1- Nos piden modelar una aplicación para administrar una cátedra, usando el paradigma de objetos. En cierta parte del código proponemos lo siguiente:

```
Clase Curso
vi: alumnos, alumnosAprobados
...
alumnosAprobados
^ alumnosAprobados

avisarAprobados
   | mailer |
  mailer := EnviadorMail new.
  self alumnosAprobados do: [ :alumno |
    mailer mandarMailA: alumno.
  ]

actualizarAlumnos
   alumnosAprobados := alumnos select:
[:alumno| alumno aproboParciales ]
...
```

```
Clase Alumno
vi: parciales

aproboParciales
 ^parciales allSatisfy: [:p |
     p nota >= 4 ]

agregarParcial: unParcial
   parciales add: unParcial.
...
```

a) Cuando el líder de desarrollo vio el código dijo: "Esto va a romperse todo el tiempo". ¿Qué lo llevó a hacer esa afirmación, qué inconvenientes presenta el código? ¿Con qué concepto/s se relaciona?
b) Indique qué ocurre en estos workspaces:

```
k1001 := Curso new.
jorge := Alumno new.
...
curso addAlumno: jorge.
...
curso avisarAprobados.
curso actualizarAlumnos.
```

```
k1001 := Curso new.
jorge := Alumno new.
...
curso addAlumno: jorge.
...
curso actualizarAlumnos
curso avisarAprobados.
```

c) Sugiera cambios al código para evitar los problemas encontrados.
d) El enviador de mails funciona así:

```
Clase EnviadorDeMail

enviarMailA: unDestinatario
  |mail |
   mail:= Mail new.
   unDestinatario completaCamposDe: mail.
   self despachar: mail.
```

Relacione el código de arriba con el concepto de polimorfismo. Ejemplifique con otros usos del enviador de mails.

2 - Sabemos que la intersección de dos conjuntos A y B es: A ∩ B = { x / x ∈ A y X ∈ B }. Tenemos las siguientes definiciones en Prolog y Haskell:

```
interseccion([] , _, []).
interseccion([X|L1], L2, [X|L3]):-
  member(X, L2),
  interseccion(L1, L2, L3).
interseccion([X|L1], L2, L3):-
    not(member(X, L2)), [**]
    interseccion(L1, L2, L3).
```

```
interseccion [] _ = []
interseccion (x:xs) ys
   |elem x ys = x : interseccion xs ys
   | otherwise = interseccion xs ys
```

a) En el código Prolog, ¿qué ocurre si eliminamos la línea marcada con [**]? Dé ejemplos de invocación y relacione con conceptos vistos en la materia.
b) ¿Qué ventaja presenta implementar esta solución en paradigma Lógico sobre Funcional?
c) Ahora tenemos el siguiente código:

```
interseccion(L1,L2,L3):-
  findall( X ,
   (member(X, L1),member(Xl, L2)),
   L3).
```

```
interseccion xs ys = [ x |
   x <- xs, any (==x) ys]
```

Compare la implementación nueva y la anterior (entre versiones del mismo lenguaje, no entre ellos) haciendo hincapié en el concepto de declaratividad. Justifique por qué una solución es más declarativa que la otra. Señale los conceptos que aparecen aplicados en las nuevas implementaciones.

d) Si tuviera que implementar este código en Smalltalk, ¿cuál sería el mensaje y quién el objeto receptor? De un ejemplo de implementación del método (si puede, más de uno), y nuevamente relacione con el concepto de declarativdad.

3- Una persona registra las cosas más importantes de su vida en un cofre de recuerdos.

- Las fotos atrás tienen las personas que están (de izquierda a derecha), el lugar y la fecha.
- Los boletos capicúas tienen la fecha, la línea de colectivo y alguna anotación del estilo: "Laura te amo" , "Tigre a la 'A'" o "La amistad es no tener que pedir perdón"
- Las cartas de amigos y novios/novias tienen la persona que lo escribió, la fecha y el texto escrito.

a) ¿Cómo representaría un recuerdo en cada uno de los lenguajes correspondientes a los diferentes paradigmas?
b) ¿Cómo representaría a una persona y su conjunto de recuerdos en Prolog?
c) Teniendo en cuenta la representación descripta en el punto a), ¿es posible tener una lista de todos los recuerdos de una persona en Haskell? Indique qué concepto/s se relacionan con su respuesta.

Se necesita conocer la cantidad de recuerdos valiosos que tiene una persona, sabiendo que:

- Una foto es valiosa si hay más de 5 personas
- Un boleto capicúa del 60 es valioso
- Una carta de más de 50 líneas es valiosa

d) Resuelva el requerimiento en Smalltalk
e) Resuelva el requerimiento en Prolog
f) De los siguientes conceptos, indique cuáles aplican específicamente en las soluciones propuestas en los puntos d) y e); y justifique su elección: Aplicación Parcial, Chequeo de Tipos, Functores, Herencia, Inversibilidad, Motor, Orden Superior, Polimorfismo, Recursividad, Tuplas, Universo Cerrado.