
(test(> ?c 1))
(test(> ?f 1))
=>
(assert(anterior ?f ?c di (- ?f 1) (- ?c 1)))
)
```

En el segundo ejercicio, se añaden reglas para que el sistema deduzca dónde caería una ficha según la columna en la que se

introduzca. Tenemos una primera regla para el caso en el que el tablero tiene la posición 6 x, es decir, última fila y cualquier columna, sin ocupar, así que esas serán las primeras posiciones donde las fichas del cuatro en raya pueden caer. Para los demás casos, partimos de un caería existente (siempre habrá puesto que se añaden los iniciales), y si para **caeria fila columna**, en esa fila y columna hay M o J, eliminamos el caería y añadimos otro con una fila menos. Por último, también se contempla el caso en el que una fila ya esté llena (por eso la comprobación test(eq ?f 1)). Es decir, la fila del caería es 1 y esa posición en el tablero ya está ocupada por algún jugador.

```
;;;EJERCICIO 2
;;;Hay que aniadir reglas para que el sistema deduzca donde caeria la
ficha segun la columna
;;;en la que se introduzca

;;;Si la columna esta vacia
(defrule caeria_ini
(Tablero ?t 6 ?c _)
=>
(assert(caeria 6 ?c)) ;;;Siempre caeria en la fila 6, por estar vacia la
columna
)

;;;Para los demas casos, si la posicion donde cae esta bien
(defrule caeria
?regla <- (caeria ?f ?c) ;;;Para posteriormente eliminarla, no nos sirve
si la va a ocupar
(Tablero ?t ?f ?c ?j)
(test (neq ?j _)) ;;Para eliminarla tiene que estar ocupada por M o J
(test (> ?f 1)) ;;Comprobacion que nos asegura que la columna no esta
llena
=>
(retract ?regla)
(assert (caeria (- ?f 1) ?c))
)

;;;Si la columna esta llena, esta regla ya no se aniade porque la fila
no es mayor que 1, ahora
;;;debemos eliminar la regla anterior
(defrule eliminar_caeria
?regla <- (caeria ?f ?c)
(Tablero ?t ?f ?c ?j)
(test (neq ?j _)) ;;Para eliminarla tiene que estar ocupada por M o J
(test (eq ?f 1))
=>
```

```
(retract ?regla)
)
```

En el ejercicio número 3, vemos una regla para añadir el hecho **conecta2 t orientación fila1 col1 fila2 col2** jugador. Este hecho se añade cuando el sistema deduce que en la posición siguiente a una dada, hay una ficha del mismo jugador. Aquí se podría hacer la comprobación tanto como con siguiente, como con la posición anterior, puesto que el sistema comprobará todas las posiciones.
El 4 es análogo al anterior pero buscando líneas de tres fichas iguales. Para ello hay dos versiones, una con siguiente, y otra con anterior. En ambos se usan los hechos conecta2 puesto que previo a que estén tres en línea, debe haber dos en línea. Además, en la versión anterior, se cambia el inicio de la secuencia al nuevo elemento y en la siguiente, se cambia el final a la posición del nuevo elemento. El hecho tiene la forma **conecta3 t orientación fila1 col1 fila2 col2 jugador.**

```
;;;EJERCICIO 4
;;;Igual que antes pero con 3 en linea
(defrule conecta3_1
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
(Tablero ?t ?f3 ?c3 ?jugador3)
(test (neq ?jugador3 _))
(test (eq ?jugador ?jugador3)) ;; 2 en linea del mismo jugador y la
tercera tambien
(siguiente ?f2 ?c2 ?d ?f3 ?c3) ;;
=>
(assert (conecta3 ?t ?d ?f1 ?c1 ?f3 ?c3 ?jugador3))
)

(defrule conecta3_2
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
(Tablero ?t ?f3 ?c3 ?jugador3)
(test (neq ?jugador3 _))
(test (eq ?jugador ?jugador3)) ;; 2 en linea del mismo jugador y la
tercera tambien
(anterior ?f1 ?c1 ?d ?f3 ?c3) ;;Uno anterior en la misma direccion que
conecta2
=>
(assert (conecta3 ?t ?d ?f3 ?c3 ?f2 ?c2 ?jugador3))
)
```

En el último ejercicio se añade el hecho ganaría en todos los defrules, pero cada uno sirve para algo diferente. El hecho tiene la forma **ganaria t jugador columna.** Si la siguiente posición a una secuencia de tres (es decir, a la última posición de esa secuencia), tomando la misma orientación, es ocupable, pues se

puede ganar. Igualmente se hace con la posición anterior a la posición de inicio de la secuencia. También se tiene en cuenta cuando hay una secuencia de 2, un hueco, y un elemento del mismo tipo que los que conforman esa secuencia. Ese hueco, tiene que ser una posición ocupable, es decir, debe existir un caería con tales coordenadas. Las comprobaciones se hacen siempre tomando la misma orientación. Igualmente se hace para cuando hay una secuencia de 2 y tenemos que tener en cuenta los elementos anteriores en vez de los siguientes.

