Corrigió Daniela

PLP - Segundo Parcial - 1er cuatrimestre de 2024

#Orden	Nro. Libreta	Apellido y Nombre	Ej1	Ej2	Ej3	Nota Final
12	1028/22	Andrés, Sebastián Franco	B-	B-	R	A

Este examen se aprueba obteniendo al menos dos ejercicios bien menos (B-) y uno regular (R). Las notas para cada ejercicio son: -, I, R, B-, B. Entregar cada ejercicio en hojas separadas. Poner nombre, apellido y número de orden en todas las hojas, y numerarlas. Se puede utilizar todo lo definido en las prácticas y todo lo que se dio en clase, colocando referencias claras. El orden de los ejercicios es arbitrario. Recomendamos leer el parcial completo antes de empezar a resolverlo.

Ejercicio 1 - Resolución

a) Representar en forma clausal las siguientes fórmulas de lógica de primer orden:

1. $\forall X.(\texttt{Radio}(X) \implies \texttt{Ruidoso}(X))$
   Todas las radios son ruidosas.
2. $\forall X.\forall Y.((\texttt{Posee}(X,Y) \land \texttt{Libro}(Y)) \implies \neg\exists Z.(\texttt{Posee}(X,Z) \land \texttt{Revista}(Z)))$
   Toda persona que posee un libro no posee revistas.
3. $\forall X.(\texttt{EsDormilon}(X) \implies \neg\exists Y.(\texttt{Posee}(X,Y) \land \texttt{Ruidoso}(Y)))$
   Las personas que son dormilonas no poseen nada ruidoso.
4. $\exists X.(\texttt{Posee}(\texttt{pepe},X) \land (\texttt{Libro}(X) \lor \texttt{Radio}(X)))$
   Pepe posee un libro o una radio.

b) Utilizando resolución, determinar si la siguiente fórmula es consecuencia del conjunto anterior:
$\texttt{EsDormilon}(\texttt{pepe}) \implies \neg\exists Z.(\texttt{Posee}(\texttt{pepe},Z) \land \texttt{Revista}(Z))$
Si Pepe es dormilón, entonces no posee revistas.

Indicar la sustitución utilizada en cada paso. Es importante tener un plan, ya sea escrito o en la cabeza.

c) Mirando la resolución utilizada en el punto anterior, ¿fue SLD? Justificar.

Ejercicio 2 - Programación Lógica

Implementar los predicados respetando en cada caso la instanciación pedida. Los generadores deben cubrir todas las instancias válidas de aquello que generan sin repetir dos veces la misma. No usar cut (!) ni predicados de alto orden como `setof`, con la única excepción de `not`.

En este ejercicio ayudaremos a un grupo de jurados a analizar las notas de las materias que cursaron algunos estudiantes para presentarse a concurso. Se cuenta con el predicado `estudiante(?E)` que es verdadero cuando `E` es un estudiante, y con el predicado `notas(-XS)` que instancia en `XS` una lista de triplas correspondientes a las notas de los estudiantes en cada materia de la siguiente forma: `(Estudiante,Materia,Nota)`. En la lista de notas pueden aparecer aplazos, pero se asume que está bien formada (es decir, puede haber muchos aplazos para un estudiante en una materia, pero a lo sumo un aprobado). Se pide:

a) Definir el predicado `tieneMateriaAprobada(+E,+M)` que es verdadero cuando el estudiante `E` tiene la materia `M` aprobada (es decir, la nota es mayor o igual a 4).

b) Definir el predicado `eliminarAplazos(+NS,-L)` que es verdadero cuando `NS` es una lista de notas y `L` es la misma lista, pero eliminando los aplazos. Los aplazos sólo pueden eliminarse si el estudiante finalmente aprobó la materia. Por ejemplo:

```
?- eliminarAplazos([(juan,plp,3),(juan,plp,9),(maria,tlen,2)],L).
L = [(juan,plp,9),(maria,tlen,2)].
```

c) Definir el predicado `promedio(+A,-P)` que es verdadero cuando `A` es un estudiante, y `P` es el promedio de todas sus notas luego de eliminar los aplazos.
Sugerencia: armar una lista de notas de `A`.

d) Definir el predicado `mejorEstudiante(-A)` que es verdadero cuando `A` es el estudiante cuyo promedio es el más alto. Puede haber más de una solución en caso de que haya más de un estudiante con el promedio más alto. En este inciso no está permitido utilizar estructuras auxiliares.

