Ejercicios Paradigma Funcional

1. Definir la función **esMultiploDeAlguno**/2, que dado un número y una lista de números, devuelve True si el número es múltiplo de alguno de los números de la lista.
Main> esMultiploDeAlguno 15 [2,3,4]
True

1.a) Resolver la función find' que encuentra el primer elemento que cumple una condición. No se puede resolver con recursividad. Si ningún elemento cumple la condición dejar que falle.

find' :: (a **-> Bool**) **->** [a] **->** a

*__Main__> find' even [**41**..**339**]
42

1.b) Aprovechar la función find' para aplicarla a este dominio.
data Politico = Politico {proyectosPresentados **::** [**String**], sueldo **:: Float**, edad **:: Int** } **deriving Show**

politicos = [**Politico** ["ser libres", "libre estacionamiento coches politicos", "ley no fumar", "ley 19182"] **20000 81**, **Politico** ["tratar de reconquistar luchas sociales"] **10000 63**, **Politico** ["tolerancia 100 para delitos"] **15500 49**]

Queremos encontrar:
a) un político joven (menos de 50 años)
b) alguno que haya presentado más de 3 proyectos
c) alguno que haya presentado algún proyecto que tenga más de 3 palabras (*)
No hay que generar funciones, sino aprovechar find' y desde la consola resuelva los tres requerimientos.

(*) Hay una función que puede venir bien que es: words :: String -> [String]

3. Definir la función **promediosAlumnos**/1, que dada una lista de alumnos devuelve una lista de tuplas que tenga el alumno y el promedio (Consideramos la división entera para el promedio y usamos la función div).
type Nombre = String
type Notas = [Int]
data Persona = Alumno {nombre :: Nombre, notas :: Notas}
Main> **promediosAlumnos**[(Alumno "juan" [8,6]), (Alumno "maria" [7,9,4]), (Alumno "ana" [6,2,4])]
[("juan",7),("maria",6),("ana",4)]

4. Definir la función **promediosSinAplazos**/1, que dada una lista de listas, devuelve la lista de los promedios de cada lista-elemento, excluyendo los que sean menores a 6 que no se cuentan.
Main> **promediosSinAplazos** [[8,6],[6,6,4]]
[7,6]

5. Definir la función **aprobó**/1, que dado un alumno devuelve True si el alumno aprobó. *Aclaración: se dice que un alumno aprobó si todas sus notas son 6 o más*.
Main> **aprobo** (Alumno "manuel" [8,6,2,4])
False

6. Definir la función **aprobaron**/1, que dada una lista de alumnos, devuelve los nombres de los alumnos que aprobaron.
Main>**aprobaron** [Alumno "manuel" [8,6,2,4] , Alumno "elena" [7,9,4,5], Alumno "ana" [6:,2,4,2], Alumno "pedro" [9,6,7,10]]
["elena", "pedro"]

7. Definir la función **productos** que dado una lista de nombres de productos y una lista de precios, devuelve una lista de tuplas.
Main> **productos** ["melon", "zapallo,", "palta"] [15, 10, 12, 7]
[("melon", 15), ("zapallo", 10), ("palta", 12)]
Definirla usando zip y usando zipWith