Examen de Prácticas - 25 de enero de 2019
LTP (Tipo A)

ALUMNO: _____ GRUPO: _____

Instrucciones

- El alumno dispone de 60 minutos para resolver el examen.
- El examen consta de 3 preguntas que deberán responderse en el mismo enunciado, en los recuadros incluidos en cada pregunta.

Pregunta 1 – Haskell (3.25 puntos)

Define una función `select` cuyo tipo sea:

```
select :: (Int -> Bool) -> [Int] -> [Int]
```

La función `select`, dadas una función de `Int` a `Bool` y una lista de enteros, devuelve una lista de enteros que contiene los enteros de la lista segundo argumento para los que la función primer argumento devuelve `True`.

Requisitos: NO se permite usar ninguna función predefinida de Haskell. Debe resolverse mediante recursión o lista intensional.

Ejemplos de uso:

```
*Main> select even [1..10]
[2, 4, 6, 8, 10]
*Main> select (>6) [1..10]
[7, 8, 9, 10]
```

SOLUCIÓN CON RECURSIÓN DIRECTA:

```
select :: (Int -> Bool) -> [Int] -> [Int]
select f [] = []
select f (x:xs)
  | f x = x : select f xs
  | otherwise = select f xs
```

SOLUCIÓN CON LISTAS INTENSIONALES:

```
select :: (Int -> Bool) -> [Int] -> [Int]
select f xs = [y | y <- xs, f y]
```

Pregunta 2 – Haskell (3.25 puntos)

Considera el siguiente código, donde se define el tipo de datos `Cube` (que representa la figura geométrica tridimensional Cubo) y la clase de tipos `Shape3D` (que representa las figuras geométricas tridimensionales, en general):

```
type Side = Float
data Cube = Cube Side
class Shape3D a where
   area   :: a -> Float
   volume :: a -> Float
```

Se pide añadir el código necesario para que `Cube` instancie `Shape3D`.

Ayuda: Las fórmulas para calcular el área y el volumen de un cubo de lado `x` son:

$$area = 6\ x^2 \qquad\qquad volume = x^3$$

Ejemplos de uso:

```
*Main> volume (Cube 3)
27.0
*Main> area (Cube 3)
54.0
```

SOLUCIÓN:

```
instance Shape3D Cube where
  area (Cube s) = 6 * s**2
  volume (Cube s) = s**3
```

Pregunta 3 – Prolog (3.50 puntos)

Resolver los 2 ejercicios que se plantean, dada la siguiente base de conocimiento:

```
/* movie(M, Y), M is a movie released in the year Y */
movie(barton_fink, 1991).
movie(the_big_lebowski, 1998).
movie(fargo, 1996).
movie(crimewave, 1985).
movie(spies_like_us, 1985).
/* director(M, D), M is a movie directed by D */
director(barton_fink, ethan_coen).
director(the_big_lebowski, joel_coen).
director(fargo, joel_coen).
director(crimewave, sam_raimi).
director(spies_like_us, john_landis).
/* actor(M, A, R), the actor A played the role of R in the movie M */
actor(barton_fink, john_turturro, barton_fink).
actor(barton_fink, john_goodman, charlie_meadows).
actor(the_big_lebowski, jeff_bridges, jeffrey_lebowski__the_dude).
actor(the_big_lebowski, john_goodman, walter_sobchak).
actor(the_big_lebowski, john_turturro, jesus_quintana).
actor(crimewave, joel_coen, reporter_at_execution).
actor(spies_like_us, joel_coen, drive_in_security).
actor(spies_like_us, sam_raimi, drive_in_security).
```

1.- Define un predicado `tandem` que permita encontrar los directores de películas en las que haya salido un actor dado. Ejemplo de uso:

```
?- tandem(D, jeff_bridges).
D = joel_coen.
```

SOLUCIÓN:

```
tandem(D, A) :- director(M, D), actor(M, A, _).
```

2.- Define un predicado `releasedBetween` que permita encontrar las películas producidas entre dos años dados (incluyéndolos). Ejemplo de uso:

```
?- releasedBetween(M, 1996, 1998).
M = the_big_lebowski ;
M = fargo.
```

SOLUCIÓN:

```
releasedBetween(M, Y1, Y2) :- movie(M, A), A >= Y1, A =< Y2.
```