Ejercicio 3 - Objetos y Deducción Natural

a) I. Considerar las siguientes definiciones:

```
Object subclass: A [
  a: x b: y                  aC
aB  ^ x a: (y c) b: self.
      aB   aC     anA
            =
            1
  c
   ^ 2.
]
```

```
B subclass: C [            1
  a: x b: y
   ^ x.

  c
   ^ [self a: super c b: self].
]
              11
```

```
A subclass: B [
  a: x b: y        anA
1  ^ y c + x value.
     =2  + 1    = 13
  c
   ^ 1.
]
```

Hacer una tabla donde se indique, en orden, cada mensaje se envía, qué objeto lo recibe, con qué colaboradores, en qué clase está el método respectivo, y cuál es el resultado final de cada colaboración tras ejecutar el siguiente código:

```
(A new) a: (B new) b: (C new)
```

II. Implementar un método para el mensaje `#divisores`, cuyo objeto receptor es un número entero, que devuelve una colección con sus divisores.
<u>Sugerencia:</u> utilizar el mensaje binario `#'\\'` que devuelve el resto de la división entera entre dos números.

b) Demostrar en deducción natural que vale la siguiente fórmula sin usar principios de razonamiento clásicos:

$$\forall X.\forall Y.(\exists Z.(P(X,Z) \wedge P(Z,Y)) \implies \exists W.P(X,W))$$

Orden 12

Sebastián Andres	1020/22

1) Resolución:

(a) PASAJE A FORMA CLAUSAL:

- 1) $\forall X.(RADIO(X) \Rightarrow RUIDOSO(X))$

$\equiv \forall X.(\neg RADIO(X) \vee RUIDOSO(X))$

$\{\neg RADIO(X_1), RUIDOSO(X_1)\}$ ✓

- 2) $\forall X. \forall Y\ ((Posee(X,Y) \wedge Libro(Y)) \Rightarrow \neg \exists Z.(Posee(X,Z) \wedge Revista(Z)))$

$\equiv \forall X. \forall Y.\ (\neg(Posee(X,Y) \wedge Libro(Y)) \vee \neg \exists Z\ (Posee(X,Z) \wedge Revista(Z))$

$\forall X. \forall Y.\ (\neg Posee(X,Y) \vee \neg Libro(Y)) \vee (\forall Z. \neg(Posee(X,Z) \wedge Revista(Z))$

$\forall X. \forall Y.\ (\neg Posee(X,Y) \vee \neg Libro(Y)) \vee (\forall Z.\ \neg Posee(X,Z) \vee \neg Revista(Z)$

$\forall X \forall Y \forall Z.\ (\neg Posee(X,Y) \vee \neg Libro(Y) \vee \neg Posee(X,Z) \vee \neg Revista(Z)$

$\{\neg Posee(X_2,Y_2), \neg Libro(Y_2), \neg Posee(X_2,Z_2), \neg Revista(Z_2)\}$ ✓

- 3) $\forall X.(esDormilon(X) \Rightarrow \neg \exists Y.(Posee(X,Y) \wedge Ruidoso(Y)))$.

$\forall X.\ \neg esDormilón(X) \vee \neg \exists Y.(Posee(X,Y) \wedge Ruidoso(Y)))$.

$\forall X.\ \neg esDormilón(X) \vee \forall Y.\ \neg(Posee(X,Y) \wedge Ruidoso(Y)))$.

