

Dans ce tp, on se fixe comme objectif de créer un code python qui permette de traduire un texte en français, dans une autre langue
.... vaste programme. On voit ça dans la suite.

1- PREPARATION :

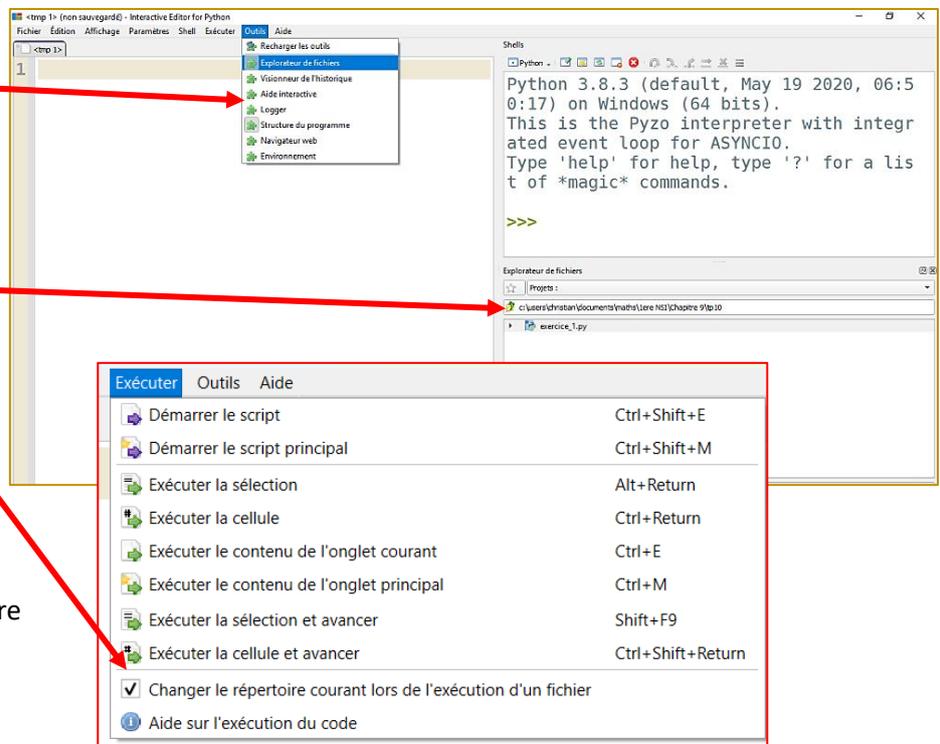
⇒ Créer un dossier nommé *tp11* dans votre espace de travail sur *U:*

⇒ Dans Pyzo, dans l'onglet *Outils*, afficher la fenêtre « *Explorateur de fichier* ».

⇒ Dans cet explorateur qui s'affiche, se positionner dans ce répertoire *tp11* sur *U:*

⇒ Dans l'onglet *Exécuter* de Pyzo, ne pas oublier de cocher la case « *changer le répertoire courant lors de l'exécution d'un fichier* »

⇒ Enregistrer votre fichier *.py*, vide pour l'instant, dans ce répertoire *tp11*.



2- RAPPELS SUR LES DICTIONNAIRES :



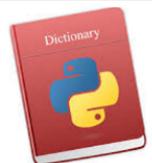
Chapitre 13 - Les dictionnaires en python

Les listes permettent de stocker des données lorsqu'elles sont nombreuses. Les éléments stockés dans une liste sont ordonnés. On y accède en utilisant un numéro qu'on appelle l'**indice** ou l'**index** de l'élément.

Un **dictionnaire** en Python va aussi permettre de rassembler des éléments mais ceux-ci seront identifiés par une **clé**. On peut faire l'analogie avec un dictionnaire de français où on accède à une définition avec un mot.

Contrairement aux listes qui sont délimitées par des crochets, on utilise des **accolades** pour les dictionnaires.

Remarque : Dans la plupart des langages de programmation, les listes sont appelées « *tableau associatif* ».



