Paradigmas de Programación – UTN – FRBA
2do cuatrimestre 2009
Paradigma Lógico – Parcial

Aclaraciones:

- *Esta evaluación es a libro abierto, pueden usar todo lo que tengan en la carpeta y los apuntes que deseen.*
- *Es muy importante poner nombre, nro. de legajo, nro. de hoja y cantidad total de hojas en cada hoja.*
- *Recuerden que la intención es medir cuánto se sabe del paradigma en cuestión; si un ejercicio es resuelto sin utilizar las ideas del paradigma no será considerado.*
- *Es importante evitar la repetición de código, tanto dentro de un ejercicio como entre los ejercicios. Es decir que se espera que en cada ejercicio reutilicen lo que hayan hecho en los ejercicios anteriores (incluyendo por supuesto los predicados auxiliares).*
- *Para demostrar un correcto conocimiento de los conceptos, es apropiado utilizar la generación sólo en los lugares en donde es necesaria, evitando "generar por generar".*

Tenemos tres tipos de usuario, representados con functores:

- Los usuarios anónimos se identifican por su IP: `anonimo(ipComoListaDe4Enteros)`.
- Registrándose se puede obtener un usuario por un mes *trial*, que se representa por su id de usuario y su fecha de registración: `trial(id, fechaRegistracion))`.
- Para tener un usuario con menos limitaciones existe la posibilidad de pagar por un usuario premium. Los usuarios premium tienen una cuota mensual (en MB) de downloads, que dependerá de cuánto paguen. Se representan por su id de usuario y la cuota mensual correspondiente: `premium(id, cuota)`.

El predicadio `usuario/2` indica los usuarios existentes y les asigna un identificador único, por ejemplo:

```
usuario(1, anonimo([168,145,32,12])).
usuario(2, trial(guille, fecha(10, 10, 2009)).
usuario(3, trial(luis, fecha(29, 9, 2009)).
usuario(4, trial(carlos, fecha(10, 9, 2009)).
usuario(5, premium(martin, 1000)).
```

Ejercicio 1

Información de los usuarios:

- El predicado `estaVigente/1` determina si una cuenta de usuario *trial* se creó hace menos de un mes. Por ejemplo:

```
?- estaVigente(guille).
Yes
?- estaVigente(luis).
Yes
?- estaVigente(carlos).
No
```

Para eso contamos con el predicado `hoy/1` permite conocer la fecha de hoy.

```
hoy(fecha(15,10,2009)).
```

Solución posible:

```
estaVigente(Cuenta):- usuario(_,trial(Cuenta, FechaAlta)), pasoMenosDeUnMes(FechaAlta)).

pasoMenosDeUnMes(fecha(_, M,A)):- hoy(fecha(_,M,A)).
pasoMenosDeUnMes(fecha(D, M,A)):- hoy(fecha(DHoy,MHoy,A)), M is MHoy - 1, D > DHoy.
pasoMenosDeUnMes(fecha(D,12,A)):- hoy(fecha(DHoy,1,AHoy), A is AHoy - 1, D > DHoy
```

- El predicado `estaDisponible` indica si un nombre de usuario está disponible, es decir, no fue tomado aún, por ningún usuario premium ni trial. Por ejemplo:

```
?- estaDisponible(guille).
No. \% Hay un usuario trial con ese nombre
?- estaDisponible(martin).
No. \% Hay un usuario premium con ese nombre
?- estaDisponible(nico).
Yes.
```

Solución posible:

```
estaDisponible(Nombre):- not(existeUsuario(Nombre)).

existeUsuario(Nombre):-usuario(_,trial(Nombre,_)), estaVigente(Nombre).
existeUsuario(Nombre):-usuario(_,premium(Nombre,_)).
```

Ejercicio 2

En la base de conocimientos también se tiene información de los archivos disponibles para bajar, indicando nombre, tamaño en MB y el nombre del servidor en que se ubica. Por ejemplo:

```
estaSubido(thundercatsEpisodio1, 250, servidor1).
estaSubido(thundercatsEpisodio2, 200, servidor2).
estaSubido(thundercatsEpisodio3, 230, servidor1).
```

También se tiene información de los servidores; de cada servidor se indica su capacidad total y si es normal o premium:

```
capacidad(servidor1, 500, normal)   \% Sí, tiene un disco de 450MB... y bueh.
capacidad(servidor2, 200000, premium).
```

A partir de esa información se desea obtener:

- El archivo más grande en cada servidor servidor y su tamaño.

```
?- archivoMasGrande(S, A, T).
S = sevidor1, A = thundercatsEpisodio1, T = 250.
S = sevidor2, A = thundercatsEpisodio2, T = 200.
```

Solución posible:

```
archivoMásGrande(S,A,Tmax):-
  estaSubido(A,Tmax,S),
  forall(estaSubido(_,T,S), Tmax >= T).
```

- El porcentaje de espacio ocupado en un servidor.

```
?- porcentajeOcupacion(servidor1, Porc).
Porc = 96    \% (250+230)*100/500
```

Solución posible:

```
porcentajeOcupacion(Servidor, Porcentaje):-
  capacidad(Servidor, Capacidad, _),
  findall(T, estaSubido(_,T,Servidor), Ts), sumlist(Ts, Ocup),
  Porcentaje is (Ocup * 100) / Capacidad
```

