Interrogation d'informatique

Ce nombre est-il une puissance de 2 ?

On souhaite savoir si un entier $n \geq 2$ peut s'écrire comme une puissance de 2, càd savoir s'il existe $p \in \mathbb{N}^*$ tel que $n = 2^p$. On propose le programme suivant :

```
1 def puissance(n):
2     '''n est un entier supérieur ou égal à 2. Cette fonction renvoie
         True si n est une puissance de 2, False sinon.'''
3     while n > 1:
4         if n % 2 == 0:
5             n = n / 2
6         else:
7             return False
8     return True
```

1) Démontrer la terminaison de cet algorithme en précisant un variant de boucle.

2) On note n_k la valeur de n après k itérations de la boucle `while`, la valeur n_0 correspondant à la valeur donnée en argument. Recopier et compléter le tableau suivant (rajouter des lignes le cas échéant).

k	n_k	n_k en base 2
0	24	...
⋮		
⋮		

Faire de même pour $n = 8$. Que constate-t-on ?

3) Étant donné un entier $n \geq 2$, on note q l'entier naturel tel que $2^{q-1} \leq n < 2^q$. Justifier que n s'écrit avec q chiffres en base 2.

4) Exprimer la complexité de la fonction `puissance` en fonction de q.

Recherche de point fixe

Étant donné une liste L et un entier naturel p, on dit que p est un point fixe si `L[p]=p` (à condition que `L[p]` ait un sens). On souhaite construire un algorithme qui recherche un tel point fixe, sachant qu'il n'est pas nécessairement unique. On commence par un algorithme "naïf".

1) Écrire une fonction `pointFixe` qui prend en argument une liste `L` et renvoie l'indice `p` d'un point fixe s'il existe, et dans le cas contraire renvoie `None`.

2) Quelle est la complexité de `pointFixe` ?

À présent, on considère que L est une liste *d'entiers relatifs tous distincts triés par ordre croissant.*

3) Si on constate que `L[k]>k`, que peut-on en déduire sur l'éventuel point fixe ? Même question si `L[k]<k`.

4) S'inspirer de la méthode de dichotomie pour concevoir une fonction `PFtri` qui prend en argument une liste `L` qui correspond aux conditions ci-dessus et qui renvoie l'indice `p` d'un point fixe s'il existe, et dans le cas contraire renvoie `None`.

Tournez la page S.V.P.

Questions diverses

1) On considère l'algorithme suivant :

```
def mystere(S):  # S est une chaine de caractères
    if S=="":
        return ""
    else:
        return S[-1] + mystere(S[0:-1])
```

Que renvoie `mystere("X")` ? et `mystere("abc")` ? Quelle opération réalise finalement la fonction `mystere` ?

2) Expliquer le principe général du tri fusion. Quelle est sa complexité en fonction de la taille de la liste à trier ?

3) Écrire une fonction `sansDoublon` qui à une liste `L` retourne une liste qui contient les valeurs contenues dans `L`, sans doublon.

4) (bonus très difficile, bon courage !!) On considère l'algorithme suivant :

```
def calc(n):
    if n>100:
        return n-9
    else:
        return calc(calc(n+10))
```

Montrer que pour tout $n \in [\![0, 100]\!]$, l'instruction `calc(n)` renvoie 92.