;;;======================================================

;;; Book Recomendation Expert System

;;;

;;; This expert system recommends books

;;;

;;; CLIPS Version 6.0 Example

;;;

;;; To execute, merely load, reset and run.

;;;======================================================

;;Ontologia --------------------------------------------------------------------

; Sun Dec 01 19:02:14 CET 2019

;

;+ (version "3.5")

;+ (build "Build 663")

(defclass %3ACLIPS\_TOP\_LEVEL\_SLOT\_CLASS "Fake class to save top-level slot information"

(is-a USER)

(role abstract)

(multislot LlibresNomPublic

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot inverse\_of\_LlibresNomPublic)

(create-accessor read-write))

(multislot CategoriaLlibre

(type INSTANCE)

;+ (allowed-classes Categoria)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot LlibresCategoria)

(create-accessor read-write))

(single-slot inverse\_of\_LlibrePreu

(type INSTANCE)

;+ (allowed-classes Preu)

;+ (cardinality 1 1)

;+ (inverse-slot LlibrePreu)

(create-accessor read-write))

(single-slot inverse\_of\_LlibresNacionalitatAutor

(type INSTANCE)

;+ (allowed-classes Nacionalitat\_Autor)

;+ (cardinality 1 1)

;+ (inverse-slot LlibresNacionalitatAutor)

(create-accessor read-write))

(multislot inverse\_of\_LlibresNumPag

(type INSTANCE)

;+ (allowed-classes Llargada)

(create-accessor read-write))

(multislot inverse\_of\_LlibresCategoria

(type INSTANCE)

;+ (allowed-classes Categoria)

(create-accessor read-write))

(single-slot categoria\_Edat

(type INSTANCE)

;+ (allowed-classes)

;+ (cardinality 1 1)

;+ (inverse-slot EdatMin)

(create-accessor read-write))

(multislot Relevancia

(type INSTANCE)

;+ (allowed-classes)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot Llibres\_Rellevancia)

(create-accessor read-write))

(multislot LlibresNumPag

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot NumPagines)

(create-accessor read-write))

(single-slot NumPags

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot LlibrePreu

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot inverse\_of\_LlibrePreu)

(create-accessor read-write))

(single-slot Rellevancia

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot inverse\_of\_LlibresNomPublic

(type INSTANCE)

;+ (allowed-classes PublicDestinatari)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot LlibresNomPublic)

(create-accessor read-write))

(single-slot NomCategoria

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot LlibresCategoria

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot CategoriaLlibre)

(create-accessor read-write))

(single-slot NumPagines

(type INSTANCE)

;+ (allowed-classes Llargada)

;+ (cardinality 1 1)

;+ (inverse-slot LlibresNumPag)

(create-accessor read-write))

(multislot LlibresNacionalitatAutor

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot inverse\_of\_LlibresNacionalitatAutor)

(create-accessor read-write))

(single-slot NomLlibre

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot EdatMin

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot categoria\_Edat)

(create-accessor read-write))

(single-slot LlibresEdatMin

(type STRING)

;+ (cardinality 0 1)

(create-accessor read-write))

(single-slot nacionalitatAutor

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot CategoriaEdat

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot preu

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot Llibres\_Rellevancia

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot Relevancia)

(create-accessor read-write))

(single-slot Pertany\_Saga

(type SYMBOL)

(allowed-values FALSE TRUE)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot inverse\_of\_LlibreSaga

(type INSTANCE)

;+ (allowed-classes Saga)

;+ (cardinality 1 1)

;+ (inverse-slot LlibreSaga)

(create-accessor read-write))

(multislot LlibresCategoriaEdat

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot LlibresCategoriaEdat)

(create-accessor read-write))

(single-slot Provisional1\_Class21

(type STRING)

;+ (cardinality 0 1)

(create-accessor read-write))

(multislot LlibreSaga

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

;+ (inverse-slot inverse\_of\_LlibreSaga)

(create-accessor read-write))

(single-slot NomPublic

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass Llibres

(is-a USER)

(role concrete)

(single-slot inverse\_of\_LlibreSaga

(type INSTANCE)

;+ (allowed-classes Saga)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot NumPagines

(type INSTANCE)

;+ (allowed-classes Llargada)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot NomLlibre

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot inverse\_of\_LlibresNomPublic

(type INSTANCE)

;+ (allowed-classes PublicDestinatari)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(multislot CategoriaLlibre

(type INSTANCE)

;+ (allowed-classes Categoria)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(single-slot inverse\_of\_LlibrePreu

(type INSTANCE)

;+ (allowed-classes Preu)

;+ (cardinality 1 1)

(create-accessor read-write))

(single-slot inverse\_of\_LlibresNacionalitatAutor

(type INSTANCE)

;+ (allowed-classes Nacionalitat\_Autor)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass Categoria

(is-a USER)

(role concrete)

(single-slot NomCategoria

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot LlibresCategoria

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write)))

(defclass Llargada

(is-a USER)

(role concrete)

(multislot LlibresNumPag

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(single-slot NumPags

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass PublicDestinatari

(is-a USER)

(role concrete)

(multislot LlibresNomPublic

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(single-slot NomPublic

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass Preu

(is-a USER)

(role concrete)

(multislot LlibrePreu

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(single-slot preu

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass Saga

(is-a USER)

(role concrete)

(multislot LlibreSaga

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write))

(single-slot Pertany\_Saga

(type SYMBOL)

(allowed-values FALSE TRUE)

;+ (cardinality 1 1)

(create-accessor read-write)))

(defclass Nacionalitat\_Autor

(is-a USER)