```
;;;EJERCICIO 5
;;;Con la ayuda de los hechos anteriores, vamos a deducir cuando uno de
los jugadores puede ganar,
;;;es decir, que haya 3 en linea y un blanco siguiendo a estos 3
(defrule conecta3_blanco
(conecta3 ?t ?d ?f1 ?c1 ?f3 ?c3 ?jugador)
(siguiente ?f3 ?c3 ?d ?f4 ?c4)
(caeria ?f4 ?c4)
=>
(assert (ganaria ?t ?jugador ?c4)) ;;;Guardamos la columna para ganar
)

;;;Procedemos de igual forma pero para casillas anteriores
(defrule blanco_conecta3
(conecta3 ?t ?d ?f1 ?c1 ?f3 ?c3 ?jugador)
(anterior ?f1 ?c1 ?d ?f4 ?c4)
(caeria ?f4 ?c4) ;;;Podemos usar esa casilla?
=>
(assert (ganaria ?t ?jugador ?c4)) ;;;Guardamos la columna para ganar
)

(defrule conecta2_blanco_solo
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
(Tablero ?t ?f4 ?c4 ?jugador3)
(test (neq ?jugador3 _))
(test (eq ?jugador ?jugador3))
(siguiente ?f2 ?c2 ?d ?f3 ?c3)
(siguiente ?f3 ?c3 ?d ?f4 ?c4) ;;;Si esta en el siguiente del siguiente
al conecta2 en la misma direccion
(caeria ?f3 ?c3) ;;Tenemos que poder tirar la ficha en el hueco para
ganar
=>
(assert (ganaria ?t ?jugador ?c3))
)

(defrule solo_blanco_conecta2
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
```

```
(Tablero ?t ?f4 ?c4 ?jugador3)
(test (neq ?jugador3 _))
(test (eq ?jugador ?jugador3))
(anterior ?f1 ?c1 ?d ?f3 ?c3)
(anterior ?f3 ?c3 ?d ?f4 ?c4) ;;;Idem a antes, si esta en el anterior
del anterior
(caeria ?f3 ?c3)
=>
(assert (ganaria ?t ?jugador ?c3))
)
```

A continuación, se detallará la funcionalidad de la máquina para que juegue con el sentido con el que yo jugaría al cuatro en raya. Antes de nada, decir que todas las reglas tienen prioridad negativa (para que se hagan después del cálculo de todos los hechos necesarios, que tienen prioridad 0 por defecto), pero esta prioridad negativa será mayor a la que tienen las reglas por las que la máquina juega aleatoriamente. Además, en todas se busca el hecho **Turno M**, puesto que es para que juegue la máquina.
Lo primero (prioridad negativa más alta), será que si la máquina puede ganar, ganará. Si tenemos el hecho ganaría para alguna posición, este ya implica que si tiramos una ficha en la columna indicada caería en la posición, por la comprobación que hemos hecho en la regla que añade los ganaria. Así que, hacemos que la máquina juegue a esa columna. Además, por pantalla se indica que la máquina ha jugado con criterio a esa columna para ganar, para facilitar que alguien con nulos conocimientos de este juego, pueda ver una estrategia que le lleve al éxito.

```
(defrule ganare
(declare (salience -9000))
(Turno M)
(ganaria ?t M ?c)
=>
(printout t "JUEGO en la columna (con criterio) " ?c " para ganar."
crlf)
(assert (Juega M ?c))
)
```

Lo segundo que se ve es evitar que la máquina pierda, es decir, si hay un hecho ganaria que lleve la J (de jugador) obligatoriamente, jugamos donde jugaría el jugador para ganar, con el fin de cortar la jugada. Igualmente se indica por pantalla lo que se ha hecho.

```
;;;Ahora se evitara que la maquina pierda, es decir, que nos corte las
jugadas en las que
;;;el jugador tenga ya 3 en linea o pueda ganar (los ganaria) meter aqui
```

```
el retract del ganaria?
(defrule no_quiero_perder
(declare (salience -9001))
(Turno M)
(ganaria ?t J ?c)
=>
(printout t "JUEGO en la columna (con criterio) " ?c " porque si no
pierdo." crlf)
(assert (Juega M ?c))
)
```

Las dos siguientes reglas se implementan con la misma prioridad, puesto que da igual cual de las dos ejecute si llega al punto de que puede ejecutar las dos. Lo que hace la máquina es cortar la series de dos que tenga el jugador (teniendo en cuenta la orientación) y también continúa sus secuencias de dos. Da igual que no corte secuencias de 2 del jugador porque estamos asegurando que si el jugador puede ganar, la máquina lo cortará.

```
;;;Ahora haremos que la maquina nos ponga las cosas dificiles e intente
cortar cuaquier
;;;secuencia de 2 del jugador o si hay una secuencia de 2 suya, la continua
(defrule no_te_crezcas
(declare (salience -9002))
(Turno M)
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
(siguiente ?f2 ?c2 ?d ?f3 ?c3)
(caeria ?f3 ?c3)
=>
(printout t "JUEGO en la columna " ?c3 " porque " ?jugador " tiene dos en
linea empezando en la posicion  " ?f1 " " ?c1 crlf)
(assert (Juega M ?c3))
)

