Problème de Noël — Maths-Info

Exercice 1 — (Centrale-Supélec TSI 2019)

La suite de Fibonacci est la suite (F_n) définie en posant :

$$F_0 = 0\,; \qquad F_1 = 1\,; \qquad \forall n \in \mathbb{N}, \quad F_{n+2} = F_{n+1} + F_n$$

1/ Ecrire une fonction Python `fibonacci(p)` qui prend en paramètre un entier naturel p, et renvoie la liste des $p+1$ premiers termes de la suite de Fibonacci. Par exemple `fibonacci(5)` doit renvoyer `[0, 1, 1, 2, 3, 5]`.

2/ Ecrire une fonction Python `recherche(n)` qui prend en paramètre un entier naturel n, et renvoie le plus grand entier naturel p tel que $F_p \leqslant n$. Par exemple, `recherche(7)` doit renvoyer 5.

Un corrigé possible

```
# Question 1

def fibonacci(p):
    u, v =0, 1 #  initialisation
    for k in range(p):
        u, v =v, u +v
    return u


# Question 2

def recherche(n):
    if n ==0:
        return 0
    elif n ==1:
        return 2
    else:
        k =2
        while fibonacci(k) <=n:
            k +=1
        k -=1
        return k
```

Exercice 2 — (CCPINP-MP 2021)

La suite des nombres de Bernoulli notée (b_n) est définie par :

$$b_0 = 1, \qquad \text{et} \qquad \forall\, n \in \mathbb{N}^*,\ b_n = \frac{-1}{n+1} \sum_{k=0}^{n-1} \binom{n+1}{k} b_k$$

Les algorithmes demandés doivent être écrits en langage Python. On sera très attentif à la rédaction du code, notamment à l'indentation.

1/ Ecrire une fonction `facto(n)` qui renvoie la factorielle d'un entier naturel n.

2/ Ecrire une fonction `binom(n,p)` qui reçoit comme paramètres deux entiers naturels n et p, et qui renvoie le coefficient binomial $\binom{n}{p}$.

3/ Ecrire une fonction `berboulli(n)` qui renvoie une valeur approchée du nombre rationnel b_n. On pourra utiliser librement la fonction `binom(n,p)` de la question précédente.

Par exemple `bernoulli(10)` renvoie $0,075\,757\,575\,757\,575\,76$ qui est une valeur approchée de $b_{10} = \dfrac{5}{66}$

Un corrigé possible

```
# Question 1

def facto(n):
    result =1
    for k in range(1,n+1):
        result = result * k
    return result

# Question 2

def binom(n,p):
    return facto(n)//(facto(p)*facto(n-p))

# Question 3

def bernoulli(n):
    B =[1]
    for m in range(1,n+1):
        if m>1 and m%2!=0:
            B.append(0)
        else:
            c =0
            for k in range(0,m):
                c +=binom(m+1,k)*B[k]
            B.append(-c/binom(m+1,m))
    return float(B[n])
```

Exercice 3 — (CCPINP-MP 2022)

M. Toutlemonde habite dans un immeuble dont la porte d'entrée est sécurisée par un code à 4 chiffres dont chacun est compris entre 0 et 9. Malheureusement, il se trouve devant cette porte et il en a oublié le code.

M. Toutlemonde essaye des alors codes au hasard, sans se soucier du fait qu'il les ait déjà essayés ou non.

1/ Compléter, en langage Python, le script suivant pour qu'il simule une personne essayant de deviner le code 4714 :

```
code =4714
n = int(input('Tapez un code à 4 chiffres : '))
k =  ..............
while  ..............
            ..............
            ..............
print('Vous avez trouvé le code en ' + str(k) +' essais .')
```

(..... représente une instruction ou une partie d'instruction à compléter.)

Pour ne plus oublier le code, M. Toutlemonde décide de l'écrire sur un papier qu'il garde dans sa poche. Pour ne pas se faire dérober le code il le crypte de la manière suivante : il remplace chacun des 4 chiffres par lui-même additionné de 5 et réduit modulo 10. Par exemple, le code 4714 est crypté 9269.

2/ Ecrire, en langage Python, une fonction crypte(m) qui reçoit en entrée une liste m de 4 chiffres et renvoie en sortie la version cryptée de cette liste. Par exemple, crypte([4,7,1,4]) renvoie [9,2,6,9].

Un corrigé possible

```
# Question 1

code =4714
n = int(input('Tapez un code à 4 chiffres : '))
k =1
while n !=code:
    n = int(input('Tapez un code à 4 chiffres : '))
    k +=1
print('Vous avez trouvé le code en ' + str(k) +' essais .')

# Question 2

def crypte(m):
    L =[0, 0, 0, 0]
    for k in range(4):
        L[k] =(m[k] +5) % 10
    return L

# Alternative plus rapide pour la question 2

def crypte2(m):
    return [((m[k] +5) % 10) for k in range(4)]
```

Exercice 4 — (CCPINP-MP 2021)

On rappelle qu'un nombre premier est un entier naturel $n \geqslant 2$ qui admet exactement deux diviseurs (1 et lui-même) dans $\mathbb{N}$.

1/ Ecrire une fonction booléenne `estPremier(n)` qui prend en argument un entier naturel non nul n et qui renvoie le booléen `True` si n est premier et le booléen `False` sinon.

On pourra utiliser le critère suivant : un entier $n \geqslant 2$ qui n'est divisible par aucun entier $d \geqslant 2$ tel que $d^2 \leqslant n$ est premier.

2/ En déduire une fonction `LPrem(n)` qui prend en argument un entier naturel non nul n, et renvoie la liste des nombres premiers inférieurs ou égaux à n.

3/ Pour p un nombre premier et n un entier naturel, on appelle valuation p-adique de n, et on note $v_p(n)$ le plus grand entier k tel que p^k divise n. [1]

Ecrire une fonction `Valpadic(n,p)` qui prend en argument un entier naturel n non nul et un nombre premier p, et qui renvoie la valuation p-adique de n.

Un corrigé possible

```
# Question 1

def estPremier(n):
    if n ==1:
        return False
    else :
        Test =True
        d =2
        while Test and d**2 <=n: # "while Test:" est synonyme de "while Test == True:"
            if n % d ==0:
                Test =False
            d +=1
        return Test

# Question 2

def LPrem(n):
    L =[ ]
    for k in range(1, n+1):
        if estPremier(k): # Synonyme de "estPremier(k) == True:"
            L =L +[k]
    return L

# Question 3

def Valpadic(n,p):
    val =0
    while n % (p**val) ==0:
        val +=1
    return val-1
```

1. Par exemple : $v_2(12) = 2$, $v_3(12) = 1$, $v_5(12) = v_7(12) = 0$.