(role concrete)

(single-slot nacionalitatAutor

(type STRING)

;+ (cardinality 1 1)

(create-accessor read-write))

(multislot LlibresNacionalitatAutor

(type INSTANCE)

;+ (allowed-classes Llibres)

(cardinality 1 ?VARIABLE)

(create-accessor read-write)))

;;Instancias --------------------------------------------------------------------

(definstances instancias

([KB\_109745\_Class17] of Categoria

(LlibresCategoria [KB\_109745\_Class34])

(NomCategoria "Drama"))

([KB\_109745\_Class19] of Llargada

(LlibresNumPag [Provisional1\_Class0])

(NumPags "Corto(<500)"))

([KB\_109745\_Class34] of Llibres

(CategoriaLlibre [KB\_109745\_Class17])

(inverse\_of\_LlibrePreu [Provisional1\_Class34])

(inverse\_of\_LlibreSaga [Provisional1\_Class29])

(inverse\_of\_LlibresNacionalitatAutor [Provisional1\_Class37])

(inverse\_of\_LlibresNomPublic

[Provisional1\_Class9]

[Provisional1\_Class8])

(NomLlibre "50 Sombras de Grey")

(NumPagines [KB\_109745\_Class35]))

([KB\_109745\_Class35] of Llargada

(LlibresNumPag

[KB\_109745\_Class34]

[Provisional1\_Class2])

(NumPags "Largo(>500)"))

([Provisional1\_Class0] of Llibres

(CategoriaLlibre [Provisional1\_Class1])

(inverse\_of\_LlibrePreu [Provisional1\_Class32])

(inverse\_of\_LlibreSaga [Provisional1\_Class28])

(inverse\_of\_LlibresNacionalitatAutor [Provisional1\_Class35])

(inverse\_of\_LlibresNomPublic [Provisional1\_Class8])

(NomLlibre "1984")

(NumPagines [KB\_109745\_Class19]))

([Provisional1\_Class1] of Categoria

(LlibresCategoria [Provisional1\_Class0])

(NomCategoria "Ciencia Ficcio"))

([Provisional1\_Class2] of Llibres

(CategoriaLlibre [Provisional1\_Class4])

(inverse\_of\_LlibrePreu [Provisional1\_Class32])

(inverse\_of\_LlibreSaga [Provisional1\_Class28])

(inverse\_of\_LlibresNacionalitatAutor [Provisional1\_Class37])

(inverse\_of\_LlibresNomPublic [Provisional1\_Class8])

(NomLlibre "El resplandor")

(NumPagines [KB\_109745\_Class35]))

([Provisional1\_Class28] of Saga

(LlibreSaga

[Provisional1\_Class0]

[Provisional1\_Class2])

(Pertany\_Saga FALSE))

([Provisional1\_Class29] of Saga

(LlibreSaga [KB\_109745\_Class34])

(Pertany\_Saga TRUE))

([Provisional1\_Class32] of Preu

(LlibrePreu

[Provisional1\_Class0]

[Provisional1\_Class2])

(preu "<11"))

([Provisional1\_Class34] of Preu

(LlibrePreu [KB\_109745\_Class34])

(preu "<15"))

([Provisional1\_Class35] of Nacionalitat\_Autor

(LlibresNacionalitatAutor [Provisional1\_Class0])

(nacionalitatAutor "UK"))

([Provisional1\_Class37] of Nacionalitat\_Autor

(LlibresNacionalitatAutor

[Provisional1\_Class2]

[KB\_109745\_Class34])

(nacionalitatAutor "USA"))

([Provisional1\_Class4] of Categoria

(LlibresCategoria [Provisional1\_Class2])

(NomCategoria "Terror"))

([Provisional1\_Class8] of PublicDestinatari

(LlibresNomPublic

[Provisional1\_Class0]

[KB\_109745\_Class34]

[Provisional1\_Class2])

(NomPublic "Adult"))

([Provisional1\_Class9] of PublicDestinatari

(LlibresNomPublic [KB\_109745\_Class34])

(NomPublic "Adolescent"))

)

;;Clases nuestras

(defclass Recomendacion

(is-a USER)

(role concrete)

(slot nombre\_cuadro

(type INSTANCE)

(create-accessor read-write))

(slot puntuacion

(type INTEGER)

(create-accessor read-write))

(multislot justificaciones

(type STRING)

(create-accessor read-write))

)

(defclass Dia

(is-a USER)

(role concrete)

(multislot recomendaciones

(type INSTANCE)

(create-accessor read-write))

(slot tiempo-maximo

(type INTEGER)

(create-accessor read-write))

)

;;; Modulo principal de utilidades

(defmodule MAIN (export ?ALL))

;;; Modulo de recopilacion de los datos del grupo + preferencias

(defmodule recopilacion-grupo

(import MAIN ?ALL)

(export ?ALL)

)

(defmodule recopilacion-preferencias

(import MAIN ?ALL)

(import recopilacion-grupo deftemplate ?ALL)

(export ?ALL)

)

(defmodule procesado-datos

(import MAIN ?ALL)

(import recopilacion-grupo deftemplate ?ALL)

(import recopilacion-preferencias deftemplate ?ALL)

(export ?ALL)

)

(defmodule generacion\_soluciones

(import MAIN ?ALL)

(export ?ALL)

)

(defmodule resultados\_al\_grupo

(import MAIN ?ALL)

(export ?ALL)

)

;;; Imprime los datos de un contenido

(defmessage-handler MAIN::Cuadro imprimir ()

(format t "Titulo: %s %n" ?self:Titulo)

(printout t crlf)

