

### a) Iterador suspenso

```
# 19-00036 por lo tanto, x=0, y=3 , z=6
def suspenso(a, b):
    if b == []:
        yield a
    else:
        yield a + b[0]
        for x in suspenso(b[0], b[1:]):
            yield x

for x in suspenso (9, [0,3,6]):
    print(x)
```

Paso	Pila	a	b	b[0]	b[1:]	valor	imprime
1	suspenso(9,[0,3,6])	9	[0,3,6]	0	[3,6]	yield a + b[0] (9 + 0)	9
2	suspenso(0,[3,6])	0	[3,6]	3	[6]	yield a + b[0] (0 +3)	3
3	suspenso(3,[6])	3	[6]	6	[]	yield a + b[0] (3 + 6)	9
4	suspenso(6,[])	6	[]			yield a	6
5	Vuelve a suspenso(3, [6])					Fin del bucle, vuelve	
6	Vuelve a suspenso(0, [3, 6])					Fin del bucle, vuelve	
7	Vuelve a suspenso(9, [0, 3, 6])					Fin del bucle,	

Caso base: si b es una lista vacía se produce el valor actual en a  
Si b no es vacío se hace la suma de a con el primer elemento de la lista b, se llama recursivamente a suspenso y se coloca al primer elemento de la lista en el lugar de a y el resto de valores en b se convierten en la nueva lista b

Por lo tanto, suspenso con X=0, Y=3, Z=6. Imprime 9, 3, 9 y 6. en ese orden

## b) Iterador misterio

```

13 # 19-00036 por lo tanto, x=0, y=3 , z=6
14 def misterio(n):
15     if n == 0:
16         yield [1]
17     else:
18         for x in misterio(n-1):
19             r = []
20             for y in suspenso(0, x):
21                 r = [*r, y]
22             yield r
23
24 for x in misterio(5):
25     print(x)

```

Llamada	n	misterio(n-1)	Suspenso(0,x)	Lista producida
Misterio(0)	0			[1] (caso base)
Misterio(1)	1	[1]	1,1	[1,1]
Misterio(2)	2	[1,1]	1, 1+1=2,1	[1,2,1]
Misterio(3)	3	[1,2,1]	1,1+2=3,2+1=3,1	[1,3,3,1]
Misterio(4)	4	[1,3,3,1]	1,1+3=4,3+3=6,3+1=4,1	[1,4,6,4,1]
Misterio(5)	5	[1, 4, 6, 4, 1]	1, 1+4=5, 4+6=10, 6+4=10, 4+1=5, 1	[1, 5, 10, 10, 5, 1]

Caso base: si n es 0, entonces entrega una lista que contiene el numero 1  
 En el else Tambien se entra al caso de negativos pero en python no lo acepta dado que se generaria un bucle infinito, asi que para numeros positivos, crea listas de numeros

Aunque el ciclo for en misterio(5) solo ejecuta la aprte recursiva, el resultado final es la lista producida por la ultima llamada recursive, y la salida impresa es: [1, 5, 10, 10, 5, 1]