;;;Lo mismo pero con la casilla anterior
(defrule no_te_crezcas2
(declare (salience -9002))
(Turno M)
(conecta2 ?t ?d ?f1 ?c1 ?f2 ?c2 ?jugador)
(anterior ?f1 ?c1 ?d ?f3 ?c3)
(caeria ?f3 ?c3)
=>
(printout t "JUEGO en la columna " ?c3 " porque " ?jugador " tiene dos en
linea empezando en la posicion  " ?f1 " " ?c1 crlf)
(assert (Juega M ?c3))
)
```

La siguiente regla se encarga de seguir las secuencias de un elemento M por donde pueda.

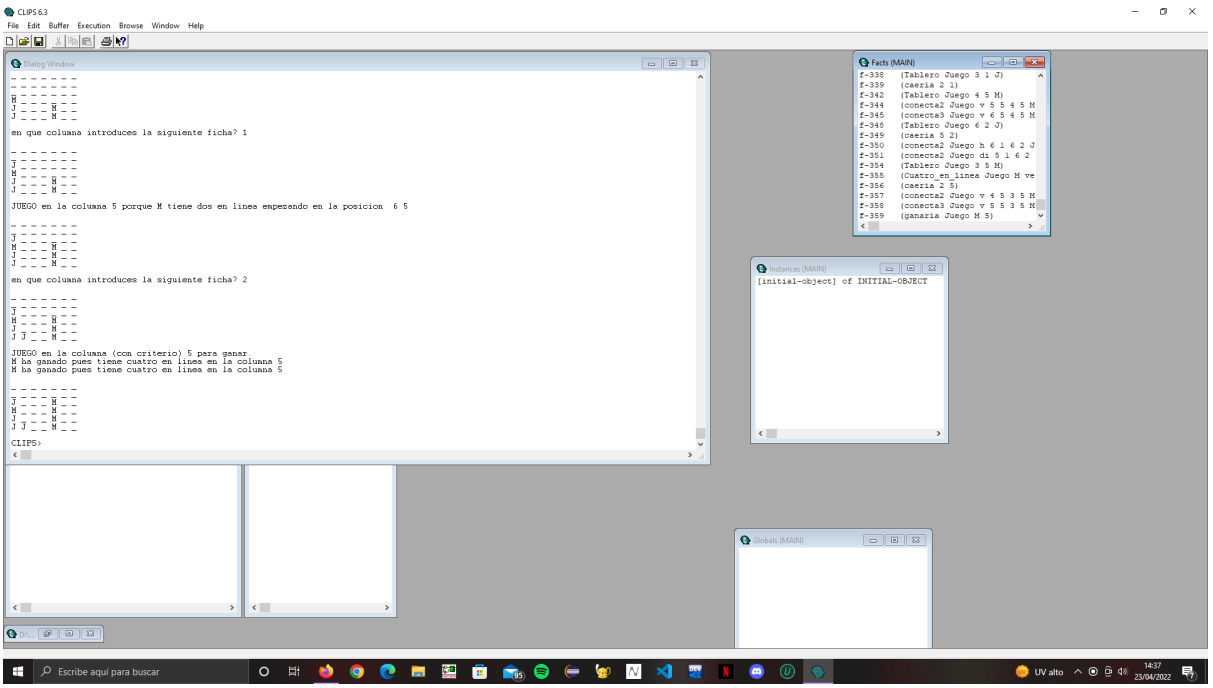
```
;;;Ahora si tengo una M, la continuo
(defrule en_linea2
(declare (salience -9003))
(Turno M)
(Tablero ?t ?f ?c M)
(siguiente ?f ?c ?d ?f2 ?c2)
(caeria ?f2 ?c2)
=>
(printout t "JUEGO en la columna " ?c2 " para intentar continuar." crlf)
(assert (Juega M ?c2))
)

(defrule en_linea2_ant
(declare (salience -9003))
(Turno M)
(Tablero ?t ?f ?c M)
(anterior ?f ?c ?d ?f2 ?c2)
(caeria ?f2 ?c2)
=>
(printout t "JUEGO en la columna " ?c2 " para intentar continuar." crlf)
(assert (Juega M ?c2))
)
```

Por último, tenemos la regla ya_no_gano, que se ejecuta con prioridad positiva y se encarga de eliminar los ganaria que ya no nos sirvan para ganar con tal de que no se tengan en cuenta al calcular la jugada.

