

## PLP - Segundo Parcial - 1<sup>er</sup> cuatrimestre de 2024

#Orden	Nro. Libreta	Apellido y Nombre	Ej1	Ej2	Ej3	Nota Final
26	348/11	BEKENSTEIN, JONATHAN	B-	B	B	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:

- $\forall X.(\text{Radio}(X) \implies \text{Ruidoso}(X))$   
*Todas las radios son ruidosas.*
- $\forall X.\forall Y.((\text{Posee}(X,Y) \wedge \text{Libro}(Y)) \implies \neg\exists Z.(\text{Posee}(X,Z) \wedge \text{Revista}(Z)))$   
*Toda persona que posee un libro no posee revistas.*
- $\forall X.(\text{EsDormilon}(X) \implies \neg\exists Y.(\text{Posee}(X,Y) \wedge \text{Ruidoso}(Y)))$   
*Las personas que son dormilonas no poseen nada ruidoso.*
- $\exists X.(\text{Posee}(\text{pepe}, X) \wedge (\text{Libro}(X) \vee \text{Radio}(X)))$   
*Pepe posee un libro o una radio.*

b) Utilizando resolución, determinar si la siguiente fórmula es consecuencia del conjunto anterior:

$\text{EsDormilon}(\text{pepe}) \implies \neg\exists Z.(\text{Posee}(\text{pepe}, Z) \wedge \text{Revista}(Z))$   
*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:

- 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).
- 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)].`

- 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
- 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  
  ^ x a: (y c) b: self.  
  
  c  
  ^ 2.  
]
```

```
B subclass: C [  
  a: x b: y  
  ^ x.  
  
  c  
  ^ [self a: super c b: self].  
]
```

```
A subclass: B [  
  a: x b: y  
  ^ y c + x value.  
  
  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.

Sugerencia: 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:

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

$$\begin{aligned} & \forall x. (\text{Radio}(x) \Rightarrow \text{Ruidoso}(x)) \\ & \equiv \forall x. (\neg \text{Radio}(x) \vee \text{Ruidoso}(x)) \\ & \equiv \{ \{ \neg \text{Radio}(x), \text{Ruidoso}(x) \} \} \quad \checkmark \end{aligned}$$

$$\begin{aligned} & \forall x. \forall y. ((\text{Posee}(x, y) \wedge \text{Libro}(y)) \Rightarrow \neg \exists z. (\text{Posee}(x, z) \wedge \text{Revista}(z))) \\ & \equiv \forall x. \forall y. (\neg (\text{Posee}(x, y) \wedge \text{Libro}(y)) \vee \forall z. \neg (\text{Posee}(x, z) \wedge \text{Revista}(z))) \\ & \equiv \forall x. \forall y. (\neg \text{Posee}(x, y) \vee \neg \text{Libro}(y) \vee \forall z. (\neg \text{Posee}(x, z) \vee \neg \text{Revista}(z))) \\ & \equiv \forall x. \forall y. \forall z. (\neg \text{Posee}(x, y) \vee \neg \text{Libro}(y) \vee \neg \text{Posee}(x, z) \vee \neg \text{Revista}(z)) \\ & \equiv \{ \{ \neg \text{Posee}(x, y), \neg \text{Libro}(y), \neg \text{Posee}(x, z), \neg \text{Revista}(z) \} \} \quad \checkmark \end{aligned}$$

$$\begin{aligned} & \forall x. (\neg \text{EsDormilon}(x) \Rightarrow \neg \exists y. (\text{Posee}(x, y) \wedge \text{Ruidoso}(y))) \\ & \equiv \forall x. (\neg \neg \text{EsDormilon}(x) \vee \forall y. \neg (\text{Posee}(x, y) \wedge \text{Ruidoso}(y))) \\ & \equiv \forall x. \forall y. (\neg \neg \text{EsDormilon}(x) \vee \neg \text{Posee}(x, y) \vee \neg \text{Ruidoso}(y)) \\ & \equiv \{ \{ \neg \neg \text{EsDormilon}(x), \neg \text{Posee}(x, y), \neg \text{Ruidoso}(y) \} \} \quad \checkmark \end{aligned}$$

$$\begin{aligned} & \exists x. (\text{Posee}(\text{pepe}, x) \wedge (\text{Libro}(x) \vee \text{Radio}(x))) \\ & \rightarrow \text{Posee}(\text{pepe}, k) \wedge (\text{Libro}(k) \vee \text{Radio}(k)) \end{aligned}$$