$\forall X.\ \neg esDormilon(X) \vee \forall Y.\ \neg Posee(X,Y) \vee \neg Ruidoso(Y))$

$\forall X \forall Y.\ \neg esDormilon(X) \vee \neg Posee(X,Y) \vee \neg Ruidoso(Y))$

$\{\neg esDormilon(X_3), \neg Posee(X_3,Y_3), \neg Ruidoso(Y_3)\}$ ✓

-4) $\exists X . (Posee(Pepe, X) \wedge (Libro(X) \vee Radio(X)))$

$\exists X . (Posee(Pepe, X) \wedge (Libro(X) \vee Radio(X)))$

$Posee(Pepe, C) \wedge (Libro(C) \vee Radio(C))$.

$$\{ \underbrace{\{Posee(Pepe, C)\}}_{(4)}, \underbrace{\{Libro(C), Radio(C)\}}_{(5)} \}$$

B) Utilizando resolución fvp.

$\sigma = EsDormilon(Pepe) \Rightarrow \neg \exists Z . (Posee(Pepe, Z) \wedge RevistA(Z))$.

$\neg\sigma = \neg(\neg esDormilon(Pepe) \vee \neg \exists Z . (Posee(Pepe, Z) \wedge RevistA(Z$

$\equiv esDormilón(Pepe) \wedge \exists Z . (Posee(Pepe, Z) \wedge RevistA(Z)))$

$\equiv esDormilon(Pepe) \wedge (Posee(Pepe, K) \wedge RevistA(K))$

$$\underbrace{\{esDormilón(Pepe)\}}_{(6)} \quad \underbrace{\{Posee(Pepe, K)\}}_{(7)} \quad \underbrace{\{revistA(K)\}}_{(8)}$$

Ordw 12

Sebastián Andrés 10/28/22

2

Tenemos:

(1) $\{\neg RADIO(x_1), Ruidoso(x_1)\}$

(2) $\{\neg Posee(x_2, y_2), \neg Libro(y_2), \neg Posee(x_2, z_2), \neg Revista(z_2)\}$

(3) $\{\neg esDormilón(x_3), \neg Posee(x_3, y_3), \neg Ruidoso(y_3)\}$

(4) $\{Posee(Pepe, c)\}$

(5) $\{Libro(c), RADIO(c)\}$

(6) $\{esDormilon(Pepe)\}$

(7) $\{Posee(Pepe, k)\}$

(8) $\{revista(k)\}$

PLAN: qvq "si Pepe es dormilón $\longrightarrow$ no tiene revistas" por ABS.

Planteamos:

- Pepe es dormilón y tiene revistas.
- Las personas dormilonas no tienen cosas ruidosas ③ $\Rightarrow$ Pepe no tiene Radio (por ①)

$\Rightarrow$ Pepe tiene un libro (por ④).

$\Rightarrow$ Pepe no tiene revistas (por ②)

Luego llegamos a un ABS !!

- ③ y ⑥ ya existen 7 y 8

$\Rightarrow$ ~~⑦~~ $= \delta\{\neg Posee(X_3, Y_3), \neg Ruidoso(Y_3)\}$

$\delta = \{X_3 \leftarrow Pepe\}$

$7 = \{\neg Posee(Pepe, Y_3), \neg Ruidoso(Y_3)\}$ ✓

- ③ y ①.

$\Rightarrow$ ⑧ $= \delta\{\neg Radio(X_1), \neg Posee(Pepe, Y_3)\}$ $\neg esDormilón(\cancel{Pepe}\ X_3)$

$\delta = \{X_1 \leftarrow Y_3\}$

⑧ $= \{\neg Radio(Y_3), \neg Posee(\cancel{Pepe}\ X_3, Y_3)\}$

- ⑧ y ⑤ :

$\Rightarrow 9 = \delta\{Libro(c), \neg Posee(Pepe, Y_3),$ $\neg esD(X_3)$ $\}$

$\delta = \{Y_3 \leftarrow C\}$

$9 = \{Libro(C), \neg Posee(\cancel{Pepe}\ X_3, C)\}$

- ⑨ y ②.