(format t "Anyo: %d" ?self:Anyo)

(printout t crlf)

(format t "Epoca del cuadro: %s" (send ?self:Epoca\_cuadro get-Nombre\_epoca))

(printout t crlf)

(format t "Dimensiones: %s" ?self:Dim)

(printout t crlf)

(format t "Sala: %d" ?self:Sala)

(printout t crlf)

(format t "Pintado por: %s" (send ?self:Pintado\_por get-Nombre))

(printout t crlf)

(format t "Tematica del cuadro: %s" (send ?self:Tematica\_cuadro get-Nombre\_tematica))

(printout t crlf)

)

(defmessage-handler MAIN::Recomendacion imprimir ()

(printout t "-----------------------------------" crlf)

(printout t (send ?self:nombre\_cuadro imprimir))

(printout t crlf)

(format t "Nivel de recomendacion: %d %n" ?self:puntuacion)

(printout t "Justificacion de la eleccion: " crlf)

(progn$ (?curr-just ?self:justificaciones)

(printout t ?curr-just crlf)

)

(printout t crlf)

(printout t "-----------------------------------" crlf)

)

(defmessage-handler MAIN::Dia imprimir ()

(printout t "============================================" crlf)

(bind $?recs ?self:recomendaciones)

(progn$ (?curr-rec $?recs)

(printout t (send ?curr-rec imprimir))

)

(printout t "============================================" crlf)

)

;;; Declaracion de templates --------------------------

;;; Template para los datos del grupo

(deftemplate MAIN::datos\_grupo

(slot descripcion (type STRING) (default "desc")) ;tamanyo del grupo

(slot nivel (type INTEGER)(default -1)) ;conocimiento

(slot edad (type INTEGER)(default -1)) ;edad general del grupo

(slot dias (type INTEGER)(default -1)) ;nº dias en visitar el museo

(slot horasdia (type INTEGER)(default -1)) ;nº horas/dia

(slot tiempo (type INTEGER)(default -1)) ;total de tiempo

)

;;; Template para las preferencias del usuario

(deftemplate MAIN::preferencias\_grupo

(multislot autores\_favoritos (type INSTANCE))

(multislot tematicas\_obras\_fav (type INSTANCE))

(multislot estilos\_favoritos (type INSTANCE))

(multislot epocas\_favoritas (type INSTANCE))

)

;;; Template para una lista de recomendaciones sin orden

(deftemplate MAIN::lista-rec-desordenada

(multislot recomendaciones (type INSTANCE))

)

;;; Template para una lista de recomendaciones con orden

(deftemplate MAIN::lista-rec-ordenada

(multislot recomendaciones (type INSTANCE))

)

(deftemplate MAIN::lista-dias

(multislot dias (type INSTANCE))

)

(deftemplate MAIN::dias-orden-sala

(multislot dias (type INSTANCE))

)

;;; Cuadros ordenados

(defclass ListaCuadros

(is-a USER)

(role concrete)

(multislot cuadros

(type INSTANCE)

(create-accessor read-write))

)

(deftemplate MAIN::lista-cuadros-ordenada

(multislot cuadros (type INSTANCE))

)

(deffunction maximo-puntuacion ($?lista)

(bind ?maximo -1)

(bind ?elemento nil)

(progn$ (?curr-rec $?lista)

(bind ?curr-cont (send ?curr-rec get-nombre\_cuadro))

(bind ?curr-punt (send ?curr-rec get-puntuacion))

(if (> ?curr-punt ?maximo)

then

(bind ?maximo ?curr-punt)

(bind ?elemento ?curr-rec)

)

)

?elemento

)

(deffunction orden-sala ($?lista)

(bind ?minimo 6)

(bind ?elemento nil)

(progn$ (?curr-rec $?lista)

(bind ?curr-cuadro (send ?curr-rec get-nombre\_cuadro))

(bind ?curr-sala (send ?curr-cuadro get-Sala))

(if (<= ?curr-sala ?minimo)

then

(bind ?minimo ?curr-sala)

(bind ?elemento ?curr-rec)

)

)

?elemento

)

;;; Funcion para hacer una pregunta no-numerica-univalor

(deffunction pregunta-datos (?pregunta)

(format t "%s " ?pregunta)

(bind ?respuesta (read))

(while (not (lexemep ?respuesta)) do

(format t "%s " ?pregunta)

(bind ?respuesta (read))

)

?respuesta

)

;;; Funcion para hacer una pregunta numerica-univalor

(deffunction MAIN::pregunta-numerica (?pregunta ?rangini ?rangfi)

(format t "%s (De %d hasta %d) " ?pregunta ?rangini ?rangfi)

(bind ?respuesta (read))

(while (not(and(>= ?respuesta ?rangini)(<= ?respuesta ?rangfi))) do

(format t "%s (De %d hasta %d) " ?pregunta ?rangini ?rangfi)

(bind ?respuesta (read))

)

?respuesta

)

;;; Funcion para hacer pregunta con muchas opciones

(deffunction MAIN::pregunta-opciones (?pregunta $?valores-posibles)

(bind ?linea (format nil "%s" ?pregunta))

(printout t ?linea crlf)

(progn$ (?var ?valores-posibles)

(bind ?linea (format nil " %d. %s" ?var-index ?var))

(printout t ?linea crlf)

)

(bind ?respuesta (pregunta-numerica "Escoge una opcion:" 1 (length$ ?valores-posibles)))

?respuesta

)

;;; Funcion para hacer una pregunta general con una serie de respuestas admitidas

(deffunction MAIN::pregunta-opciones2 (?question $?allowed-values)

(format t "%s "?question)

(progn$ (?curr-value $?allowed-values)

