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

EXERCICI PUNTUABLE TECPRO

17/05/2016 Grau en Enginyeria de Sistemes TIC

COGNOMS:

NOM: GRUP de LAB:

Exercici 1. Donada les següents funcions recursives, a) Justifiqueu el resultat de cadascun dels seus doctestos b) Pel primer doctest, justifiqueu quantes crides recursives s'efecutaran c) Digueu quin és el cost assimptòtic en cas pitjor de dita funció.

```
Apartat a)
def uala(a,i):
    """
    >>> uala([0,1,2,8,13,17,19,32,42],3)
    #doctest1
    >>> uala([-1,1,8,13,34],1)
    #doctest2
    """
    if len(a)==0:
        return False
    else:
        m=len(a)/2
        if a[m]==i:
            return True
        else:
            if i<a[m]:
                return uala(a[:m],i)
            else:
                return uala(a[m+1:],i)
Apartat b)
def carai(n):
    """
    >>> carai(1)
    #doctest1
    >>> carai(2)
    #doctest2
    >>> carai(3)
    #doctest3
    """
    if n==1:
        return 1
    else:
        return carai(n-1)+n*n
```

Exercici 2. Apartat a) Escriu una funció **recursiva** que obtingui el valor mes petit d'una llista d'elements. Òbviament, no es pot utilitzar la funció *min* de Python.

```
def minim(llista):
    """
    >>> minim([2,34,-1,-4])
    -4
    >>> minim([])
    'no data'
    """
```

Apartat b) Escriu una funció **recursiva**, de nom *ordena*, tal que, donada una llista no ordenada, *laLlista*, retorni la llista resultant d'ordenar-la ascendentment, seguint l'estratègia que es detalla a continuació. L'element més petit anirà a la primera posició de la nova llista, i després cal realitzar l'ordenació amb la resta de la llista, de manera successiva fins que tota ella estigui ordenada. Si ho considereu necessari, podeu utilitzar els mètodes *len* i *remove* sobre llistes (l'ús de qualsevol altre mètode predefinit de Python comportarà puntuació nul·la). Podeu utilitzar la funció *minim* que heu implementat en l'Apartat a).

```
def ordena(laLlista):
    """
    >>> ordena([-4,22,11,0,2,8,99,-2])
    [-4, -2, 0, 2, 8, 11, 22, 99]
    >>> ordena([7, 6, 5, 4, 3, 2, 1])
    [1, 2, 3, 4, 5, 6, 7]
    """
```

Exercici 3. Suposant la implementació de l'estructura de dades lineal pila (Stack), i les seves operacions push(), top(), pop(), len.

Apartat a) Se us demana que implementeu únicament el mètode *reverse(s)*, tals que, donada una pila *s*, permeti girar l'ordre dels elements de *s*. Suposeu que aquest nou mètode s'afegirà a la classe Stack(). Podeu utilitzar una pila addicional, i els mètodes de la classe Stack. Òbviament, l'ús de funcions predefinides de Python comportarà una puntuació nul·la. Comproveu-ne a continuació el seu ús.

```
if __name__=='__main__':
    s=Stack()
    for i in range(10):
        s.push(i)
    t=s.reverse()
    print t # Escriura 9 8 7 6 5 4 3 2 1 0
    s=t.reverse()
    print s # Escriura 0 1 2 3 4 5 6 7 8 9
```

Apartat b) Ara passeu a recursiu aquest mètode i anomeneu-lo *reverseRec(s)*.

Exercici 4. Donada la classe *Arbre*, amb els mètodes implementats a classe de teoria, en què cada node pot tenir més de 2 fills, se us demana que implementeu el mètode **recursiu** *quantesFullesS*, tal que, donat un arbre, retorni el nombre de fulles amb valors senars que hi ha a l'arbre. Per exemple, en l'arbre següent, la resposta hauria de ser 5.