Instituto técnico Renault

Exámen - Programación II

Ejercicio 1

Intersección y listas

Implementar una función intersect que reciba por parámetros dos listas de números enteros. Luego se solicita retornar una nueva lista que contenga sólo los elementos que se encuentren presentes en ambas listas sin duplicados.

Ejemplos:

```
intersect([1,2,3,4,5],[3,4,5,6,7]) => [3,4,5]
```

Ejercicio 2

Filtrando valores

Solicitar al usuario a través de la consola del sistema que ingrese la cantidad de sublistas que se desean crear. Luego se debe solicitar los números para una determinada sublista culminando cuando el usuario ingrese el valor especial -1.

Luego de que se tenga creada la lista principal con sus sublistas se solicita implementar una función uniqueValues que tome una lista de listas como entrada y devuelva una lista que contenga sólo los elementos únicos presentes en todas las sublistas.

Ejemplos:

uniqueValues([1,2,3],[2,3,4],[3,4,5]) => [2,3]

Aclaración: Los elementos únicos presentes en todas las sublistas serían 2 y 3, ya que estos son los únicos elementos que aparecen en todas las sublistas.

Ejercicio 3

Sumando valores

Implementar una función sumValues que reciba por parámetro una matriz de números enteros y luego retorne la suma de aquellos valores presentes en la misma que cumplan con las siguiente condiciones:

- La suma de sus coordenadas x,y dan como resultado un número impar.
- El valor presente en las posición x,y es un número primo.

* Este ejercicio amerita la implementación de una función auxiliar que determine si un número es primo o no (retornando True o False respectivamente).

Importante: Recordar que un número es considerado como primo si es un número mayor natural que 1 que tiene exactamente dos divisores diferentes el 1 y el mismo.