1- COMMENT CREER UN DICTIONNAIRE :

a. CREER UN DICTIONNAIRE VIDE ET LE REMPLIR :

On donne ci-contre les codes de 3 fonctions, accompagnés des lignes qui les exécutent. Les indentations ont malheureusement disparu et 2 lignes sont incomplètes.

⇒ Copier-coller ce code dans votre fichier .py

⇒ Remettre les indentations et corriger les 2 lignes incomplètes afin que l'exécution de ce script permette d'obtenir le contenu suivant pour les dictionnaires *uk*, *de* et *it* (contenu de ces dictionnaires à lire dans la console) :

```
>>> uk
{'je': 'i', 'aime': 'love', 'toi': 'you'}

>>> de
{'je': 'ich', 'aime': 'liebe', 'toi': 'du'}

>>> it
{'je': 'io', 'aime': 'amore', 'toi': 'voi'}
```



```
def anglais():
dic = {}
dic['je'] = 'i'
dic['aime'] = 'love'
dic['toi'] = 'you'
return dic
```

```
def allemand():
d = {'je':'ich', 'aime': 'liebe', 'toi': 'du'}
return
```

```
def italien():
d = {}
listeFrancais = ['je','aime','toi']
listeItalien = ['io','amore','voi']
for i in range(3):
cle = listeFrancais[i]
d[cle] =
return d
```

```
# Programme principal
uk = anglais()
de = allemand()
it = italien()
```

⇒ Modifier ces fonctions afin de pouvoir à présent obtenir le contenu suivant :

```
>>> uk
{'je': 'i', 'aime': 'love', 'toi': 'you', 'moi': 'me'}

>>> de
{'je': 'ich', 'aime': 'liebe', 'toi': 'du', 'moi': 'mich'}

>>> it
{'je': 'io', 'aime': 'amore', 'toi': 'voi', 'moi': 'me'}
```

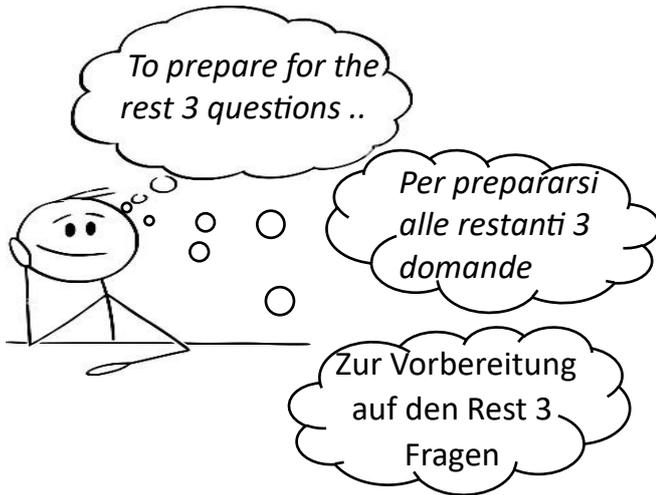
```
def anglais():
dic = {}
dic['je'] = 'i'
dic['aime'] = 'love'
dic['toi'] = 'you'
dic['moi'] = 'me'
return dic

def allemand():
d = {'je':'ich', 'aime': 'liebe', 'toi': 'du', 'moi': 'mich'}
return d

def italien():
d = {}
listeFrancais = ['je','aime','toi','moi']
listeItalien = ['io','amore','voi','me']
for i in range(4):
cle = listeFrancais[i]
d[cle] = listeItalien[i]
return d
```

Corrigé possible

⇒ 3 questions :



Question 1 : que faut-il exécuter dans la console pour obtenir

```
>>> uk['aime']  
'love'
```

Corrigé

Question 2 : que faut-il exécuter dans la console pour obtenir

```
>>> de["je"]  
'ich'
```

Question 3 : que faut-il exécuter dans la console pour obtenir

```
>>> 'je' in uk  
True
```

3- TRADUCTION – NIVEAU 1 :

On souhaite créer une fonction nommée *traductionMot()*.

