FKLIIUH#GH#FHVDU

!!! abstract “Objectif” L’objectif de ce projet est de décrypter la phrase suivante:

```
`PRZRFFNTRARPBAGVRAGEVRAQVAGRERFFNAGZNVFVYRFGFHSSVFNZZRAGYBATCBHEARCNFYRQRPELCGRENYNZNVA`

Le principe est simple: chaque lettre de la phrase d'origine (le message *clair*) a été déca

Comme il n'y a que 25 décalages possibles, il vous faudra tous les tester un par un (méthode
```

!!! info “Pré-requis” En cryptographie, les méthodes sont très souvent numériques. Plutôt que de manipuler les caractères eux-mêmes, on préfère manipuler leurs représentations dans un encodage (on étudiera cette question un peu plus tard dans l’année).

```
Avec Python, on peut récupérer le [code Unicode](https://fr.wikipedia.org/wiki/Table_des_car

```python
>>> ord('A')
65
```

et le caractère correspondant à un code Unicode avec la fonction `chr`:

```python
>>> chr(65)
'A'
```
```

!!! example “Pour vous guider” === “1” Définissez une fonction `decale(lettre)` qui décale de 3 rangs dans l’alphabet la lettre majuscule `lettre` passée en argument (après Z, on recommencera à A. . .)

```
=== "2"
    Rajoutez un paramètre `n` à la fonction précédente pour pouvoir décaler la lettre de `n`

=== "3"
    Utilisez la fonction précédente pour créer la fonction `decale_phrase(p, n)` qui décale
```

!!! note “Ouverture” Écrire une fonction qui crypte une chaine de caractères avec au choix:

```
- le chiffre de Vigenère;
- la méthode du masque jetable;
```

!!! check “Proposition de correction” 1. `python linenums='1'  def decale_lettre(lettre: str) -> str:  '''  Renvoie la lettre située 3 places plus loin dans l'alphabet,  en revenant éventuellement au début.  '''  rang =`

```
ord(lettre) + 3         if rang > ord('A') + 25:             rang
= rang - 26         return chr(rang)
```

2.
```python linenums='1'
def decale_lettre(lettre: str, n: int) -> str:
    '''
    Renvoie la lettre située n places plus loin dans l'alphabet,
    en revenant éventuellement au début.
    '''
    rang = ord(lettre) + n
    if rang > ord('A') + 25:
        rang = rang - 26
    return chr(rang)
```

3.
```python linenums='1'
def decale_phrase(p: str, n: int) -> str:
    '''
    Renvoie la chaine des caractères de p tous décalés de n places
    '''
    phrase_decalee = ''
    for lettre in p:
        phrase_decalee += decale_lettre(lettre, n)
    return phrase_decalee

```