$\Rightarrow 10 = \delta\{\neg Posee(X_2, Y_2), \neg Posee(X_2, Z_2), \neg RevistA(Z_2)\}$

$\delta = \{Y_2 \leftarrow C\}$

$\neg Posee(X_3, c), \neg esD(X_3),$

$10 = \{\neg Posee(X_2, C), \neg Posee(X_2, Z_2), \neg RevistA(Z_2)\}$

- ⑩ y ⑧ ¿cuál 8?

$\Rightarrow 11 = \{\neg Posee(X_2, C), \neg Posee(X_2, K)\}$

$\uparrow$

$\delta = \{Z_2 \leftarrow K\}$

Ordv 12

Sebastián Andrés | 1028/22

(11) y (4).

$\Rightarrow 12 = \{\neg Posee(Pepe, K)\}$

$\sigma = \{x_2 \leftarrow Pepe\}$

(12) y (7).

$\Rightarrow 13 = \{\}$

Luego $\vdash \sigma$. (σ es válida).

(c) Fue SLD?

NO. Aunque lo hice de forma lineal y arranqué por una cláusula objetivo, también usé cláusulas con más de un literal positivo (cláusulas que no son de Horn).

$\Rightarrow$ NO fue SLD.

Orden 12

(4)

Sebastian Andés	1028/22

(2) Programación Lógica

Contamos con:

- ESTUDIANTE(?E).
- NOTAS(-XS), XS LISTA DE TRIPLAS <ESTUDIANTE, MATERIA, NOTA>.

(a) Definir predicado:

```
% tieneMateriaAprobada(+E, +M).
tieneMateriaAprobada(E,M) :- ESTUDIANTE(E), NOTAS(XS),
                              HayTriplaAprobada(E,M,XS).

% hayTriplaAprobada(+E, +M, +Xs).
(not(hayTriplaAprobada(_, _, []))).
hayTriplaAprobada(E, M, [(E,M,N)|Xs]) :- N >= 4.
hayTriplaAprobada(E, M, [(E,M,N)|Xs]) :-
        N < 4,
        hayTriplaAprobada(E,M,Xs).
hayTriplaAprobada(E, M, [(X,_,_)|Xs]) :-
        X \= E,
        hayTriplaAprobada(E,M,Xs).
```

¿¿Esto no es def. de un predicado

¿Por qué no usás member?

(b) Definir predicado:

% eliminarAplazos(+NS, -L).

IDEA: No hay en L un aplazo de un estudiante que aprobó (pero sí está el resto de notas).

% eliminarAplazos(+NS, -L).

• eliminarAplazos([], []).

% Se me ocurre analizar cada caso posible con
% la cabeza de la lista.
% H ∈ Lista ⟺ (H aprobada) ∨
% (H aplazo ∧ el estudiante de H no aprobó esa materia de H)

• eliminarAplazos([(E,M,N)|Nss], [X|Xs]) :-
esAplazo(E,M,N),
tieneMateriaAprobada(E,M), } Aplazo que no está en L.
eliminarAplazos(Nss, [X|Xs]).

xq lo escribís así? si no accedés a sus componentes.

• eliminarAplazos([(E,M,N)|Nss], [(E,M,N)|Xs]) :-
esAplazo(E,M,N),
not(tieneMateriaAprobada(E,M)), } Aplazo que está en L
eliminarAplazos(Nss, Xs).

• eliminarAplazos([(E,M,N)|Nss], [(E,M,N)|Xs]) :-
not(esAplazo(E,M,N)), } nota aprobada.
eliminarAplazos(Nss, Xs).

% esAplazo(+E, +M, +N).
esAplazo(E,_,N) :- estudiante(E), N ~~>~~= 4.

