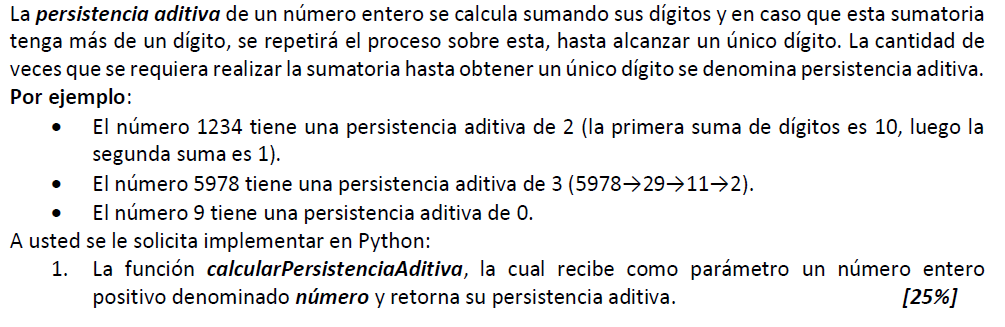
Escriba una función que dado un string verifique si el string está formado solo por letras y que las letras sean minúsculas.



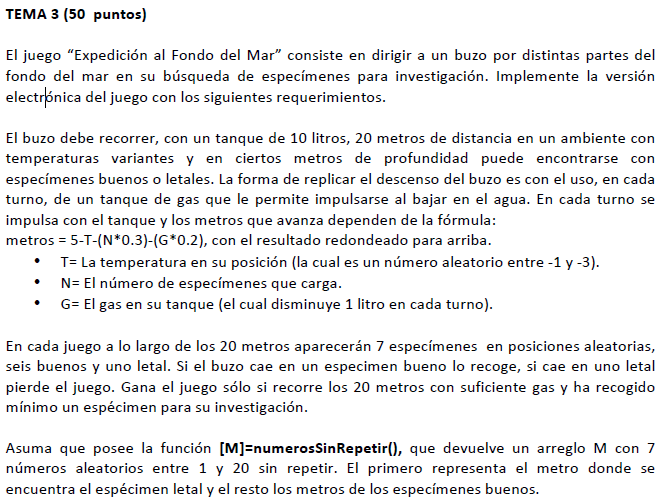
Un número krapekar es todo entero no negativo n que cuando se eleva al cuadrado, el numero resultante puede ser dividido en 2 partes a y b donde a + b = n

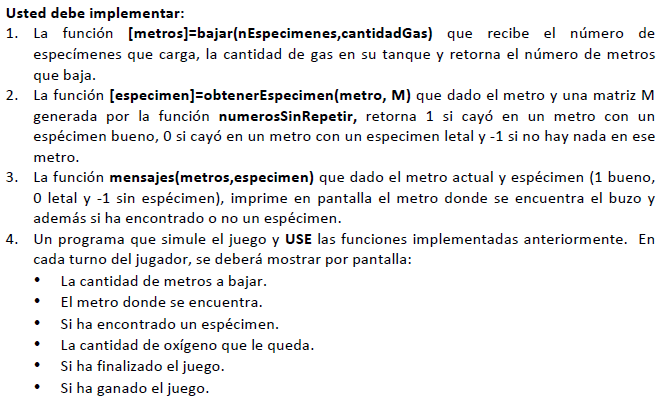
Por ejemplo, 9 y 297 son números Krapekar:

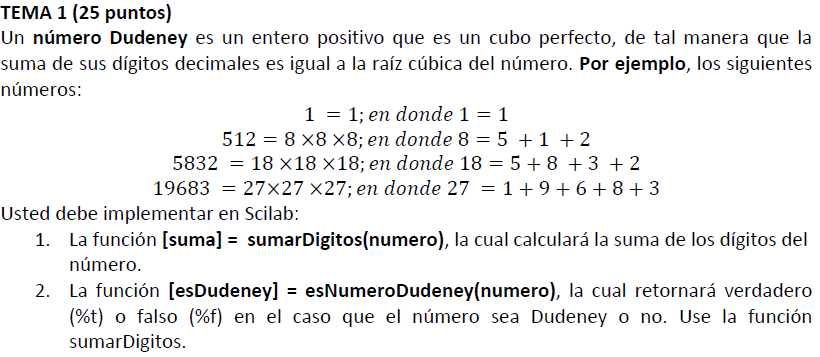
* 9^2 = 81
  + 8+1 = 9 ☺
* 297^2 = 88209
  + 8820 + 9 = 8829 x
  + 882 + 09 = 891 x
  + 88 + 209 = 297 ☺

En cambio 143 no es un número Krapekar:

* 143^2 = 20449
  + 2 + 0449 = 551 x
  + 20 + 449 = 469 x
  + 204 + 49 = 253 x
  + 2044 + 9 = 2053 x







**Tema 3 [45%]**

El acumula-tesoros es un juego de tablero en el que dos jugadores deben atravesar una ciudad avanzando y retrocediendo por un corredor de 50 metros de largo. Mientras atraviesan la ciudad, a cada jugador se le asigna, aleatoriamente, al llegar a cada posición uno de los siguientes ***estados***: fortaleza (-1), hambre (-2) y súper-héroe (-3). Algunos metros pueden tener asignados uno de los siguientes ***elementos***: armas (1), víveres (2) o tesoros (3), elementos que podrán ser recogidos si el jugador está en el estado adecuado:

 Para recoger armas, el jugador debe estar en el estado de fortaleza.

