

**PRUEBA PRÁCTICA I. OCTUBRE 2024.****Pregunta 1**

Para definir la función *number*

1. usa la variable *n\_c* para la cantidad de cifras que tiene el número *a*
2. usa a variable *repe* para cargar el número *aa*. Por ejemplo si *a* es 257, entonces *repe* será 257257
3. usa la variable *pos* para la posición que ocupa *L* en LNIF
4. determina el primeir número mayor o igual que *repe* que dividido ente 23 dá como resto *pos*
5. retorna este número

```
def number(a,L):
```

```
    """
```

```
    sendo a un enteiro positivo, L unha letra maiúscula das que aparecen en LNIF
```

```
    fabrica un número seguindo as intrucións dadas no enunciado anterior
```

```
    """
```

```
    LNIF = 'TRWAGMYFPDXBNJZSQVHLCKE'
```

**TU CÓDIGO**

```
    return n
```

```
# Algunos tests para tu función
```

```
from nose.tools import assert_equal
```

```
assert_equal(number(1,'B'),11)
```

```
assert_equal(number(10,'B'),1023)
```

```
assert_equal(number(10^10,'A'),1000000000010000000003)
```

## Pregunta 2

Define una función que, para un número  $n$  devuelve el segundo en orden creciente de todos los primos mayores que  $n$ .

```
def sprimo(n):
```

```
    """
```

```
    siendo n menor dos primos maiores que n
```

```
    """
```

### TU CÓDIGO

```
    return p
```

### COMPROBACIÓN

```
from nose.tools import assert_equal
```

```
assert_equal(sprimo(1), 3)
```

```
assert_equal(sprimo(11), 17)
```

## Pregunta 3

define a función `public(a, s)` como segue

1. define a variable  $p$  como `sprimo(a)`
2. define a variable  $q$  como `sprimo(number(a,s))`
3. define a variable  $n$  como  $p \cdot q$
4. define a variable  $e$  como 15
5. se NON existe o inverso de  $e$  módulo  $(p-1) \cdot (q-1)$ , proba sucesivamente con 17, 19, 21, ... ata atopar o primeiro valor para o que existe o inverso pedido
6. retorna a lista  $[e, n]$

```
def public(a,s): # modify function name and parameters
```

```
"""
```

A partir del número entero positivo a, y de la letra mayúscula s

define una clave pública siguiendo las especificaciones del enunciado anterior

```
"""
```

```
p= sprimo (a)
```

```
q = sprimo (number(a,s))
```

```
n = p*q
```

```
e = 15
```

```
phi = (p-1)*(q-1)
```

```
n = inverse_mod (e,)
```

```
from nose.tools import assert_equal
```

```
assert_equal(public(1,'B'), [15,51])
```

```
assert_equal(public(10,'B'), [17, 13429])
```

### **Tu clave pública**

1. En las variables dni y letra introduce tu número de dni y la letra del nif en mayúscula, respectivamente.

```
dni =
```

```
letra =
```

```
clave = public(dni,letra)
```

```
clave
```