(format t "[%s]" ?curr-value)

)

(printout t ": ")

(bind ?answer (read))

(if (lexemep ?answer)

then (bind ?answer (lowcase ?answer)))

(while (not (member ?answer ?allowed-values)) do

(format t "%s "?question)

(progn$ (?curr-value $?allowed-values)

(format t "[%s]" ?curr-value)

)

(printout t ": ")

(bind ?answer (read))

(if (lexemep ?answer)

then (bind ?answer (lowcase ?answer))))

?answer

)

;;; Funcion para hacer una pregunta de tipo si/no

(deffunction MAIN::pregunta-si-no (?question)

(bind ?response (pregunta-opciones ?question si no))

(if (or (eq ?response si) (eq ?response s))

then TRUE

else FALSE)

)

;;; Funcion para hacer una pregunta multi-respuesta con indices

(deffunction MAIN::pregunta-multirespuesta (?pregunta $?valores-posibles)

(bind ?linea (format nil "%s" ?pregunta))

(printout t ?linea crlf)

(progn$ (?var ?valores-posibles)

(bind ?linea (format nil " %d. %s" ?var-index ?var))

(printout t ?linea crlf)

)

(format t "%s" "Indica los numeros referentes a las preferencias separados por un espacio: ")

(bind ?resp (readline))

(bind ?numeros (str-explode ?resp))

(bind $?lista (create$))

(progn$ (?var ?numeros)

(if (and (integerp ?var) (and (>= ?var 0) (<= ?var (length$ ?valores-posibles))))

then

(if (not (member$ ?var ?lista))

then (bind ?lista (insert$ ?lista (+ (length$ ?lista) 1) ?var))

)

)

)

(if (or(member$ 0 ?lista)(= (length$ ?lista) 0)) then (bind ?lista (create$ )))

;(if (member$ 0 ?lista) then (bind ?lista (create$ 0)))

?lista

)

;;; Funcion para hacer pregunta con indice de respuestas posibles

(deffunction MAIN::pregunta-indice (?pregunta $?valores-posibles)

(bind ?linea (format nil "%s" ?pregunta))

(printout t ?linea crlf)

(progn$ (?var ?valores-posibles)

(bind ?linea (format nil " %d. %s" ?var-index ?var))

(printout t ?linea crlf)

)

(bind ?respuesta (pregunta-numerica "Escoge una opcion:" 1 (length$ ?valores-posibles)))

?respuesta

)

(defrule MAIN::initialRule "Regla inicial"

(declare (salience 10))

=>

(printout t"----------------------------------------------------------" crlf)

(printout t" Personalizacion de visitas a un museo " crlf)

(printout t"----------------------------------------------------------" crlf)

(printout t crlf)

(printout t"¡Bienvenido! A continuacion se le formularan una serie de preguntas para poder recomendarle una visita adecuada a sus preferencias." crlf)

(printout t crlf)

(focus recopilacion-grupo)

)

;;; Recopilacion de datos de entrada -------------------------------------------------

(defrule recopilacion-grupo::establecer-tamanyo "Establece el tamanyo del grupo"

(not (datos\_grupo))

=>

(bind ?d (pregunta-numerica "¿De cuantos visitantes esta formado el grupo? " 1 100))

(if (= ?d 1) then (bind ?descripcion "Individual"))

(if (= ?d 2) then (bind ?descripcion "Pareja"))

(if (and(> ?d 2) (< ?d 13)) then (bind ?descripcion "Grupo pequeno (3-12)"))

(if (and(> ?d 12) (< ?d 26)) then (bind ?descripcion "Grupo mediano (13-25)"))

(if (> ?d 25) then (bind ?descripcion "Grupo grande (+25)"))

(assert (datos\_grupo (descripcion ?descripcion)))

)

(defrule recopilacion-grupo::establecer-edad "Establece la edad media del grupo"

?g <- (datos\_grupo (edad ?edad))

(test (< ?edad 0))

=>

(bind ?edad (pregunta-numerica "¿Cual es la media de edad del grupo? " 1 110))

(modify ?g (edad ?edad))

)

(defrule recopilacion-grupo::establecer-dias "Establece el nº dias de la visita"

?g <- (datos\_grupo (dias ?dias))

(test (< ?dias 0)

)

=>

(bind ?dias (pregunta-numerica "¿Durante cuantos dias realizara la visita?" 1 7))

(modify ?g (dias ?dias))

)

(defrule recopilacion-grupo::establecer-horas "Establece el nº dias de horas de la visita"

?g <- (datos\_grupo (horasdia ?horasdia))

(test (< ?horasdia 0))

=>

(bind ?horasdia (pregunta-numerica "¿Cuantas horas dedicara diariamente a visitar el museo?" 1 6))

(modify ?g (horasdia ?horasdia))

)

(defrule recopilacion-grupo::establecer-tiempo "Establece el nº dias de horas de la visita"

?g <- (datos\_grupo (dias ?dias))

?d <- (datos\_grupo (horasdia ?horasdia))

?t <- (datos\_grupo (tiempo ?tiempo))

(test (< ?tiempo 0))

=>

(modify ?t (tiempo (bind ?tiempo (\* ?horasdia ?dias))))

)

(defrule recopilacion-grupo::preguntas-calcula-nivel "Pregunta al usuario sus conocimientos"

?g <- (datos\_grupo (nivel ?nivel))

(test( < ?nivel 0))

=>

(bind ?puntuacio 0)

(bind ?formatos (create$ "Si" "No"))

(bind ?respuesta (pregunta-indice "Conoces 'El Grito' de Munch?" ?formatos))

(if (= ?respuesta 1) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "Si" "No"))

(bind ?respuesta (pregunta-indice "Conoces 'Las Meninas' de Velazquez?" ?formatos))

(if (= ?respuesta 1) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "Si" "No"))

(bind ?respuesta (pregunta-indice "Conoces 'El nacimiento de Venus' de Boticelli" ?formatos))

(if (= ?respuesta 1) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "La Gioconda (la Mona Lisa)." "El jardin de las delicias." "La ultima cena."))

