Correction des exercices
Semaine 3

Cours Turing

A Compter les votes

Pour compter les votes, on peut créer ce qui s'appelle une liste de fréquence. Comme il y a k votants, on initialise une liste de taille k avec 0 votes pour chaque candidat, puis si on lit que le candidat i a reçu un vote, on augmente le i-ème élement de la liste de 1. Suite à cela, on peut itérer dans la liste avec la fonction "enumerate" ce qui permet d'avoir en même temps le nombre de votes et le numéro du vote. Il suffit en suite de vérifier si une des entrée est plus grande que $\frac{n}{2}$.

```
n,k = map(int,input().split())
a = list(map(int,input().split()))
f = [0]*k
for i in a:
    f[i] += 1

s = "AUCUN"
for k,i in enumerate(f):
    if i > n/2:
        s = k
print(s)

for i in f:
    print(i, end=" ")
```

B Téléphérique

Dans cet exercie, on veut vérifier si la pente est régulière, géométriquement cela correspond à avoir une différence constante entre chaque pillier. On commence par supposer que c'est le cas, donc la différence des deux premier est la même qu'entre les deux suivant, etc... Si ce n'est pas le cas quelque part, on peut donc conclure que la pente du téléphérique n'est pas régulière.

```python
n = int(input())
a = list(map(int,input().split()))
ans = "OUI"
for i in range(n-2):
    if (a[i+2]-a[i+1] != a[i+1]-a[i]):
        ans = "NON"
print(ans)
```

C Magasin

Ici, on vérifie notre capacité à utiliser les fonctions associées aux sets. Plus précisement, les fonctions d'ajout ("add"), de suppression ("remove"), de longueur ("len"), et d'appartenance ("in").

```python
n = int(input())
s = set()

for _ in range(n):
    x = int(input())
    if x==0:
        print(len(s))
    if x>0:
        if x not in s:
            print("Merci !")
            s.add(x)
        else:
            print("Pas de place !")
    if x<0:
        if -x in s:
            print("Et voila !")
            s.remove(-x)
        else:
            print("Desole !")
```

D Extension du magasin

Maintenant, nous avons un exercice similaire mais avec des dictionnaires. Cependant, la taille n'est plus aussi facile à calculer. Il faut maintenant une variable supplémentaire pour garder le nombre total de fromages dans le magasin.

```python
n = int(input())
s = {}
tot = 0
for _ in range(n):
    x = int(input())
    if x == 0:
        print(tot)
    if x>0:
        tot += 1
        if x not in s:
            s[x] = 0
        s[x] += 1
        print("Merci !")
    if x<0:
        if -x not in s:
            print("Desole !")
        elif s[-x] <= 0:
            print("Desole !")
        else:
            s[-x] -= 1
            tot -= 1
            print("Et voila !")
```

E Problème de Josèphe

Cet exercice est clairement plus difficile à implémenter. Voici une manière de faire. Initialement je crée une liste de n 1. Chaque 1 représente un soldat en vie. Puis j'initialise 4 variables.

La première, compte le nombre de soldat en en vie, je sais qu'à un soldat en vie je dois m'arrêter.

J'ai ensuite une deuxième variable qui garde la somme du numéro de tous les soldats. Dès qu'un meurt je soustrais son numéro à sa somme, ce qui me permet de déterminer le numéro du dernier en vie. Cette étape n'est pas obligatoire, il est également possible de faire une boucle dans la liste une fois qu'il ne reste qu'un soldat en vie pour le trouver.

Puis une troisième qui garde en compte quel soldat je regarde.

Et une quatrième qui vérifie si le prochain soldat doit être épargné.

Puis j'itère dans ma liste, et à chaque fois que je vois un soldat en vie, soit je le "tue", soit je change la variable *skip* pour indiquer que le prochain doit mourir.

BONUS : analyser la complexité temporelle de ce programme. Ce n'est pas si facile non plus. On peut se rendre compte, qu'à chaque fois que je traverse ma liste j'élimine à peu près la moitié des soldats en vie (± 1). Et comme il faut diviser le nombre de soldat en vie par deux $\log_2(n)$ fois pour arriver à 1 soldat en vie et chacune de ces fois il faut $O(n)$ opérations, on peut conclure que notre programme fonctionne en $O(n \log n)$.

```python
n = int(input())

a = [1]*n
cnt = n
ans = n*(n+1) // 2
i = 0
skip = False
while cnt > 1:
    i = (i+1)%n
    if a[i] == 0:
        continue
    if skip:
        skip = False
    else:
        skip = True
        a[i] = 0
        cnt -= 1
        ans -= i+1

print(ans)
```