La skolemización preserva la satisfacibilidad pero no la validez, es decir no son fórmulas equivalentes.  $\checkmark$

~~$$\equiv \{ \{ \text{Posee}(\text{pepe}, k), \text{Libro}(k), \text{Radio}(k) \} \}$$~~

$$\equiv \{ \{ \text{Posee}(\text{pepe}, k) \}, \{ \text{Libro}(k), \text{Radio}(k) \} \} \quad \checkmark$$

$$\uparrow: \text{EsDormilon(pepe)} \Rightarrow \neg \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)})$$

~~$$\uparrow: \neg \text{EsDormilon(pepe)} \vee \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)})$$~~  
~~$$\uparrow: \text{EsDormilon(pepe)} \wedge \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)})$$~~

$$\neg \uparrow: \neg (\text{EsDormilon(pepe)} \Rightarrow \neg \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)}))$$

$$\equiv \neg (\neg \text{EsDormilon(pepe)} \vee \neg \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)}))$$

$$\equiv \text{EsDormilon(pepe)} \wedge \exists z. (\text{Posee(pepe, z)} \wedge \text{Revista(z)})$$

$$\rightarrow \text{EsDormilon(pepe)} \wedge (\text{Posee(pepe, R)} \wedge \text{Revista(R)})$$

$$\equiv \{ \{ \text{EsDormilon(pepe)} \}, \{ \text{Posee(pepe, R)} \}, \{ \text{Revista(R)} \} \}$$

Cláusulas:

$$1: \{ \neg \text{Radio}(x), \text{Ruidoso}(x) \}$$

$$2: \{ \neg \text{Posee}(x, y), \neg \text{Libro}(y), \neg \text{Posee}(x, z), \neg \text{Revista}(z) \}$$

$$3: \{ \neg \text{EsDormilon}(x), \neg \text{Posee}(x, y), \neg \text{Ruidoso}(y) \}$$

$$4: \{ \text{Posee}(pepe, k) \}$$

$$5: \{ \text{Libro}(k), \text{Radio}(k) \}$$

$$6: \{ \text{EsDormilon}(pepe) \}$$

$$7: \{ \text{Posee}(pepe, R) \}$$

$$8: \{ \text{Revista}(R) \}$$

(No era más fácil numerar acá las variables?)

Plan:

Pepe posee un libro o una radio. Como pepe es dormilón, no tiene nada ruidoso. Las radios son ruidosas. Como Entonces pepe tiene un libro y no revistas. ~~pepe tiene un libro~~

3 y 6

$$3: \{ \neg \text{EsDormilon}(x_q), \neg \text{Posee}(x_q, y_q), \neg \text{Ruidoso}(y_q) \}$$

$$6: \{ \text{EsDormilon}(\text{pepe}) \}$$

$$S_9 = \text{mgu}(\{ x_q = \text{pepe} \}) = \{ x_q := \text{pepe} \}$$

$$9: \{ \neg \text{Posee}(\text{pepe}, y_q), \neg \text{Ruidoso}(y_q) \}$$

4 y 9

$$4: \{ \text{Posee}(\text{pepe}, k) \}$$

$$9: \{ \neg \text{Posee}(\text{pepe}, y_q), \neg \text{Ruidoso}(y_q) \}$$

$$S_{10} = \text{mgu}(\{ \text{pepe} = \text{pepe}, y_q = k \}) = \{ y_q := k \}$$

$$10: \{ \neg \text{Ruidoso}(k) \}$$

1 y 10

$$1: \{ \neg \text{Radio}(x_{11}), \text{Ruidoso}(x_{11}) \}$$

$$10: \{ \neg \text{Ruidoso}(k) \}$$

$$S_{11} = \text{mgu}(\{ k = x_{11} \}) = \{ x_{11} := k \}$$

$$11: \{ \neg \text{Radio}(k) \}$$

5 y 11 (Me quedo sin tiempo, no reescribo las fórmulas)

$$S_{12} = \text{mgu}(\{ k = k \}) = \{ \}$$

$$12: \{ \text{Libro}(k) \}$$

12 y 2: Falta la sustitución

$$13: \{ \neg \text{Posee}(x, k), \neg \text{Posee}(x, z), \neg \text{Revista}(z) \}$$

~~13 y 2~~

4 y 13: sustitución?

