palindromes

idée : sur le site du Palais de la Découverte

Consignes	exemple
1. Choisir un nombre	1. 129
2. L'écrire à l'envers	2. 921
3. Additionner les deux nombres	3. $129 + 921 = 1\,050$
4. Recommencer avec ce nouveau nombre	4. $1\,050 + 501 = 1\,551$
5. Continuer de cette façon jusqu'à obtenir un palindrome	5. c'est terminé pour 129

1. Tester cet algorithme pour trois ou quatre entiers de votre choix.
2. A votre avis, obtient-on toujours un palindrome ?
3. Traduire cet algorithme en Python et le tester pour les entiers de 1 à 100.
4. Utiliser (ou modifier) ce programme afin de déterminer les entiers qui nécessitent le plus grand nombre d'étapes.
5. Tester pour des entiers supérieurs à 100.

Trace l'inégal palindrome. Neige. Bagatelle, dira Hercule. […] Haridelle, ta gabegie ne mord ni la plage ni l'écart.
Au moulin d'Andé : un palindrome de Georges Perec de 1 247 mots !
https://jeretiens.net/palindrome-de-georges-perec-au-moulin-dande

- conversion de type
- travail sur les chaînes

palindromes

```python
#-*- coding: utf8 -*-
# python3

def palindrome(a, b):
    """ pour chq entier <k> de [a ; b[, affiche le nb. d'étapes
    nécessaires pour obtenir le palindrome de <k>
    """
    for k in range(a, b):
        # <cpt> compte le nb d'étapes
        # <n> entier, <n_> est <n> écrit à l'envers
        # conversion de l'entier <n> en chaîne de
        # caractères, puis écriture "à l'envers" de
        # la chaîne, et la reconvertir en entier pour
        # obtenir <n_>
        n, n_ = k, int(str(k)[::-1])
        cpt = 0
        # tant que l'entier n est différent de l'entier n_
        # et on limite le nombre d'itérations !
        while n != n_ and cpt < 1000:
            n += n_ # n est augmenté de n_
            n_ = int(str(n)[::-1]) # palindrome de n
            cpt += 1
        print(f"{k} donne {n} en {cpt} étapes")
    return cpt # c'est mieux de renvoyer une valeur
```