Correction des exercices
Semaine 2

Cours Turing

A Suite de Fibonacci

Dans cet exercice, il faut calculer une suite définie récursivement. Comme on sait que $f(n) = f(n-1) + f(n-2)$ on peut faire cet appel récursif. Il faut néanmoins faire attention au cas de base, qui est $f(n) = n, \quad n \leq 1$

```python
def fib(k):
    if (k<=1):
        return k
    return fib(k-1)+fib(k-2)

n = int(input())
print(fib(n))
```

B Est-ce une puissance de x ?

Ici on a utilisé le fait que si $x^k = n$ alors $n \ \% \ x = 0$ et donc on peut répéter le même processus pour $x^{k-1} = \frac{n}{x}$ récursivement. Avec comme cas de base $n = 1$ est vrai car $x^0 = 1$ et que si $n \neq 1$ mais $x = 1$ alors c'est faux car $1^k = 1 \neq n$

```python
def solve(x,n):
    if n==1:
        return True
    if x==1 or n%x != 0:
        return False
    return solve(x,n/x)


t = int(input())
for _ in range(t):
    x,n = map(int,input().split())
    print("YES" if solve(x,n) else "NO")
```

C Jeu de Nim II

Ici, bien qu'existante, la solution avec de l'arithmétique modulaire est bien plus difficile à trouver. Comme les limites sont faibles ($n \leq 40$) il est plus simple d'utiliser un processus récursif. Le cas de base est que $n = 1$ est perdant et le cas récursif est que si on peut atteindre une position perdante on gagne (puisqu'on donne cette position à l'adversaire).

```
def nim(n):
    if n<=1:
        return n<1
    ans = nim(n-1) and nim(n-3) and nim(n-4)
    return not ans
n = int(input())
print("ALICE" if nim(n) else "BOB")
```

D Vacances familiales

Cet exercice est un peu plus difficile. Il faut se rendre compte que résoudre le problème revient à le résoudre pour $n-1$ deux fois et bouger la plaque la plus grande entre deux. Cet-à-dire, déplacer un stack de taille $n-1$ au milieu, déplacer le grand disque à droite, et le stack de taille $n-1$ du millieu à la droite. Ce qui donne un code comme suit. On peut également prouver par induction que le nombre de coups minimum est $2^n - 1$ mais il est peut-être plus facile de compter les coups et imprimer le reste après.

```
def hanoi(n,de,a,par):
    if n==1:
        print(de,a)
        return
    hanoi(n-1,de,par,a)
    print(de,a)
    hanoi(n-1,par,a,de)

n = int(input())
print(2**n - 1)
hanoi(n,1,3,2)

```