(bind ?respuesta (pregunta-indice "Cual de los siguientes titulos no pertenece a un cuadro de Leonardo da Vinci " ?formatos))

(if (= ?respuesta 2) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "La persistencia de la memoria." "Alegoria de la poesia." "American Gothic." "Alegoria de la fe."))

(bind ?respuesta (pregunta-indice "Cual de estas obras es de Dali" ?formatos))

(if (= ?respuesta 1) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "El Greco." "Francisco de Goya." "Diego Velazquez."))

(bind ?respuesta (pregunta-indice "¿Quien pinto el cuadro 'Las Hilanderas'?" ?formatos))

(if (= ?respuesta 2) then (bind ?puntuacio (+ 1 ?puntuacio)))

(bind ?formatos (create$ "Klimt" "Tiziano" "Yanyez" "El Greco"))

(bind ?respuesta (pregunta-indice "¿Quien pinto 'El Beso'?" ?formatos))

(if (= ?respuesta 1) then (bind ?puntuacio (+ 1 ?puntuacio)))

(modify ?g (nivel ?puntuacio))

)

(defrule recopilacion-grupo::pasar-a-preferencias "Pasa a la recopilacion de preferencias"

(declare (salience 10))

?g <- (datos\_grupo (descripcion ~"desc")(edad ?e) (dias ?d) (horasdia ?horasdia) (nivel ?nivel) (tiempo ?tiempo))

(test (> ?e -1))

(test (> ?d -1))

(test (> ?nivel -1))

(test (> ?tiempo -1))

(test (> ?horasdia -1))

=>

(focus recopilacion-preferencias)

)

(deffacts recopilacion-preferencias::hechos-iniciales "Establece hechos para poder recopilar informacion"

(autores\_fav ask)

(tematicas\_obras ask)

(estilos\_fav ask)

(epocas\_fav ask)

(preferencias\_grupo )

)

(defrule recopilacion-preferencias::establecer-pintores-favoritos "Establece los pintores favoritos del grupo"

?hecho <- (autores\_fav ask)

?pref <- (preferencias\_grupo)

=>

(bind $?obj-pintores (find-all-instances ((?inst Pintor)) TRUE))

(bind $?nom-pintores (create$ ))

(loop-for-count (?i 1 (length$ $?obj-pintores)) do

(bind ?curr-obj (nth$ ?i ?obj-pintores))

(bind ?curr-nom (send ?curr-obj get-Nombre))

(bind $?nom-pintores(insert$ $?nom-pintores (+ (length$ $?nom-pintores) 1) ?curr-nom))

)

(bind ?escogido (pregunta-multirespuesta "Escoja sus pintores favoritos(o 0 en el caso contrario): " $?nom-pintores))

(assert (autores\_fav TRUE))

(bind $?respuesta (create$ ))

(loop-for-count (?i 1 (length$ ?escogido)) do

(bind ?curr-index (nth$ ?i ?escogido))

(if (= ?curr-index 0) then (assert (autores\_fav FALSE)))

(bind ?curr-autor (nth$ ?curr-index ?obj-pintores))

(bind $?respuesta(insert$ $?respuesta (+ (length$ $?respuesta) 1) ?curr-autor))

)

(retract ?hecho)

(modify ?pref (autores\_favoritos $?respuesta))

)

(defrule recopilacion-preferencias::establecer-tematicas-favorias "Establece las tematicas favoritas del grupo "

?hecho <- (tematicas\_obras ask)

?pref <- (preferencias\_grupo)

=>

(bind $?obj-tematicas (find-all-instances ((?inst Tematica)) TRUE))

(bind $?nom-tematicas (create$ ))

(loop-for-count (?i 1 (length$ $?obj-tematicas)) do

(bind ?curr-obj (nth$ ?i ?obj-tematicas))

(bind ?curr-nom (send ?curr-obj get-Nombre\_tematica))

(bind $?nom-tematicas(insert$ $?nom-tematicas (+ (length$ $?nom-tematicas) 1) ?curr-nom))

)

(bind ?escogido (pregunta-multirespuesta "Escoja sus tematicas favoritas(o 0 en el caso contrario): " $?nom-tematicas))

(bind $?respuesta (create$ ))

(assert (tematicas\_obras TRUE))

(loop-for-count (?i 1 (length$ ?escogido)) do

(bind ?curr-index (nth$ ?i ?escogido))

(if (= ?curr-index 0) then (assert (tematicas\_obras\_fav FALSE)))

(bind ?curr-tematica (nth$ ?curr-index ?obj-tematicas))

(bind $?respuesta(insert$ $?respuesta (+ (length$ $?respuesta) 1) ?curr-tematica))

)

(retract ?hecho)

(modify ?pref (tematicas\_obras\_fav $?respuesta))

)

(defrule recopilacion-preferencias::establecer-estilos-favoritos "Establece los estilos favoritos del grupo"

?hecho <- (estilos\_fav ask)

?pref <- (preferencias\_grupo)

=>

(bind $?obj-estilos (find-all-instances ((?inst Estilo)) TRUE))

(bind $?nom-estilos (create$ ))