?¿?

Orden 12

Sebastián Andrés 1020/22 (S)

→ estudiante
→ promedio (sin aplazos).

% promedio(+A, -P)

```
• promedio(A,P) :- estudiante(A),
                   notas(XS),
                   eliminarAplazos(XS,L),
                   notasDeAlumno(A,L,L1),
                   length(L1,N),
                   sumlist(L1,N1),
                   P is N1/N.
```

Estás agregando la triple, no solo la nota.

```
%notasDeAlumno(+E, +L, -S)
• notasDeAlumno(E,[],[]).
• notasDeAlumno(E,[(E,M,N)|Lss],[(E,M,N)|Xss]) :-
      notasDeAlumno(E,Lss,Xss).
• notasDeAlumno(E,[(Y,_,_)|Lss],S) :-
      Y \= E, notasDeAlumno(E,Lss,S).
      (evitar repetidos).
```

N?

% mejorEstudiante(-A).

mayor promedio

ge lo instancia promedio según tu implementación

```
• mejorEstudiante(A) :- estudiante(A),
                        promedio(A,P),
                        not((estudiante(A2),
                             promedio(A2,P2),
                             P2 > P)).
```

✓

Sebastián Andrés	10/28/22

3 - Objetos y Deducción Natural.

Objeto	Mensaje	Colaboradores	Ubicación	Respuesta
A	new	—	OBJECT	anA
anA	a: b:	aB, aC	A	3
B	new	—	OBJECT	aB
C	new	—	OBJECT	aC
aB	a: b:	[...] ~~(aC c)~~, anA	B	3
aC	c	—	C	~~1~~ El bloque
aC	a: b:	(sper) aC, aC	C	1
aC	c (super)	—	B	1
anA	c	—	A	2
2	+	1	SmallInteger	3

Falta el value al bloque.

están desordenados!

Donde { anA : una instancia de A
aB : una instancia de B
aC : una instancia de C.

Hice/
Hay algunas anotaciones en el código de la consigna para clarificar la tabla

(II) Método para #Divisores.

```
SmallInteger >> Divisores:
  |res, index|
  res := OrderedCollection new.
  1 to: self do: [ :index |
   ((index \\ self) = 0) ifTrue: [res addLast: index]
  ].
  ^res.
```

* Los paréntesis no son necesarios pero aclaran la lectura.

(b) Demostrar con DN.

" $\forall x. \forall y. (\exists z. (P(x,z) \land P(z,y)) \Rightarrow \exists w. P(x,w))$ ".

Sebastián Andrés 10/08/22

Ej: DEDUCCIÓN NATURAL

X

$\overline{}$ Ax

$\Delta \vdash P(x,z) \wedge P(z,y)$

$\overline{}$ $\wedge e_2$

Ax

$\Gamma \vdash \forall x \forall y\, \exists z\, (P(x,z) \wedge P(z,y))$ $\forall e$

$\Gamma \vdash \forall y\, \exists z\, (P(x,z) \wedge P(z,y))$ $\forall e$

$\Gamma \vdash \exists z.(P(x,z) \wedge P(z,y)) \qquad \Gamma, (P(x,z) \wedge P(z,y)) \vdash P(x,\underset{z}{z})$ $\cancel{\exists e}$

$\Gamma = \forall x \forall y\, (\exists z (P(x,z) \wedge P(z,y))) \vdash P(x,\underset{z}{z})$

$\exists i_{\{w=z\}}$

$\forall x \forall y\, (\exists z\, (P(x,z) \wedge P(z,y)) \vdash \exists w.\, P(x,w)$

$\cancel{\Rightarrow i}$

$\vdash \forall x \forall y\, (\exists z.\, (P(x,z) \wedge P(z,y)) \Rightarrow \exists w.\, P(x,w))$

No lo podés aplicar xq z está libre acá

llegan al final dos $\forall$s.