Departament de Disseny i Programació de Sistemes Electrònics
UNIVERSITAT POLITÈCNICA DE CATALUNYA

# EXERCICI PUNTUABLE INFORMÀTICA

13/11/2017 — Grau en Enginyeria de Sistemes TIC

COGNOMS:

NOM: GRUP de LAB:

**Exercici 1.** Completa els doctests que segueixen.

```
Apartat a)
def mistery(dades):
    """
    >>> mistery([10, 20, 30, 40, 50])
    #apartat a.1)
    >>> mistery([])
    #apartat a.2)
    """
    t=len(dades)
    tmp=dades[t-1]
    for i in range(t):
        if i!=0:
            dades[i]=dades[i-1]
    dades[0]=tmp
    return dades
```

```
Apartat b)
def idontknow(s):
    """
    >>> idontknow('supercalifragilisticoexpialidoso')
    #apartat b.1)
    >>> idontknow('AaA')
    #apartat b.2)
    """
    z=[0]*len(s)
    for j,ch in enumerate(s):
        if ch in z:
            z[z.indexof(ch)]+=1
        else:
            z[j]=1
    return len(z)
```

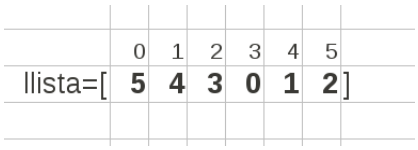
```
Apartat c)
def none(l):
    """
    >>> none(['visualBasic', 'Java', 'Python', 'C'])
    #apartat c.1)
    """
    print [[v[-1],i] for i,v in enumerate(l)]
    print [l[i] for i in range(len(l))][i]
    print "#".join([d[0] for d in l])
```

```
Apartat d)
def npi(m):
    """
    >>> npi([[0, 2, 3], [2, 1, 11], [3, 8, 2]])
    #apartat d.1)
    >>> npi([[0, 2, 3, 4], [2, 1, 11, 5]])
    #apartat d.2)
    >>> npi([[1, 3], [2, 2]])
    #apartat d.3)
    """
    d1=[m[i][i] for i,v in enumerate(m) if i==v[i]]
    print d1
    d2=[m[i][len(m)-1-i] for i in range(len(m))]
    print d2
    return len([d1[i] for i in range(len(d1)) if d1[i]==d2[i]])
```

**Exercici 2. Encriptant frases**. Implementa òptimament la funció *encriptaFrase* tal que, donada una frase separada per espais, retorni el resultat d'encriptar la frase d'acord amb el procediment que segueix: per cada paraula de la frase, s'agafa la darrera lletra de la paraula, i així es va construint una nova paraula encriptada. A continuació segueixen els doctests.

```
def encriptaFrase(frase):
    """
    >>> encriptaFrase('python programming is funny and tricky')
    'ngsydy'
    """
```

**Exercici 3. El rastrejador**. Suposant que tens una llista de mida aleatòria, correctament inicialitzada amb valors corresponents als índexs possibles de la llista. L'objectiu consisteix en rastrejar, des d'un posició donada, anant a l'índex emmagatzemat com a valor a cada posició, fins tornar a la posició 0. Per exemple, suposant la llista que mostra la figura i començant a l'índex 0, caldria rastrejar fins a la posició 5. En aquesta posició hi ha el valor 2. Caldria anar fins a la posició 2, on hi ha el valor 3. Caldria anar a la posició 3, on hi ha el valor 0, i s'acabaria el rastreig perquè hem tornat a la posició 0.




Se us demana que implementeu òptimament la funció *rastrejaPassos* d'acord amb el funcionament anterior, exemplificat a través dels doctest que segueixen. Fixeu-vos que la funció retorna el número de passos que ha calgut fer fins retornar a la posició 0, partint de la posició index. Suposeu també que els valors inclosos a la llista no porten mai a accessos que no permetin arribar a la posició 0.

```
def rastrejaPassos(l,index):
    """
    >>> rastrejaPassos([5,4,3,0,1,2], 0)
    4
    >>> rastrejaPassos([2,4,3,0,1,5], 3)
    1
    >>> rastrejaPassos([1,2,3,0],2)
    2
    >>> rastrejaPassos([], 1)
    Not data
    """
```

**Exercici 4. La matriu matemàtica**. Donada una llista de llistes que representa una matriu matemàtica de qualsevol ordre, se us demana que dissenyeu òptimament i implementeu una funcio pura de nom *canvi* que permeti l'operació d'intercanviar files per columnes, seguint l'ordre que es mostra en el doctest que segueix. Suposeu que la matriu està construïda correctament.

```
def canvi(m):
    """
    returns the resulting matrix of interchanging rows and columns in matrix m
    >>> canvi([[1, -1], [2, -2], [3, -3]])
    [[-1, -2, -3], [1, 2, 3]]
    >>> canvi([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
    [[3, 6, 9], [2, 5, 8], [1, 4, 7]]
    >>> canvi([])
    []
    """
```