```
;;;Elimino los ganaria que ya no me sirven para ganar
(defrule ya_no_gano
(declare (salience 1))
?regla <- (ganaria ?t ?jugador ?c)
(Juega ?jugador2 ?c)
=>
(retract ?regla)
)
```

A continuación, se muestra una captura de una ejecución:



Práctica 3:

El problema consiste en diseñar un sistema experto que asesore a un estudiante de ingeniería informática sobre qué rama elegir de forma que el sistema actúe tal y como lo haríais vosotros.

Así, la práctica consiste en crear un programa en CLIPS que:

1. **Le pregunte al usuario que pide asesoramiento lo que le preguntaríais a alguien que os pida consejo en ese sentido, y de la forma y orden en que lo preguntaríais vosotros.**
2. **Razone y tome decisiones cómo lo haríais vosotros para esta tarea.**

Le aconseje la rama o las ramas que le aconsejaríais vosotros, junto con los motivos por los que se le aconseja.

Mi sistema hace una serie de preguntas a partir de la cual guarda diversos hechos y da una recomendación, en base a la que razona las posibles recomendaciones que podría hacer. He decidido que el sistema siempre de una sola recomendación, que se puede ver en la lista de hechos puesto que es el último que se añade y es el único de la forma Consejo Rama "Texto". Todo lo referente a lo anteriormente explicado está comentado en el código de la práctica.

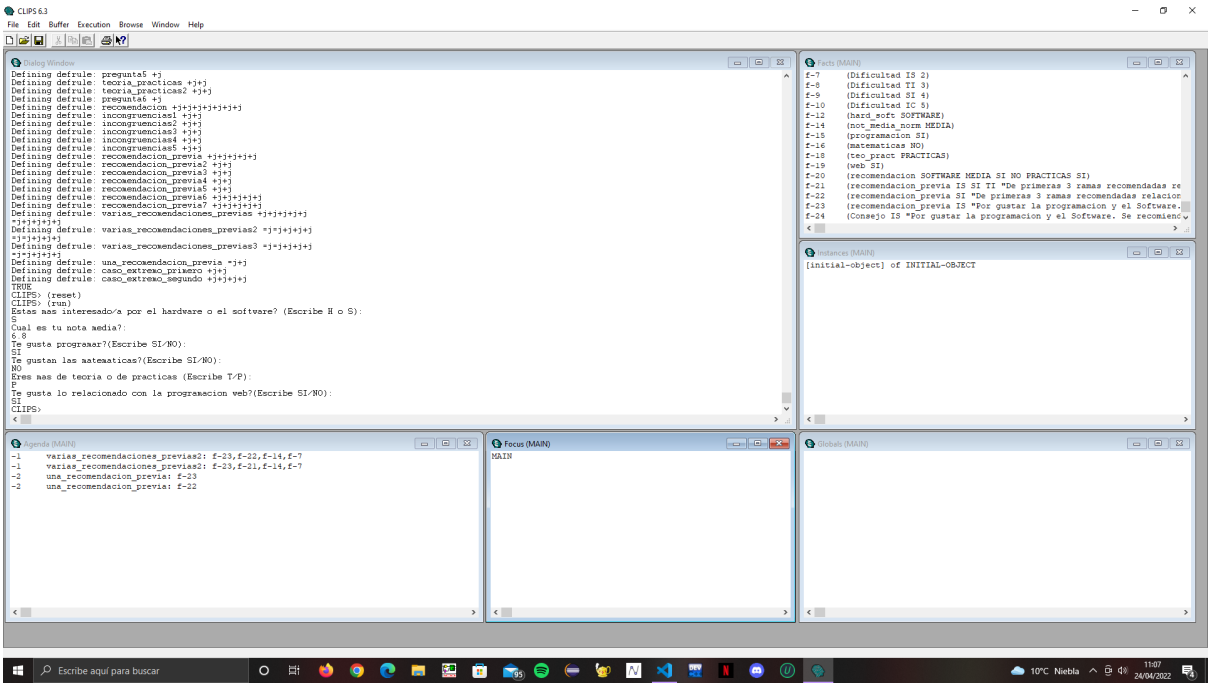
Cabe mencionar que además de los *deffacts* que se nos pedía usar, he añadido los siguientes hechos:

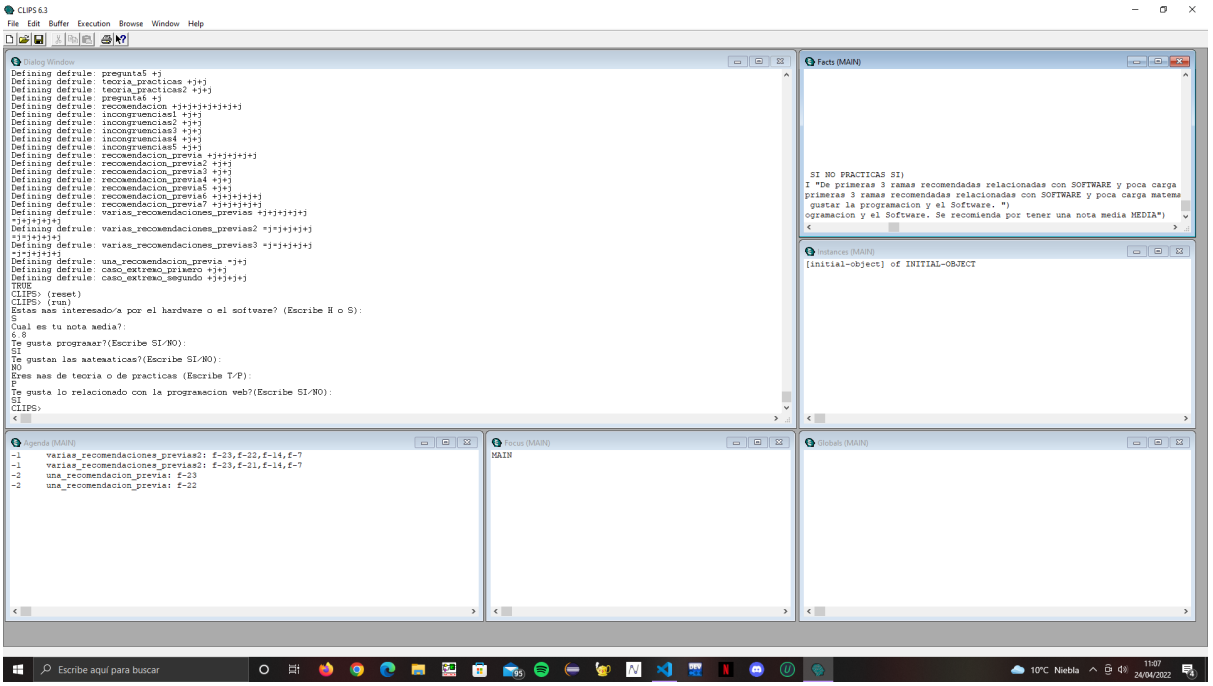
```
;;;Si la nota es media o alta cogeremos la de mayor dificultad
disponible y si es baja la de menor
```

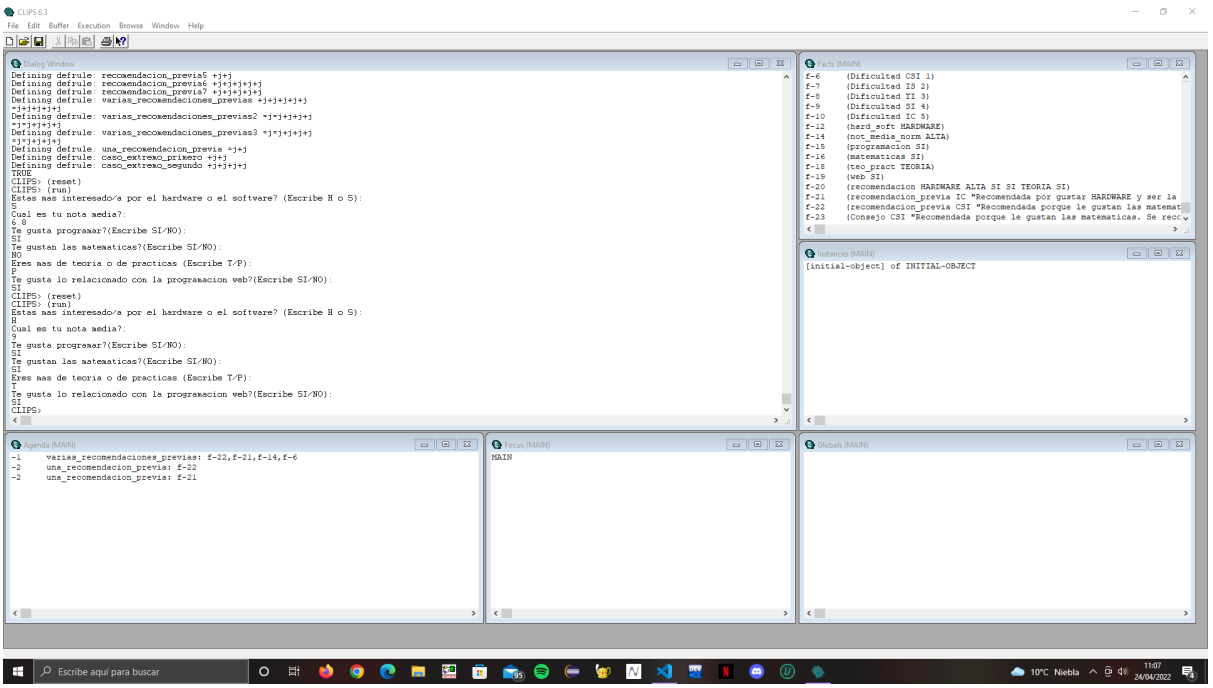
```
(deffacts Dificultades
(Dificultad CSI 1)
(Dificultad IS 2)
(Dificultad TI 3)
(Dificultad SI 4)
(Dificultad IC 5)
)
```

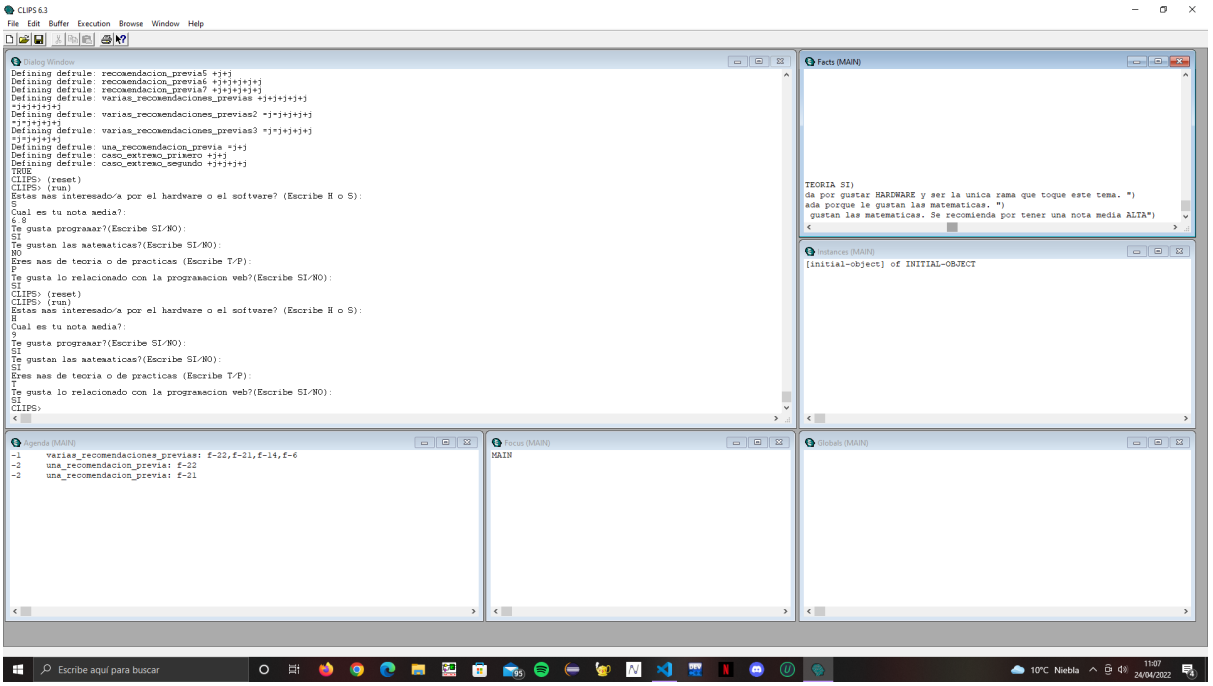
Las dificultades están pensadas en base a mi propia experiencia y siendo 1 la más difícil. La nota media se usa como criterio junto con estas dificultades cuando tenemos varias recomendaciones por el sistema, puesto que interesa que los mejores expedientes vayan hacia las más difíciles.