Elle prend en argument un string et un dictionnaire. En exécutant le programme principal ci-contre, on obtient dans la console :

```
>>> (executing file "tp11.py")  
love  
amore
```

Programme principal

```
uk = anglais()  
de = allemand()  
it = italien()  
m = traductionMot("aime",uk)  
print(m)  
print(traductionMot("aime",it))
```

⇒ Ecrire le script python de cette fonction *traductionMot()*

Corrigé possible

```
def traductionMot(mot,dictionnaire):  
    motTraduit = dictionnaire[mot]  
    return motTraduit
```

4- TRADUCTION – NIVEAU 2 :

On souhaite à présent créer une fonction nommée *traductionPhrase()*. Elle prend aussi en argument un string et un dictionnaire. En exécutant le programme principal ci-contre, on obtient dans la console :

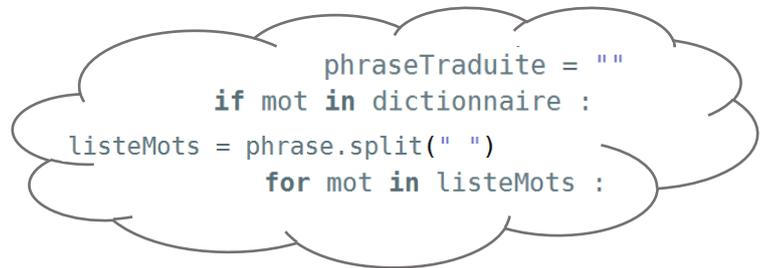
Programme principal

```
uk = anglais()  
de = allemand()  
it = italien()  
m = traductionPhrase("je aime toi",uk)  
print(m)  
print(traductionPhrase("je aime toi beaucoup",de))
```

```
>>> (executing file "tp11.py")  
i love you  
ich liebe du beaucoup
```

⇒ Ecrire le script python de cette fonction *traductionPhrase()*

Si tu veux de l'aide

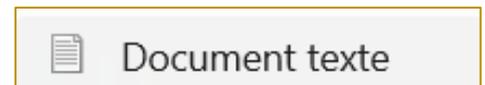


```
def traductionPhrase(phrase,dictionnaire) :  
    listeMots = phrase.split(" ")  
    phraseTraduite = ""  
    for mot in listeMots :  
        if mot in dictionnaire :  
            phraseTraduite = phraseTraduite + dictionnaire[mot] + " "  
        else :  
            phraseTraduite = phraseTraduite + mot + " "  
    return phraseTraduite
```

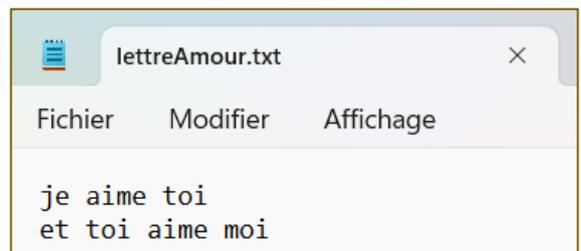
Corrigé possible

5- TRADUCTION – NIVEAU 3 :

⇒ Se positionner dans le dossier tp11 avec l'explorateur de fichier windows. En utilisant les options du clic droit, créer un nouveau fichier texte que vous nommerez *lettreAmour.txt*

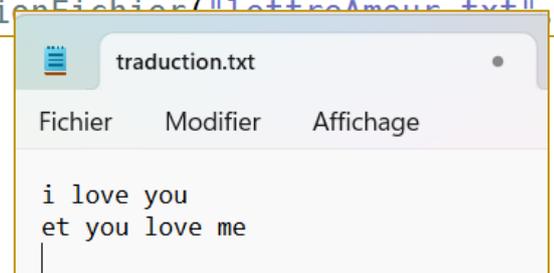


⇒ Ouvrir ce fichier et y écrire les 2 lignes ci-contre, avec un retour à la ligne en fin de seconde ligne.

