LE THÉORÈME DE ZECKENDORF : CORRECTION

Partie 1 : préliminaires

1. • Pour tout entier naturel $n \geqslant 1$: $F_{n+1} - F_n = F_n + F_{n-1} - F_n = F_{n-1}$.
Or, $F_{n-1} > 0$ donc : $F_{n+1} - F_n > 0$ et $F_{n+1} > F_n$.
• De plus : $F_1 > F_0$.
Conclusion : la suite (F_n) est strictement croissante.

2. (F_n) est une suite strictement croissante d'entier naturels…
Par conséquent, elle ne peut pas être majorée.

Partie 2 : démonstration du théorème de Zeckendorf

1. Existence

$P(n)$: « tout entier naturel inférieur à n s'écrit comme somme de nombres de Fibonacci non consécutifs ».

Initialisation : $n = 1$
Démontrer $P(1)$ revient à démontrer que 1 s'écrit comme somme de nombres de Fibonacci non consécutifs. Or, on a bien : $1 = F_1$.
Donc $P(1)$ est vraie.

Hérédité : soit $n \in \mathbb{N}$. On suppose que $P(n)$ est vraie.
Montrons que $P(n+1)$ est vraie, c'est-à-dire que « tout entier inférieur à $n+1$ s'écrit comme somme de nombres de Fibonacci non consécutifs ».
Soit un entier inférieur à $n+1$, noté k.
• ***1er cas*** : $k \leqslant n$
D'après l'hypothèse de récurrence, k s'écrit comme somme de nombres de Fibonacci non consécutifs.
• ***2ème cas*** : $k > n$
Alors cet entier est nécessairement $n+1$.

- Si $n+1$ est un nombre de Fibonacci, alors $P(n+1)$ est évidemment vraie.
- Si $n+1$ n'est pas un nombre de Fibonacci :

  (F_n) est strictement croissante et non majorée, donc il existe un entier r tel que $F_r < n+1 < F_{r+1}$.
  On pose $k = n+1-F_r$.
  Alors : • $\boxed{k > 0}$
  • $k < F_{r+1} - F_r$ *ie* $k < F_{r-1}$.
  Or : $F_{r-1} < F_r < n+1$ donc $F_{r-1} < n+1$ *ie* $F_{r-1} \leqslant n$.
  Et donc : $\boxed{k < n}$.
  D'après l'hypothèse de récurrence, l'entier k s'écrit donc comme somme de nombres de Fibonacci non consécutifs. Puisque $n+1 = k + F_r$, alors $n+1$ s'écrit bien comme somme de nombres de Fibonacci.
  Sont-ils non consécutifs ? Oui, car $k < F_{r-1}$ et donc tous les termes de la somme sont strictement inférieurs à F_{r-1}.

Donc $P(n+1)$ est vraie.

Conclusion : d'après le raisonnement par récurrence, $P(n)$ est vraie pour tout entier naturel n.

2. Unicité

a) $P(p)$: « si on considère p nombres de Fibonacci non consécutifs (dans l'ordre croissant) notés F_{r_1}, F_{r_2}, … , F_{r_p}, alors leur somme est strictement inférieure à F_{r_p+1} ».

Initialisation : $p=1$

$F_{r_1}<F_{r_1+1}$ car la suite (F_n) est strictement croissante.

Donc $P(1)$ est vraie.

Hérédité : soit $p \in \mathbb{N}^*$. On suppose que $P(p)$ est vraie.

Montrons que $P(p+1)$ est vraie.

On considère $p+1$ nombres de Fibonacci non consécutifs (dans l'ordre croissant) notés F_{r_1}, F_{r_2}, … , F_{r_p}, $F_{r_{p+1}}$.

D'après l'hypothèse de récurrence : $F_{r_1}+F_{r_2}+\ldots+F_{r_p}<F_{r_p+1}$.

D'où : $F_{r_1}+F_{r_2}+\ldots+F_{r_p}+F_{r_{p+1}}<F_{r_p+1}+F_{r_{p+1}}$.

Or, F_{r_p} et $F_{r_{p+1}}$ ne sont pas consécutifs donc le nombre de Fibonacci suivant F_{r_p} est strictement inférieur à F_{r_p+1} : $F_{r_p}<F_{r_p+1}<F_{r_{p+1}}$. Et alors : $F_{r_p+1} \leqslant F_{r_{p+1}-1}$.

On a donc : $F_{r_1}+F_{r_2}+\ldots+F_{r_p}+F_{r_{p+1}}<F_{r_p+1}+F_{r_{p+1}} \leqslant F_{r_{p+1}-1}+F_{r_{p+1}}$. Mais par définition : $F_{r_{p+1}-1}+F_{r_{p+1}}=F_{r_{p+1}+1}$.

Donc $F_{r_1}+F_{r_2}+\ldots+F_{r_p}+F_{r_{p+1}}<F_{r_{p+1}+1}$. CQFD

Conclusion : d'après le raisonnement par récurrence, $P(p)$ est vraie pour tout entier naturel non nul p.

b) $F_{r_p}<F_{s_q}$ donc $F_{r_p+1} \leqslant F_{s_q}$.

D'après le lemme : $n<F_{r_p+1}$.

D'autre part : $F_{s_q} \leqslant n$.

On a donc : $n<F_{r_p+1} \leqslant F_{s_q} \leqslant n$ et donc $n<n$. Ceci est absurde.

Conclusion : un entier naturel n ne peut pas s'écrire de deux façons différentes comme somme de nombres de Fibonacci non consécutifs. Son écriture est donc unique.

Partie 3 : en pratique

```
def liste_fibo_max(max):
    ''' Renvoie tous les nombres de Fibonacci inférieurs à max '''
    list_fibo = [1,2]
    x = 3
    while x <= max:
        list_fibo.append(x)
        x = list_fibo[-1] + list_fibo[-2]
    return list_fibo

def maxi_fibo(nb, l): #l est une liste qui contient des nombres (de Fibonacci) décroissants
    ''' Renvoie le plus grand nombre de Fibonacci inférieur à nb, et son rang '''
    j = l[0]
    compt = 0
    while j > nb:
        compt += 1
        j = l[compt]
    return j, len(l)-1-compt

print(maxi_fibo(56432,liste_fibo_max(56432)))

def zeckendorf(n):
    liste_fibo = liste_fibo_max(n)
    liste_fibo.reverse() #on trie la liste dans l'ordre décroissant
    decompo, decompo_rang = [], [] #listes des termes de la décompo. de Zeckendorf et des
rangs
    while n>0:
        temp_maxi, rang = maxi_fibo(n, liste_fibo)
        decompo.append(temp_maxi)
        decompo_rang.append(rang)
        n = n - temp_maxi
    return decompo, decompo_rang

n = input("Entier non nul n à décomposer en base Fibonacci = ?")
z, z_rang = zeckendorf(int(n))
txt = str(n)+" = "+str(z[0])
del z[0]
for elt in z:
    txt += " + "+str(elt)
txt += "\n= F"+str(z_rang[0])
del z_rang[0]
for elt in z_rang:
    txt += " + F"+str(elt)
print(txt)
```