 Para recoger víveres, el jugador debe estar en el estado de hambre.

 Para recoger tesoros, el jugador debe estar en el estado de súper-héroe.

Cada jugador avanza usando un dado de 6 caras y en cada movimiento se verifica:

 Si la ubicación ya ha sido visitada **primero** por el otro jugador, entonces se debe retroceder 10 metros o volver al inicio si estuviera a menos de 10 metros del mismo. En caso que el jugador ***desee*** evitar el retroceso, ***puede evitarlo*** deshaciéndose de un tesoro de los que haya recogido.

 Si la ubicación no ha sido visitada aún, el jugador puede recoger algún elemento dependiendo del estado en el que se encuentre.

Al inicio del juego, usted debe distribuir aleatoriamente 8 armas, 8 víveres y 8 tesoros a lo largo del corredor, ***excepto*** en la posición 1, en donde se ubican ambos jugadores al iniciar la partida. El juego termina cuando uno de los jugadores llega ***exactamente*** al final del corredor. Se declara ganador al jugador que obtenga mayor riqueza, la misma que depende de los elementos recogidos a lo largo del juego y se calcula mediante la siguiente fórmula:

**Riqueza** = (armas/2+1)\*100 + víveres\*0.5 + tesoros

A usted se le solicita implementar en Python:

1. La función ***asignarPosicionElementos*** que retorna una colección que contiene 3 listas, cada una representando las ***posiciones*** en el corredor de las 8 armas, víveres y tesoros, respectivamente.

2. La función ***lanzarDado*** que retorna aleatoriamente el valor de una de las ***caras*** del dado.

3. La función ***generarEstado*** que retorna aleatoriamente uno de los posibles ***estados***.

4. La función ***mostrarElemento*** que dada una ***posición*** y la colección de ***posiciones de los elementos*** imprime en pantalla el tipo de elemento que existe en esa posición.

5. La función ***recogerElemento*** que dada una posición, el estado de un jugador y la colección de ***posiciones de los elementos*** retorna el ***tipo de elemento*** que se ha recogido (1, 2 ó 3) o 0 si no ha sido posible.

6. La función ***mostrarMensaje***, la cual recibe como parámetro un ***jugador***, el ***estado*** del mismo y su ***posición*** para imprimirlos en pantalla.

7. La función ***calcularRiqueza*** que dada una cantidad de ***armas***, ***víveres*** y ***tesoros*** retorna la ***riqueza*** obtenida.

8. Un programa que simule el juego y **use** las funciones implementadas anteriormente.

Al finalizar el juego, se debe declarar al ganador y la riqueza de cada uno.

En el turno de cada jugador, se deberá mostrar por pantalla:

 El jugador al que corresponde el turno actual

 El resultado del lanzamiento del dado y la nueva posición del jugador

 Si hay un retroceso

Un número perfecto, es un número cuya suma de todos sus divisores enteros sin incluir al mismo número, es igual a ese número.

Ej:

*N = 28*

*Suma de divisores de 28: 1+2+4+7+14 = 28*

*28 es número perfecto*

Escriba una función llamada **hallarDivisores(n)** que recibe un entero dado n como ***parámetro*** y retorna una lista que contiene todos los divisores de n.

Luego, escriba una función llamada **esPerfecto(n)** que determine si un número entero dado n enviado como ***parámetro*** es un número perfecto. Utilice dentro de esta función, la función **hallarDivisores(n)**.

Cree una función que elimine los elementos repetidos de una lista.

Cree una función llamada **intersección(A, B)**, que recibe dos listas como parámetros. Y retorna una lista cuyos elementos son la intersección de ambas listas. Es decir, los elementos que aparecen en ambas listas.

Cree una función llamada **unión(A, B)**, que recibe dos listas como parámetros. Y retorna una lista cuyos elementos son la unión de ambas listas.

Cree una función llamada **sonIguales(A, B)**, que recibe dos listas como parámetros. Y retorna True si las dos listas son iguales, caso contrario retorna Falso

Realice una función que dada una URL en String (Ej: “www.espol.edu.ec”), retorne 0 si la URL pertenece a la de una universidad, retorne 1 si la URL pertenece a una universidad de Ecuador y retorna -1 si el URL no es de una universidad.

Nota: Si un URL es de una universidad, tiene .edu

Cree una función que determine en una matriz Numpy cuadrada A si la suma de la Diagonal Principal es igual a la suma de la Diagonal Secundaria.