TP Python n°6

Table des matières

Chiffrement et déchiffrement

En bon français, on dit **chiffrer** et **déchiffrer**. Le terme **crypter** est un horrible angliscisme et **décrypter** est une opération différente (voir https://fr.wiktionary.org/wiki/crypter).

Principe : L'utilisateur veut écrire une phrase de façon chiffrée en utilisant une clé de chiffrement.

On vous fournit un exemple de clé de chiffrement représentée par un dictionnaire dans lequel chaque lettre à chiffrer est associée à sa version chiffrée.

Par exemple, si on chiffre `BONJOUR` avec la clé ci-dessous, on obtient `YLMQLFI`

```
cle_chiffrement = { "A": "Z",
                    "B": "Y",
                    "C": "X",
                    ...
                    "J": "Q",
                    ...
                    "N": "M",
                    "O": "L",
                    ...
                    "R": "I",
                    ...
                    "U": "F",
                    ...
                  }

# "BONJOUR" chiffré donne "YLMQLFI"
```

Travail demandé

En utilisant la structure de code décrite ci-dessus, écrivez un programme qui chiffre et déchiffre un mot avec le dictionnaire fourni (c.f. fichier `dictionnaire_cle.txt`)

Pensez à mettre tous les mots en majuscules avant de les chiffrer/déchiffrer.

Si un caractère du mot à chiffrer n'est pas dans le dictionnaire, on le laisse tel quel dans le chiffrement (caractères de ponctuation notamment)

```
def  chiffrer(mot_a_chiffrer , cle_chiffrement):
    """ Renvoie  la  version  chiffrée de  mot_a_chiffrer  en
        utilisant  le  dictionnaire  cle_chiffrement """
    # code de la  fonction  chiffrer


def  dechiffrer (mot_a_dechiffrer , cle_chiffrement):
    """ Renvoie  la  version déchiffrée de  mot_a_dechiffrer  en
        utilisant  le  dictionnaire  cle_chiffrement """
    # code de la  fonction  déchiffrer


if  __name__ =="__main__":
    cle_chiff = {...}

    # demander à l  utilisateur  un mot à chiffrer
    # chiffrer  ce mot
    # demander à l  utilisateur  un mot à déchiffrer
    # déchiffrer  ce mot
```

Le chiffre de César (clé = 1 lettre)

La clé du chiffrement de César peut se résumer à une lettre, par exemple `C`. Le dictionnaire est ensuite créé en décalant l'alphabet pour le faire commencer à la lettre clé.

Exemple de clé de chiffrement de César à partir de la lettre `C` :

```
cle_chiffrement = { "A": "C",
                    "B": "D",
                    "C": "E",
                    ...
                    "J": "L",
                    ...
                    "N": "P",
                    "O": "Q",
                    ...
                    "R": "T",
                    ...
                    "U": "W",
                    ...
                    "Y": "A",
                    "Z": "B"
                  }

# "BONJOUR" chiffré donne "DQPLQWT"
```

On utilise l'alphabet latin en majuscules "ABCDEFGHIJKLMNOPQRSTUVWXYZ" pour la création du dictionnaire.

Travail demandé

- Écrivez une fonction `rotate` qui, à partir d'un mot et d'un nombre de positions, renvoie une version de ce mot, décalée du nombre de positions indiqué.

  Par exemple, `rotate("ABCDEFGHIJKLMNOPQRSTUVWXYZ", 2)` renvoie `CDEFGHIJKLMNOPQRSTUVWXYZAB`.

  

  Utilisez les tranches.

- Écrivez une fonction `creer_dico` qui, à partir d'une lettre, renvoie le dictionnaire de chiffrement. Vous pouvez notamment utiliser la fonction `rotate` que vous venez d'écrire.
- Modifiez votre code pour demander une lettre à l'utilisateur et créer le dictionnaire de chiffrement à partir de cette lettre.

Clé = 1 mot

On peut pousser le chiffrement de César plus loin et utiliser un mot comme clé.

Pour chaque lettre du mot-clé, on crée un nouveau dictionnaire par décalage. Si le mot à chiffrer est plus long que la clé, on répète la clé autant de fois que nécessaire.

Par exemple, on peut chiffrer `BONJOUR` avec le mot-clé `POULE` : dans ce cas, on chiffre `B` avec la clé `P`, puis `O` avec la clé `O`, puis `N` avec la clé `U`, etc. Le mot complet chiffré sera donc `QCHUSJF`

Travail demandé

- Écrivez une fonction qui crée une liste de dictionnaires : une pour chaque lettre du mot-clé
- Écrivez deux nouvelles fonctions `chiffrer2` et `dechiffrer2` qui utilisent cette liste de dictionnaires pour chiffrer et déchiffrer un mot.