Labo Maths – Énigme du mois Décembre:
Il faut faire la lumière sur cette affaire

– Démonstration de Hedi Bazazi –
en 1°10

Pour contacter: *hedi.bazazi-e@ert.tn*

- **Consigne:** Nous avons 6 ampoules numérotés de manières croissantes (14, 15, 21, 22, 33, 35) et qui, lorsqu'on actionne l'interrupteur N d'une ampoules, elle change d'état Cependant, celle dont les numéros ont avec N un diviseur commun > à 1, changent elles aussi d'état.
  Ainsi il est possible de les allumer toutes en même temps en actionnant successivement 4 de ses 6 interrupteurs…

  Pour déterminer lesquelles nous allons procédés ainsi :

J'ai ainsi essayé de faire **un code python** qui illustre le problème pour déterminer ces interrupteurs.
De ce fait, on va chercher un nombre entier > à 1 qui divise chacun des N entiers de chaque ampoule, c'est-à-dire un **diviseur commun à chaque ampoule.**

- (par exemple: pour 14 on aura 21, 22 et 35)

dans la foulé, **le code répondra au problème :**

```python
def pgcd(a, b):
    while b:
        a, b = b, a % b
    return a

def chrch_interrupteurs_a_actionner(nbr, nbr_interrupteurs_a_actionner):
    interrupteurs_a_actionner = []  #initialisation de la liste de départ

    for i in range(1, len(nbr) + 1):
        diviseur_commun = [j for j in range(1, len(nbr) + 1) if i % j == 0]

        if sum(nbr[j - 1] % 2 == 0 for j in diviseur_commun) % 2 == 1:
           interrupteurs_a_actionner.append(i)

        if len(interrupteurs_a_actionner) == nbr_interrupteurs_a_actionner:
            break

    return interrupteurs_a_actionner

nbr = [14, 15, 21, 22, 33, 35]
nbr_interrupteurs_a_actionner = 4
interrupteurs = chrch_interrupteurs_a_actionner(nbr, nbr_interrupteurs_a_actionner)
print("Pour allumer toutes les ampoules, actionnez successivement les interrupteurs:", interrupteurs)
```

```
%Run engime.py
our allumer toutes les ampoules, actionnez successivement les interrupteurs: [1, 2, 3, 5]
```

Interpréteur python: Thonny

Explication du code:

1) Ici, La fonction (chrch_interrupteurs_a_actionne) prend comme entrée une liste d'ampoules (nbr) et le nombre d'interrupteurs à actionner (interrupteur_a_actionner).

2) Elle renvoie une liste d'interrupteurs à actionner pour allumer toutes les ampoules. Le code utilise deux boucles for.

3) La première boucle (for i in range(1, len(nbr) + 1)) Vérifie tous les interrupteurs possibles (de 1 à N, où N est le nombre total d'ampoules).

Pour chaque interrupteur i, la liste des diviseurs communs avec i est calculée.

1) La deuxième boucle (for j in diviseur_commun) compte le nombre d'ampoules dont les numéros ont un diviseur commun avec l'interrupteur actuel, et qui sont également éteintes (valeur paire).

- Si le nombre d'ampoules éteintes avec un diviseur commun est impair, cela signifie qu'en actionnant cet interrupteur, l'ensemble des ampoules changera d'état.
- Enfin, si le nombre d'interrupteurs à actionner (len(interrupteurs_a_actionner)) atteint le nombre recherché (nbr_interrupteur_a_actionner), la fonction s'arrête (break) et renvoie la liste des interrupteurs à actionner.

Le programme nous affiche dans la console:

- "Pour allumer toutes les ampoules, actionnez successivement les interrupteurs: [1, 2, 3, 5]"

Par conséquent, d'après le programme il faudra actionner les interrupteurs 1, 2, 3 et 5 successivement pour les allumer toutes en même temps.

- La fonction *(chrch_interrupteurs_a_actionne)* parcourt chaque interrupteur disponible et identifie les ampoules qui changent d'état lorsqu'on actionne l'interrupteur N.
- Ainsi elle choisit les interrupteurs nécessaires pour atteindre l'objectif défini par le nombre demander *(nbr_interrupteur_a_actionner).*
- Ce code cherche en premier le pgcd pour résoudre le problème de manière algorithmique car en cherchant le pgcd, on est au moins sûr d'avoir un diviseur commun supérieur à 1

du moin si les entiers n'ont pas entre eux que 1 comme diviseur en commun.