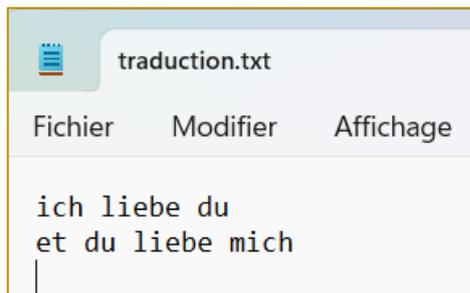


On veut à présent créer une fonction nommée *traductionFichier()*. Elle prend en argument un string et un dictionnaire. En exécutant le programme principal ci-contre, un fichier nommé *traduction.txt* avec le contenu suivant est créé :

```
# Programme principal  
uk = anglais()  
de = allemand()  
it = italien()  
traductionFichier("lettreAmour.txt", uk)
```



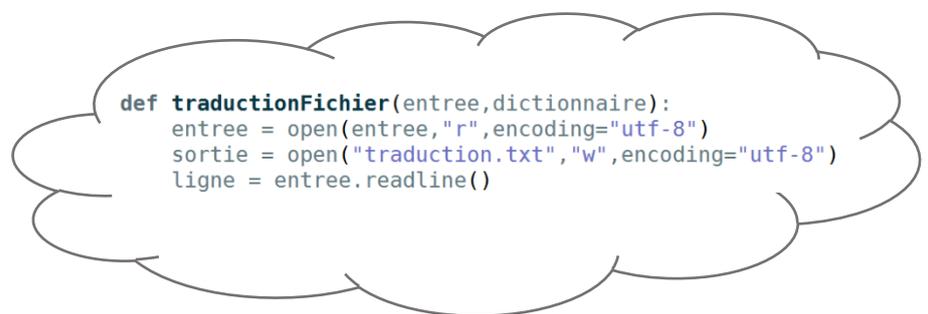
En remplaçant la ligne `traductionFichier("lettreAmour.txt",uk)` par la ligne ,
`traductionFichier("lettreAmour.txt",de)` , le contenu du fichier *traduction.txt*
devient :



```
traduction.txt
Fichier  Modifier  Affichage
ich liebe du
et du liebe mich
```

⇒ Ecrire le script python de cette fonction *traductionFichier()*

Si tu veux de l'aide



```
def traductionFichier(entree,dictionnaire):
entree = open(entree,"r",encoding="utf-8")
sortie = open("traduction.txt","w",encoding="utf-8")
ligne = entree.readline()
while ligne != "" :
    ligne = ligne[:-1]
    ligne = traductionPhrase(ligne,dictionnaire)
    ligne = ligne + "\n"
    sortie.write(ligne)
    ligne = entree.readline()
entree.close()
sortie.close()
```

Corrigé possible

6- TRADUCTION – NIVEAU 4 :

⇒ Sur nsibrantly.fr,
télécharger le zip lié à ce tp.
Il contient les 4 fichiers
textes de la figure ci-contre :

Nom	Statut	Modifié le	Type	Taille
dicAllemand.txt	🔄	04/02/2024 11:13	Document texte	23 Ko
dicAnglais.txt	🔄	04/02/2024 11:13	Document texte	22 Ko
dicItalien.txt	🔄	04/02/2024 11:13	Document texte	24 Ko
laChambreDesSecrets.txt	🔄	04/02/2024 10:34	Document texte	555 Ko

```
dicAnglais.txt
Fichier  Modifier  Affichage
|abandonner,give up
abri,shelter
absence,absence
absolu,absolute
absolument,absolutely
accent,accent
accord,agreement
acte,act
action,action
```

⇒ Décompresser ces 4 fichiers dans votre répertoire de travail tp11.

⇒ Les fichiers *dicAnglais.txt*, *dicAllemand.txt* et *dicItalien.txt* contiennent chacun 1461 lignes de mots les plus couramment utilisés dans la langue française, avec leur traduction respectivement en anglais, allemand ou italien.