(loop-for-count (?i 1 (length$ $?obj-estilos)) do

(bind ?curr-obj (nth$ ?i ?obj-estilos))

(bind ?curr-nom (send ?curr-obj get-Nombre\_estilo))

(bind $?nom-estilos(insert$ $?nom-estilos (+ (length$ $?nom-estilos) 1) ?curr-nom))

)

(bind ?escogido (pregunta-multirespuesta "Escoja sus estilos favoritos(o 0 en el caso contrario): " $?nom-estilos))

(bind $?respuesta (create$ ))

(assert (estilos\_fav TRUE))

(loop-for-count (?i 1 (length$ ?escogido)) do

(bind ?curr-index (nth$ ?i ?escogido))

(if (= ?curr-index 0) then (assert (estilos\_favoritos FALSE)))

(bind ?curr-estilos (nth$ ?curr-index ?obj-estilos))

(bind $?respuesta(insert$ $?respuesta (+ (length$ $?respuesta) 1) ?curr-estilos))

)

(retract ?hecho)

(modify ?pref (estilos\_favoritos $?respuesta))

)

(defrule recopilacion-preferencias::establecer-epocas-favoritas "Establece las epocas favoritas del grupo"

?hecho <- (epocas\_fav ask)

?pref <- (preferencias\_grupo)

=>

(bind $?obj-epocas (find-all-instances ((?inst Epoca)) TRUE))

(bind $?nom-epocas (create$ ))

(loop-for-count (?i 1 (length$ $?obj-epocas)) do

(bind ?curr-obj (nth$ ?i ?obj-epocas))

(bind ?curr-nom (send ?curr-obj get-Nombre\_epoca))

(bind $?nom-epocas(insert$ $?nom-epocas (+ (length$ $?nom-epocas) 1) ?curr-nom))

)

(bind ?escogido (pregunta-multirespuesta "Escoja sus epocas favoritas(o 0 en el caso contrario): " $?nom-epocas))

(bind $?respuesta (create$ ))

(assert (epocas\_fav TRUE))

(loop-for-count (?i 1 (length$ ?escogido)) do

(bind ?curr-index (nth$ ?i ?escogido))

(if (= ?curr-index 0) then (assert (epocas\_favoritas FALSE)))

(bind ?curr-epocas (nth$ ?curr-index ?obj-epocas))

(bind $?respuesta(insert$ $?respuesta (+ (length$ $?respuesta) 1) ?curr-epocas))

)

(retract ?hecho)

(modify ?pref (epocas\_favoritas $?respuesta))

)

(defrule recopilacion-preferencias::pasar\_procesado\_datos "Pasa al modulo de procesado de datos"

(declare (salience -1))

?h1 <- (autores\_fav TRUE|FALSE)

?h2 <- (tematicas\_obras TRUE|FALSE)

?h3 <- (estilos\_fav TRUE|FALSE)

?h4 <- (epocas\_fav TRUE|FALSE)

=>

(focus procesado-datos)

(printout t "Procesando los datos obtenidos..." crlf)

)

;;; Modulo procesado de datos ---------------------------------------------------

(defrule procesado-datos::anadir-cuadros "Se añaden todos los cuadros"

(declare (salience 10))

=>

(bind $?lista (find-all-instances ((?inst Cuadro)) TRUE))

(progn$ (?curr-con ?lista)

(make-instance (gensym) of Recomendacion (nombre\_cuadro ?curr-con)(puntuacion 0))

)

(printout t "..." crlf)

)

(defrule procesado-datos::aux-autores "Crea hechos para poder procesar los autores favoritos"

(preferencias\_grupo (autores\_favoritos $?gen))

?hecho <- (autores\_fav ?aux)

(test (or (eq ?aux TRUE) (eq ?aux FALSE)))

=>

(retract ?hecho)

(if (eq ?aux TRUE)then

(progn$ (?curr-gen $?gen)

(assert (autores ?curr-gen))

)

)

(printout t "..." crlf)

)

(defrule procesado-datos::aux-tematicas "Crea hechos para poder procesar las tematicas favoritas"

(preferencias\_grupo (tematicas\_obras\_fav $?gen))

?hecho <- (tematicas\_obras ?aux)

(test (or (eq ?aux TRUE) (eq ?aux FALSE)))

=>

(retract ?hecho)

(if (eq ?aux TRUE)then

(progn$ (?curr-gen $?gen)

(assert (tematicas ?curr-gen))

)

)

(printout t "..." crlf)

)

(defrule procesado-datos::aux-estilos "Crea hechos para poder procesar los estilos favoritos"

(preferencias\_grupo (estilos\_favoritos $?gen))

?hecho <- (estilos\_fav ?aux)

(test (or (eq ?aux TRUE) (eq ?aux FALSE)))

=>

(retract ?hecho)

(if (eq ?aux TRUE)then

(progn$ (?curr-gen $?gen)

(assert (estilos ?curr-gen))

)

)

)

(defrule procesado-datos::aux-epocas "Crea hechos para poder procesar las espocas favoritas"

(preferencias\_grupo (epocas\_favoritas $?gen))

?hecho <- (epocas\_fav ?aux)

(test (or (eq ?aux TRUE) (eq ?aux FALSE)))

=>

(retract ?hecho)

(if (eq ?aux TRUE)then

(progn$ (?curr-gen $?gen)

(assert (epocas ?curr-gen))

)

)

(printout t "..." crlf)

)

;;; Aplicamos los filtros de las preguntas -------------------------------------------------

(defrule procesado-datos::valorar-nivel-mayor-a-4 "Se mejora la puntuacion de los cuadros"

(datos\_grupo (nivel ?nivel))