- Una lista con functores de la forma `espacio(nombreServidor, espacioDisponible)`, uno por cada servidor.

```
?- disponibilidadTotal(Lista).
Lista = [espacio(servidor1, 96), espacio(servidor2, 0.1)].
```

Solución posible:

```
disponibilidadTotal(Disp):- findall(espacio(S,P), porcentajeOcupacion(S,P), Disp).
```

Ejercicio 3

En la base de conocimientos también se tiene información de los downloads realizados por cada usuario, indicando id, archivo que se bajó y fecha. Por ejemplo:

```
download(2, thundercatsEpisodio1, fecha(12,10,2009)).
download(5, thundercatsEpisodio1, fecha(12,10,2009)).
download(1, thundercatsEpisodio2, fecha(13,9,2009)).
download(4, thundercatsEpisodio2, fecha(13,9,2009)).
download(5, thundercatsEpisodio2, fecha(13,10,2009)).
```

A partir de esa información se desea obtener:

- El archivo con más downloads históricos.

```
?- masDownloads(Archivo, Cantidad).
Archivo=thundercatsEpisodio2, Cantidad=3.
```

Solución posible:

```
masDownloads(Archivo, CantMax):-
  cantDownloads(Archivo, CantMax),
  forall(cantDownloads(_, OtraCant), CantMax >= OtraCant).

cantDownloads(Archivo, Cant):-
  estaSubido(Archivo, _, _),
  findall(_, download(_, Archivo, _), Downloads),
  length(Downloads, Cant).
```

- El predicado `sePuedeBorrar/2` permite buscar archivos que nunca hayan sido bajados y que se encuentren en un servidor con ocupación mayor al 90 %

```
?- sePuedeBorrar(Archivo, Servidor, Ocupacion).
Archivo=thundercatsEpisodio3, Servidor=1, Ocupacion=96.
```

Solución posible:

```
sePuedeBorrar(Archivo, Servidor, Ocupacion):-
  estaSubido(Archivo, Tam, Servidor),
  not(download(_ , Archivo, _),
  porcentajeOcupacion(Servidor, Ocupacion),
  Ocupacion > 90.
```

- El volumen de downloads (en MB) de un usuario en el mes en curso.

```
?- volumenDownloads(5, Volumen).
Volumen = 450.
```

Solución posible:

```
volumenDownloads(Usuario, Volumen):-
  \% Se podría generar pero no se pide inversibilidad en este caso,
  \% En caso de hacerlo se agregaría:
  \%    usuario(Usuario, _),
  hoy(fecha(_,M,A)),
  findall(Tam, (download(Usuario, Archivo, fecha(_,M,A)),
                estaSubido(Archivo, Tam, _)), Tams),
  sumlist(Tams, Volumen).
```

Ejercicio 4

Finalmente se desea controlar si un usuario puede bajar un archivo o no. Para eso se deben contemplar las siguientes reglas:

- Los usuarios anónimos pueden bajar un solo archivo por día (controlar por IP), que no puede superar los 10 MB.
- Los usuarios trial pueden bajar hasta 300 MB. También se debe verificar que la suscripción esté vigente.
- Los usuarios premium pueden bajar según su cuota mensual.

Sólo los usuarios premium pueden bajar de los servidores premium, todos los usuarios pueden bajar de los servidores normales.

```
?- puedeBajar(Usuario, thundercatsEpisodio3).
\% El usuario anónimo no puede porque es muy grande,
\% guille excedería su cuota de 300 MB y carlos tiene su cuenta vencida.
\% Los demás no tienen problemas.
Usuario = trial(luis, fecha(29, 9, 2009);
Usuario = premium(martin, 1000);
No.
```

Debe ser totalmente inversible.

Solución posible:

```
\% Este tiene muchas soluciones posibles, acá va una de todas esas.
\% Nótese la búsqueda de abstracciones para partir el problema en partes más simples,
\% así se logra más claridad, evitando que la solución sea un gran bodoque de condiciones.
puedeBajar(Usuario, Archivo):-
  estaSubido(Archivo, Tamano, Servidor),
  usuario(_, Usuario),
  tienePermisos(Usuario, Servidor),
  cuotaDisponible(Id, Usuario, Disp), Disp > Tamano.

tienePermisos(anonimo(_  ), S):- servidor(_, normal).
tienePermisos(  trial(N,_), S):- servidor(_, normal), estaVigente(N).
tienePermisos(premium(_,_), S):- servidor(_, _).

cuotaDisponible(Id, anonimo(_), 10):-
  \% Los anónimos pueden bajar 10MB pero sólo una vez por día
  \% Entonces puede bajar 10MB si aún no hizo un download hoy.
  hoy(Hoy),
  not(download(Id, _, Hoy).

cuotaDisponible(Id, trial(_,_), Disponible):-
  \% Para los trial no se puede usar 3c porque ese filtraría los de este mes
  \% y acá se deben considerar todos.
  findall(Tam, (download(Id,Arch,_), estaSubido(Arch,Tam,_)), Tams),
  sumlist(Tams, BajadoHastaAhora),
  Disponible is 300 - BajadoHastaAhora.

cuotaDisponible(Id, premium(_,CuotaTotal), Disponible):-
  \% Acá sí puedo usar 3c.
  volumenDownload(Id, BajadoHastaAhora),
  Disponible is CuotaTotal - BajadoHastaAhora.
```