Además, el sistema tiene bastante escalabilidad, puesto que añadiendo más filtros a la hora de hacer recomendaciones, haciendo recomendaciones previas de mayor tamaño y considerando un diferente criterio de selección para estas recomendaciones (las que tienen varias menciones), se podría abarcar muchas más respuestas y de mayor calidad.

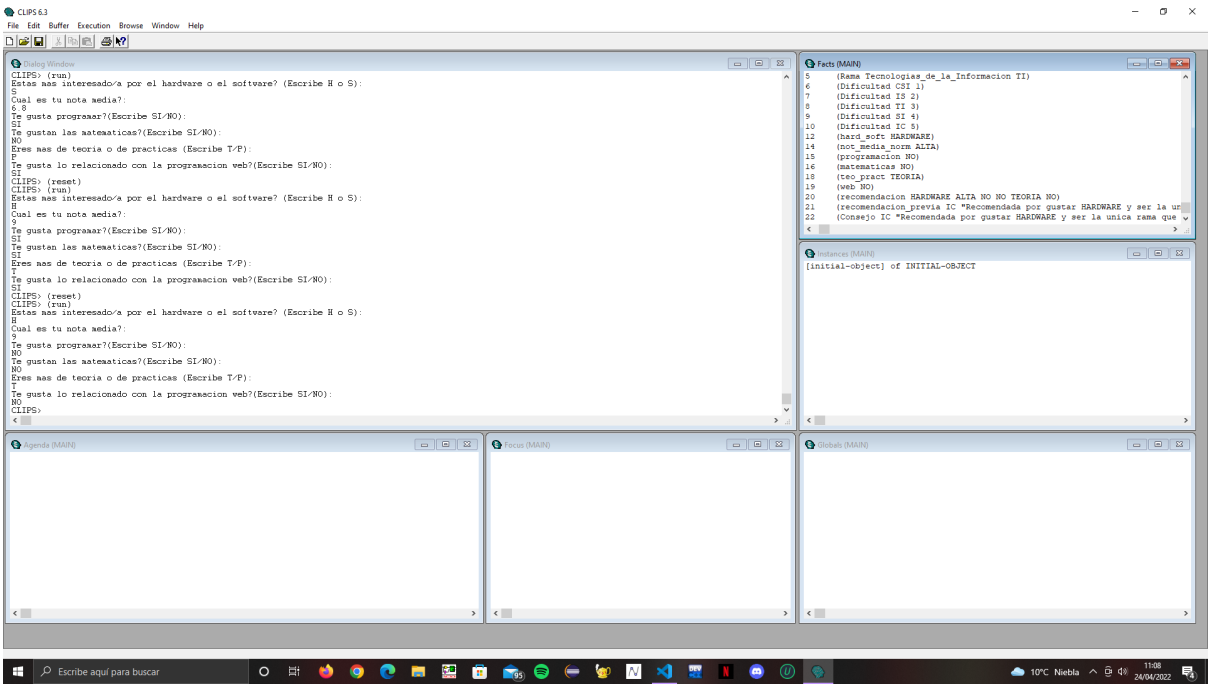
A continuación, se muestran las capturas de 3 ejecuciones. La segunda captura de cada para es para mostrar la correcta concatenación de strings.

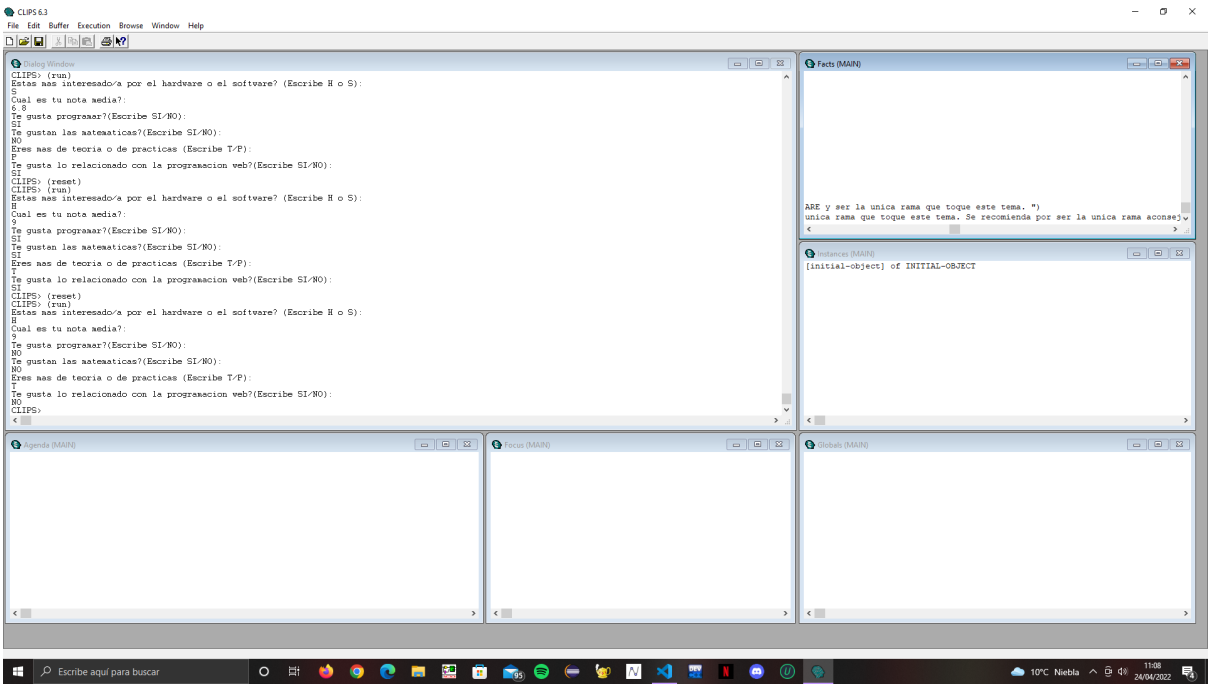



CLIPS 6.3
File Edit Buffer Execution Browse Window Help
Dialog Window
Defining defrule: pregunta5 +j
Defining defrule: teoria_practicas +j+j
Defining defrule: teoria_practicas2 +j+j
Defining defrule: pregunta6 +j
Defining defrule: recomendacion +j+j+j+j+j+j+j
Defining defrule: incongruencias1 +j+j
Defining defrule: incongruencias2 +j+j
Defining defrule: incongruencias3 +j+j
Defining defrule: incongruencias4 +j+j
Defining defrule: incongruencias5 +j+j
Defining defrule: recomendacion_previa +j+j+j+j+j
Defining defrule: recomendacion_previa2 +j+j
Defining defrule: recomendacion_previa3 +j+j
Defining defrule: recomendacion_previa4 +j+j
Defining defrule: recomendacion_previa5 +j+j
Defining defrule: recomendacion_previa6 +j+j+j+j+j
Defining defrule: recomendacion_previa7 +j+j+j+j+j
Defining defrule: varias_recomendaciones_previas +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas2 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas3 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: una_recomendacion_previa +j+j
Defining defrule: caso_extremo_primero +j+j
Defining defrule: caso_extremo_segundo +j+j+j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
S
Cual es tu nota media?:
6.8
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
NO
Eres mas de teoria o de practicas (Escribe T/P):
P
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS>
Facts (MAIN)
SI NO PRACTICAS SI)