(test (> ?nivel 3))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

?cont <-(object (is-a Cuadro) (Relevancia ?relevancia) (Complejidad ?complejidad))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorado-nivel ?cont ?nivel))

=>

(if (> ?complejidad 10000) then

(bind ?p (+ ?p 60))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) "Tiene una complejidad alta acorde al nivel del visitante -> +60"))

)

(if (< ?relevancia 4) then

(bind ?p (+ ?p 40))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) "Tiene una relevancia mediana/baja acorde al interes del visitante -> +40"))

)

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorado-nivel ?cont ?nivel))

(printout t "Valorando nivel del grupo..." crlf)

)

(defrule procesado-datos::valorar-nivel-menor-a-4 "Se mejora la puntuacion de los cuadros"

(datos\_grupo (nivel ?nivel))

(test (< ?nivel 4))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

?cont <-(object (is-a Cuadro) (Relevancia ?relevancia) (Complejidad ?complejidad))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorado-nivel ?cont ?nivel))

=>

(if (< ?complejidad 10000) then

(bind ?p (+ ?p 30))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) "Tiene una complejidad baja acorde al nivel del visitante -> +30"))

)

(if (> ?relevancia 2) t¡hen

(bind ?p (+ ?p 70))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) "Tiene una relevancia alta acorde al interes del visitante -> +70"))

)

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorado-nivel ?cont ?nivel))

(printout t "Valorando nivel del grupo..." crlf)

)

(defrule procesado-datos::valorar-autores-favoritos "Se mejora la puntuacion de los cuadros de autores favoritos"

?hecho <- (autores ?auto)

?cont <-(object (is-a Cuadro) (Pintado\_por ?autor))

(test (eq (instance-name ?auto) ?autor))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorado-autor-favorito ?cont ?auto)) ;?auto al final

=>

(bind ?p (+ ?p 50))

(bind ?text (str-cat "Pertenece al autor favorito: " (send ?auto get-Nombre) " -> +50"))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) ?text))

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorado-autor-favorito ?cont ?auto))

(printout t "Comprobando autores favoritos..." crlf)

)

(defrule procesado-datos::valorar-tematicas-favoritas "Se mejora la puntuacion de las tematicas favoritas"

?hecho <- (tematicas ?tem)

?cont <-(object (is-a Cuadro) (Tematica\_cuadro ?tema))

(test (eq (instance-name ?tem) ?tema))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorar-tematicas-favoritas ?cont ?tem))

=>

(bind ?p (+ ?p 50))

(bind ?text (str-cat "Pertenece a la tematica favorita: " (send ?tem get-Nombre\_tematica) " -> +50"))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) ?text))

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorar-tematicas-favoritas ?cont ?tem))

(printout t "Comprobando tematicas favoritas..." crlf)

)

(defrule procesado-datos::valorar-estilos-favoritos "Se mejora la puntuacion de los estilos favoritos"

?hecho <- (estilos ?estilo)

?cont <-(object (is-a Cuadro) (Estilo\_cuadro ?estilos))

(test (eq (instance-name ?estilo) ?estilos))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorar-estilos-favoritos ?cont ?estilo))

=>

(bind ?p (+ ?p 50))

(bind ?text (str-cat "Pertenece al estilo favorito: " (send ?estilo get-Nombre\_estilo) " -> +50"))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) ?text))

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorar-estilos-favoritos ?cont ?estilo))

(printout t "Comprobando estilos favoritos..." crlf)

)

(defrule procesado-datos::valorar-epocas-favorias "Se mejora la puntuacion de las epocas favoritas"

?hecho <- (epocas ?epoca)

?cont <-(object (is-a Cuadro) (Epoca\_cuadro ?epocas))

(test (eq (instance-name ?epoca) ?epocas))

?rec <- (object (is-a Recomendacion) (nombre\_cuadro ?conta) (puntuacion ?p) (justificaciones $?just))

(test (eq (instance-name ?cont) (instance-name ?conta)))

(not (valorar-epocas-favorias ?cont ?epoca))

=>

(bind ?p (+ ?p 50))

(bind ?text (str-cat "Pertenece a la epoca favorita: " (send ?epoca get-Nombre\_epoca) " -> +50"))

(bind $?just (insert$ $?just (+ (length$ $?just) 1) ?text))

(send ?rec put-puntuacion ?p)

(send ?rec put-justificaciones $?just)

(assert (valorar-epocas-favorias ?cont ?epoca))

(printout t "Comprobando epocas favoritas..." crlf)

)

(defrule procesado-datos::pasar-a-generacion "Pasa al modulo de generacion de respuestas"

(declare(salience -10))

=>

(printout t "Generando respuesta..." crlf)

(focus generacion\_soluciones)

)

;;; Módulo de generacion de respuestas -------------------------------------------------

(defrule generacion\_soluciones::crea-lista-recomendaciones "Se crea una lista de recomendaciones para ordenarlas"

(not (lista-rec-desordenada))

=>

(assert (lista-rec-desordenada))

)

(defrule generacion\_soluciones::anyadir-recomendacion "Anyade una recomendacion a la lista de recomendaciones"

(declare (salience 10))

?rec <- (object (is-a Recomendacion))

?hecho <- (lista-rec-desordenada (recomendaciones $?lista))

(test (not (member$ ?rec $?lista)))

=>

(bind $?lista (insert$ $?lista (+ (length$ $?lista) 1) ?rec))

(modify ?hecho (recomendaciones $?lista))

)

(defrule generacion\_soluciones::crea-lista-ordenada "Se crea una lista ordenada de contenido"

(not (lista-rec-ordenada))

(lista-rec-desordenada (recomendaciones $?lista))