On souhaite créer une fonction nommée *dicFichier()*. Elle prend en argument un string et renvoie un dictionnaire. En exécutant le programme principal ci-contre, les dictionnaires nommés *uk*, *de* et *it* déjà utilisés dans les paragraphes précédents, sont ici recréés en se basant sur les informations lues dans le fichier dont le nom est mis en argument. Par exemple, pour le début du dictionnaires nommé *uk*, on a :

```
# Programme principal
uk = dicFichier("dicAnglais.txt")
de = dicFichier("dicAllemand.txt")
it = dicFichier("dicItalien.txt")
```

```
>>> uk
{'abandonner': 'give up', 'abri': 'shelter', 'absence': 'absence', 'absolu': 'absolute', 'absolument': 'absolutely', 'accent': 'accent', 'accord': 'agreement', 'acte': 'act', 'action': 'action', 'affaire': 'affair', 'afin de': 'in order to'}
```

⇒ Ecrire le script python de cette fonction *dicFichier()*

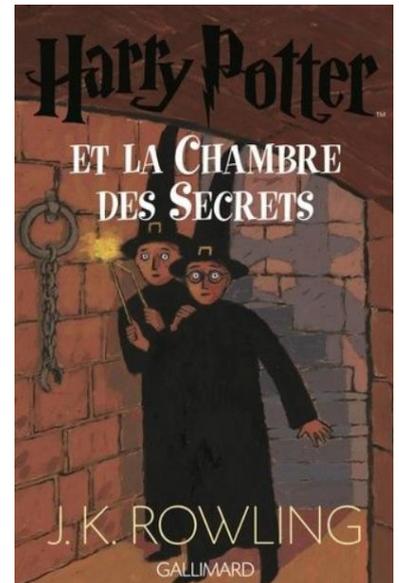
```
def dicFichier(nom) : Corrigé possible
    dic = {}
    fichier = open(nom,"r",encoding="utf-8")
    ligne = fichier.readline()
    while ligne != "":
        ligne = ligne[:-1]
        ligne = ligne.split(",")
        dic[ligne[0]]=ligne[1]
        ligne = fichier.readline()
    fichier.close()
    return dic
```

7- TRADUCTION – FINALISATION :

Le fichier *laChambreDesSecrets.txt* contient 6784 lignes extraites du roman de la saga des Harry Potters.

⇒ Utiliser les fonctions déjà écrites, pour traduire ces lignes dans un autre fichier texte. Le fichier contenant le roman traduit en anglais sera nommé *potterAnglais.txt*, celui traduit en allemand sera nommé *potterAllemand.txt*, celui traduit en italien sera nommé *potterItalien.txt*.

⇒ Le fichier *tp11.py* contenant toutes les fonctions et les fichiers *potterAnglais.txt*, *potterAllemand.txt*, *potterItalien.txt* sont à uploader sur nsibrantly.fr avec le code **tp11**.



```
# Programme principal Corrigé possible
uk = dicFichier("dicAnglais.txt")
de = dicFichier("dicAllemand.txt")
it = dicFichier("dicItalien.txt")

traductionFichier("laChambreDesSecrets.txt",uk)
```

8- BONUS :

Une fonction *valeurMot()* prend en argument un string composé de lettres majuscules. Cette fonction retourne le nombre de points que rapporte ce mot au scrabble.

Par exemple l'exécution de

```
>>> valeurmot("TAXI")
13
```

retourne 13.

Ecrire le script de cette fonction.

Remarque : Dans ce script, un dictionnaire nommé scrabble est créé. Il associe à chaque lettre, son nombre de points. Remplir ce dictionnaire avec un minimum d'instructions.

A ₁	B ₃	C ₃	D ₂	E ₁
F ₄	G ₂	H ₄	I ₁	J ₈
K ₁₀	L ₁	M ₂	N ₁	O ₁
P ₃	Q ₈	R ₁	S ₁	T ₁
U ₁	V ₄	W ₁₀	X ₁₀	Y ₁₀
		Z ₁₀		

```
def valeursMot(mot) :
    points = [1,3,3,2,1,4,2,4,1,8,10,1,2,1,1,3,8,1,1,1,1,4,10,10,10,10]
    dic = {}
    majuscules = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

    for i in range(26) :
        dic[majuscules[i]] = points[i]
    p = 0
    for l in mot :
        p = p + dic[l]
    return p
```

Corrigé possible