14:  $\{ \neg \text{Posee}(\text{pepe}, z), \neg \text{Revista}(z) \}$  ✓

7 y 14: Mgu:  $\{ z := R \}$

15:  $\{ \neg \text{Revista}(R) \}$  ✓

8 y 15:

16:  $\emptyset$  ✓

$\neg \neg \text{entonces } \vdash \text{f}$

No es resolución SLD porque las cláusulas no son todas de Horn (en particular la 5). ✓

tieneMateriaAprobada(E) :-

estudiante(E),

notas(NS),

between(4, 10, N),

member((E, M, N), NS).

OK, pero no era más eficiente hacer:

member((E, M, N), NS), N >= 4. ?

En vez del between.

NS tiene todas las notas. El estudiante puede reprobar varias veces con la misma nota, pero si aprobo, tiene una única nota para la materia M en el rango [4, 10]. Por eso este predicado retorna true a lo sumo una única vez.

eliminarAplazos([I, I]).

eliminarAplazos([(E, M, N) | NS], [(E, M, N) | LS]) :-

N >= 4,

eliminarAplazos(NS, LS). ✓

eliminarAplazos([(E, M, N) | NS], [(E, M, N) | LS]) :-

N < 4,

not(tieneMateriaAprobada(E, M)),

eliminarAplazos(NS, LS). ✓

eliminarAplazos([(E, M, N) | NS], LS) :-

N < 4,

tieneMateriaAprobada(E, M),

eliminarAplazos(NS, LS). ✓

Asumo que tieneMateriaAprobada revisa la misma lista de notas que recibe eliminarAplazos.

promedio(A, P) :-

notas(NS1),

eliminarAplazos(NS1, NS2),

estudiante(A),

~~notasDe(A, NS2),~~

notasDe(A, NS2, ANS), ✓

sumlist(ANS, S),

length(S, N),

P is S / N.

Asumo que el alumno tiene al menos 1 nota. ✓

⚡ notasDe(+A, +NS2, -ANS)

notasDe(\_, [], []).

notasDe(A, [(A, -, N) | NS], [N] | LS) :- notasDe(A, NS, LS).

notasDe(A, [(B, -, -) | NS], LS) :-

A \= B,

notasDe(A, NS, LS). ✓

MejorEstudiante(A) :-

estudiante(A),

promedio(A, P),

not((estudiante(B),

promedio(B, P2),

P2 > P

)).

Asumo que estudiante/1 instancia todos los estudiantes.

ok, pero por cómo hiciste promedio (reversible en A), no necesitabas instanciar el A :)

✓



EJ3 a) #26

JONATHAN BEKENSTEIN 348/11

2024-07-05

Mensaje	Receptor	Colaboradores	Implementado en	Resultado
new	A		Object	unA ✓
new	B		object	unB ✓
new	C		Object	unC ✓
a:b:	unA	unB, unC	A	3 ✓
c	unC		C	bloque ✓
a:b:	unB	bloque, unA	B	3 ✓
value	bloque		BlockClosure	1 ✓
c	unC		B	1 ✓
a:b:	unC	1, unC	C	1 ✓
c	unA		A	2
+	z	1	SmallInteger	3 ✓

bloque: [self a: super c b: self]

Resultado de (A new) a: (B new) b: (C new) es 3.

SmallInteger << divisors

| res |

res := OrderedCollection new.

1 to: self do: [:d | ((self // d) = 0)

ifTrue: [res add: d] ]

^ res



E13 b) #26

JONATHAN BEKENSTEIN 348/11

2024-07-05

$$\begin{array}{c}
 \frac{}{\Delta \vdash P(x,z) \wedge P(z,y)} \text{ax} \\
 \frac{}{\Delta \vdash P(x,z) \quad \{w=z\}} \text{ax} \\
 \frac{}{\Gamma \vdash \exists z (P(x,z) \wedge P(z,y))} \text{ax} \quad \frac{}{\Delta: \Gamma, (P(x,z) \wedge P(z,y)) \vdash \exists w.P(x,w)} \text{ax} \\
 \frac{}{\Gamma: \exists z (P(x,z) \wedge P(z,y)) \vdash \exists w.P(x,w)} \text{ax} \\
 \frac{}{\vdash \exists z (P(x,z) \wedge P(z,y)) \Rightarrow \exists w.P(x,w)} \text{ax} \\
 \frac{}{\vdash \forall y (\exists z (P(x,z) \wedge P(z,y)) \Rightarrow \exists w.P(x,w))} \forall_i \\
 \frac{}{\vdash \forall x \forall y (\exists z (P(x,z) \wedge P(z,y)) \Rightarrow \exists w.P(x,w))} \forall_i
 \end{array}$$