=>

(bind $?resultado (create$ ))

(while (not (eq (length$ $?lista) 0)) do

(bind ?curr-rec (maximo-puntuacion $?lista))

(bind $?lista (delete-member$ $?lista ?curr-rec))

(bind $?resultado (insert$ $?resultado (+ (length$ $?resultado) 1) ?curr-rec))

)

(assert (lista-rec-ordenada (recomendaciones $?resultado)))

(printout t "Ordenando obras de arte..." crlf)

)

(defrule generacion\_soluciones::asigna-contenido-a-dias "Se asigna los contenidos recomendados a dias"

?g <- (datos\_grupo (dias ?dias) (horasdia ?horas) (descripcion ?descripcion) (nivel ?nivel));

(lista-rec-ordenada (recomendaciones $?recs))

(not (lista-dias))

=>

(bind ?horas (\* ?horas 60))

(bind $?lista (create$ ))

(while (not(= (length$ $?lista) ?dias)) do

(bind $?lista (insert$ $?lista (+ (length$ $?lista) 1) (make-instance (gensym) of Dia (tiempo-maximo ?horas))))

)

(bind ?i 1)

(bind ?rec-ant FALSE)

(while (and (> (length$ $?recs) 0) (<= ?i ?dias))

(bind ?dia (nth$ ?i $?lista))

(bind $?recs-dia (create$ ))

(bind ?t-max (send ?dia get-tiempo-maximo))

(bind ?t-ocu 0)

(bind ?try 1)

(bind ?asignados 0)

(bind ?j 1)

(while (and(and(< ?t-ocu ?t-max) (< ?try 4)) (> (length$ $?recs) 0) (<= ?j (length$ ?recs))) do

(bind ?rec (nth$ ?j $?recs))

(bind ?cont (send ?rec get-nombre\_cuadro))

(bind ?a (send ?cont get-Complejidad))

(if (or (eq ?descripcion "Pareja") (eq ?descripcion "Individual")) then

(if (> ?a 120000) then (bind ?t 13))

(if (and (> ?a 13000) (< ?a 120000)) then (bind ?t 10))

(if (and (> ?a 2000) (< ?a 13000)) then (bind ?t 6))

(if (and (> ?a 0) (< ?a 2000)) then (bind ?t 4))

)

(if (eq ?descripcion "Grupo pequeno (3-12)") then

(if (> ?a 120000) then (bind ?t 16))

(if (and (> ?a 13000) (< ?a 120000)) then (bind ?t 12))

(if (and (> ?a 2000) (< ?a 13000)) then (bind ?t 8))

(if (and (> ?a 0) (< ?a 2000)) then (bind ?t 5))

)

(if (eq ?descripcion "Grupo mediano (13-25)") then

(if (> ?a 120000) then (bind ?t 18))

(if (and (> ?a 13000) (< ?a 120000)) then (bind ?t 14))

(if (and (> ?a 2000) (< ?a 13000)) then (bind ?t 10))

(if (and (> ?a 0) (< ?a 2000)) then (bind ?t 7))

)

(if (eq ?descripcion "Grupo grande (+25)") then

(if (> ?a 120000) then (bind ?t 20))

(if (and (> ?a 13000) (< ?a 120000)) then (bind ?t 15))

(if (and (> ?a 2000) (< ?a 13000)) then (bind ?t 12))

(if (and (> ?a 0) (< ?a 2000)) then (bind ?t 8))

)

(if (< (+ ?t-ocu ?t) ?t-max)

then

(bind ?t-ocu (+ ?t-ocu ?t))

(bind ?try 1)

(bind ?asignados (+ ?asignados 1))

(bind ?recs-dia (insert$ $?recs-dia (+ (length$ $?recs-dia) 1) ?rec))

(bind $?recs (delete-member$ $?recs ?rec))

else

(bind ?try (+ ?try 1))

)

(bind ?j (+ ?j 1))

)

(send ?dia put-recomendaciones $?recs-dia)

(bind ?i (+ ?i 1))

)

(assert (lista-dias (dias $?lista)))

(printout t "Computando una ruta optima de visitas..." crlf)

)

(defrule generacion\_soluciones::ordena-por-salas "Ordena cada dia por salas."

(lista-dias (dias $?lista))

=>

(progn$ (?curr-dia $?lista)

(bind $?resultado (create$ ))

(bind $?recs (send ?curr-dia get-recomendaciones))

(while (not (eq (length$ $?recs) 0)) do

(bind ?curr-rec (orden-sala $?recs))

(bind $?recs (delete-member$ $?recs ?curr-rec))

(bind $?resultado (insert$ $?resultado (+ (length$ $?resultado) 1) ?curr-rec))

)

(send ?curr-dia put-recomendaciones $?resultado)

)

(assert (dias-orden-sala (dias $?lista)))

)

(defrule generacion\_soluciones::pasar-a-resultados "Se pasa al modulo de presentacion"

(dias-orden-sala)

=>

(focus resultados\_al\_grupo)

)

(defrule resultados\_al\_grupo::mostrar-respuesta "Muestra el contenido escogido"

(dias-orden-sala (dias $?dias))

(not (final))

=>

(printout t crlf)

(format t "Esta es nuestra recomendacion de ruta para el grupo. Esperamos que la disfruteis.")

(printout t crlf)

(format t "%n")

(printout t crlf)

(printout t "============================================" crlf)

(bind ?i 0)

(progn$ (?curr-dia $?dias)

(bind ?i(+ ?i 1))

(format t "Dia %d" ?i)

(printout t crlf)

(printout t (send ?curr-dia imprimir))

)

(assert (final))

)