Se cuenta con la historia clínica de los pacientes de un dentista, a través de la siguiente lista:

```
pacientes = [("karl",(10,10,1993), "OSDE", ["Conducto", "Puente"]),
             ("achi",(18,11,1987), "Swiss med", []),
             ("mike",(12,04,1978), "OSDE", ["Limpieza"])]
```

Cada tupla representa:

- El nombre del paciente (que no puede repetirse con otro, cada nombre es único)
- Una tupla día / mes / año que modela la fecha de nacimiento
- Qué obra social tiene
- La lista de servicios o tratamientos que el dentista le practicó

Por otra parte se tiene el monto de cada servicio por obra social:

```
precios =
   [("OSDE", "Conducto", 100), ("OSDE", "Puente", 50),
    ("OSDE", "Limpieza", 60), ("Swiss medical", "Conducto", 30),
    ("Swiss med", "Puente", 50), ("Swiss med", "Limpieza",60)]
```

El formato que sigue la tupla es:

- Nombre de la obra social
- Servicio
- Valor que la obra social le paga al dentista

Se cuenta con estas funciones:

```
servicios (_, _, _, s) = s          obraSocial (_, _, os, _) = os
paciente (p, _, _, _) = p           fechaNacimiento (_, f, _, _) = f
anio (_, _, a) = a
find criterio = head . filter criterio
```

Codificar las funciones indicadas a continuación. En la resolución tenga presente que deben aparecer aplicados, al menos una vez, los siguientes conceptos:

- Aplicación parcial
- Composición
- Funciones de orden superior
- Listas por comprensión

No se puede usar recursividad, a menos que esté indicado en el punto.

Se pide

1) a) Encontrar un paciente en la lista por el nombre

```
>datosDe "karl"
("karl",(10,10,1993), "OSDE", ["Conducto", "Puente"])
```

b) Conocer el monto de facturación de un paciente, o sea, lo que el dentista le facturó a la obra social por los servicios / tratamientos que le dio:

```
>montoFacturacionDe "karl"
150 (100 del conducto + 50 del puente según paga OSDE)
```

Consejo: usar las funciones definidas en 1.a) y la que se muestra a continuación

```
cuantoCuesta obraSocial servicio ((obraSocial1, servicio1, costo)
:servicios) | obraSocial == obraSocial1 && servicio == servicio1 = costo
            | otherwise = cuantoCuesta obraSocial servicio servicios
```

2) Saber si la lista de pacientes está ordenada en forma creciente por monto de facturación:

```
>pacientesOrdenadosPorFacturacion
False (con Karl facturó 150 pesos, pero con Achi 0…)
```

En este punto sí se puede usar recursividad (y sólo en éste).

3) Queremos saber cuáles son los pacientes que cumplen un determinado criterio, sin repetidos. Tenemos esta función.

```
quienesCumplen unCriterio = (map paciente . filter unCriterio) pacientes
```

Utilizar la función para resolver:

- Qué pacientes nacieron en el año 1993
- Con qué pacientes el dentista facturó más de 100 pesos
- Qué pacientes no tuvieron tratamiento/servicio

Nota: no se pueden definir funciones auxiliares

4) Queremos conocer quién es mejor paciente según un criterio dado sobre la tupla paciente.

```
>mejorPaciente "karl" "achi" (length . servicios)
"karl"
```

(porque tiene más servicios, siempre considerar que el mejor es el que tiene más que el otro, en caso de igualdad considerar cualquiera)

5) a) Definir cuánto facturó el dentista a la fecha (considerar todos los servicios practicados a los pacientes)

```
>recaudacionTotal
 210 (150 de "karl" + 60 de "mike")
```

b) Agregar un parámetro que permita definir cuánto facturó el dentista a pacientes que cumplan todos los criterios que le pase como parámetro. Los criterios son funciones que aplican sobre una tupla paciente.

```
>recaudacionTotal [lista de criterios que aplican sobre paciente]
```

c) Muestre cómo resolver cuánto facturó a pacientes de OSDE que nacieron después del año 1992" con la función `recaudacionTotal` del punto 5.b.

6) Inferir los tipos de la función funcionHeavy

```
funcionHeavy a b (c:cs) | a > c      = funcionHeavy a b cs
                        | otherwise  = b c
```