I "De primeras 3 ramas recomendadas relacionadas con SOFTWARE y poca carga
primeras 3 ramas recomendadas relacionadas con SOFTWARE y poca carga matema
gustar la programacion y el Software. ")
ogramacion y el Software. Se recomienda por tener una nota media MEDIA")
Instances (MAIN)
[initial-object] of INITIAL-OBJECT
Agenda (MAIN)
-1 varias_recomendaciones_previas2: f-23,f-22,f-14,f-7
-1 varias_recomendaciones_previas2: f-23,f-21,f-14,f-7
-2 una_recomendacion_previa: f-23
-2 una_recomendacion_previa: f-22
Focus (MAIN)
MAIN
Globals (MAIN)
Escribe aquí para buscar
10°C Niebla
11:07
24/04/2022


CLIPS 6.3
File Edit Buffer Execution Browse Window Help
Dialog Window
Defining defrule: recomendacion_previa5 +j+j
Defining defrule: recomendacion_previa6 +j+j+j+j+j
Defining defrule: recomendacion_previa7 +j+j+j+j+j
Defining defrule: varias_recomendaciones_previas +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas2 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas3 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: una_recomendacion_previa +j+j
Defining defrule: caso_extremo_primero +j+j
Defining defrule: caso_extremo_segundo +j+j+j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
S
Cual es tu nota media?:
6.8
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
NO
Eres mas de teoria o de practicas (Escribe T/P):
P
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
H
Cual es tu nota media?:
9
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
SI
Eres mas de teoria o de practicas (Escribe T/P):
T
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS>
Facts (MAIN)
f-6 (Dificultad CSI 1)
f-7 (Dificultad IS 2)
f-8 (Dificultad TI 3)
f-9 (Dificultad SI 4)
f-10 (Dificultad IC 5)
f-12 (hard_soft HARDWARE)
f-14 (not_media_norm ALTA)
f-15 (programacion SI)
f-16 (matematicas SI)
f-18 (teo_pract TEORIA)
f-19 (web SI)
f-20 (recomendacion HARDWARE ALTA SI SI TEORIA SI)
f-21 (recomendacion_previa IC "Recomendada por gustar HARDWARE y ser la
f-22 (recomendacion_previa CSI "Recomendada porque le gustan las matemat
f-23 (Consejo CSI "Recomendada porque le gustan las matematicas. Se recc
Instances (MAIN)
[initial-object] of INITIAL-OBJECT
Agenda (MAIN)
-1 varias_recomendaciones_previas: f-22,f-21,f-14,f-6
-2 una_recomendacion_previa: f-22
-2 una_recomendacion_previa: f-21
Focus (MAIN)
MAIN
Globals (MAIN)
Escribe aquí para buscar
10°C Niebla
11:07
24/04/2022


CLIPS 6.3
File Edit Buffer Execution Browse Window Help
Dialog Window
Defining defrule: recomendacion_previa5 +j+j
Defining defrule: recomendacion_previa6 +j+j+j+j+j
Defining defrule: recomendacion_previa7 +j+j+j+j+j
Defining defrule: varias_recomendaciones_previas +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas2 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: varias_recomendaciones_previas3 +j+j+j+j+j
+j+j+j+j+j
Defining defrule: una_recomendacion_previa +j+j
Defining defrule: caso_extremo_primero +j+j
Defining defrule: caso_extremo_segundo +j+j+j+j
TRUE
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
S
Cual es tu nota media?:
6.8
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
NO
Eres mas de teoria o de practicas (Escribe T/P):
P
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
H
Cual es tu nota media?:
9
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
SI
Eres mas de teoria o de practicas (Escribe T/P):
T
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS>
Facts (MAIN)
TEORIA SI)
da por gustar HARDWARE y ser la unica rama que toque este tema. ")
ada porque le gustan las matematicas. ")
gustan las matematicas. Se recomienda por tener una nota media ALTA")
Instances (MAIN)
[initial-object] of INITIAL-OBJECT
Agenda (MAIN)
-1 varias_recomendaciones_previas: f-22,f-21,f-14,f-6
-2 una_recomendacion_previa: f-22
-2 una_recomendacion_previa: f-21
Focus (MAIN)
MAIN
Globals (MAIN)
Escribe aquí para buscar
10°C Niebla
11:07
24/04/2022


CLIPS 6.3
File Edit Buffer Execution Browse Window Help
Dialog Window
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
S
Cual es tu nota media?:
6.8
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
NO
Eres mas de teoria o de practicas (Escribe T/P):
P
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
H
Cual es tu nota media?:
9
Te gusta programaar?(Escribe SI/NO):
SI
Te gustan las mateaaticas?(Escribe SI/NO):
SI
Eres mas de teoria o de practicas (Escribe T/P):
T
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
SI
CLIPS> (reset)
CLIPS> (run)
Estas mas interesado/a por el hardware o el software? (Escribe H o S):
H
Cual es tu nota media?:
9
Te gusta programaar?(Escribe SI/NO):
NO
Te gustan las mateaaticas?(Escribe SI/NO):
NO
Eres mas de teoria o de practicas (Escribe T/P):
T
Te gusta lo relacionado con la programacion web?(Escribe SI/NO):
NO
CLIPS>
Facts (MAIN)
5 (Rama Tecnologias_de_la_Informacion TI)
6 (Dificultad CSI 1)
7 (Dificultad IS 2)
8 (Dificultad TI 3)
9 (Dificultad SI 4)
10 (Dificultad IC 5)
12 (hard_soft HARDWARE)
14 (not_media_norm ALTA)
15 (programacion NO)
16 (matematicas NO)
18 (teo_pract TEORIA)
19 (web NO)
20 (recomendacion HARDWARE ALTA NO NO TEORIA NO)
21 (recomendacion_previa IC "Recomendada por gustar HARDWARE y ser la un
22 (Consejo IC "Recomendada por gustar HARDWARE y ser la unica rama que
Instances (MAIN)
[initial-object] of INITIAL-OBJECT
Agenda (MAIN)
Focus (MAIN)
Globals (MAIN)
Escribe aquí para buscar
10°C Niebla
11:08
24/04/2022



Por último, destacar que el sistema admite no sé cómo respuesta, poniendo cualquier cosa sin sentido (únicamente no lo admite en la nota media), aunque preferiblemente, y por una ejecución más clara, se recomienda poner ne (de next), para evadir la pregunta.

Bibliografía

(n.d.). FAQ de CLIPS - Intel.ligència Artificial. https://www.cs.upc.edu/~bejar/ia/material/laboratorio/clips/FAQ-CLIPS.pdf

(n.d.). 12.3.1 String Concatenation. https://www.csie.ntu.edu.tw/~sylee/courses/clips/bpg/node12.3.1.html

Berzal, F. (n.d.). *Sistemas Inteligentes de Gestión Tutorial de CLIPS*. https://elvex.ugr.es/decsai/intelligent/workbook/ai/CLIPS.pdf

Tema 1: Programación basada en reglas con CLIPS. (n.d.). Dpto. Ciencias de la Computación e Inteligencia Artificial. https://www.cs.us.es/~jalonso/cursos/ia2-02/temas/tema-1.pdf