Projet : Jeu du Pendu

principe du projet

- Vous travaillerez par binomes sur ce projet.
- Vous rendrez le code complet + votre dossier personnel pour le 16 octobre 2021 au plus tard.
- Vous aurez un temps en classe pour réaliser le projet, mais ce temps ne sera pas suffisant ! Vous devrez vous coordonner pour arriver à vos fins !

1. Description du projet

jeu du Pendu

Le principe retenu pour le jeu est le suivant :

1. Un mot français est tiré aléatoirement depuis un fichier externe.
2. Ce mot est *nettoyé* des accents et autres signes diacritiques français (`ç`, `œ`, `æ`, ...) puis converti en majuscules. **Les tirets des mots composés sont conservés !**
3. On affiche une série de tirets correspondant aux emplacements des lettres ainsi que la potence de départ du pendu.
4. On demande à l'utilisateur une lettre saisie au clavier, et on vérifie que :
   - c'est bien **une unique lettre** ;
   - elle n'a pas déjà été proposée.
5. Si la lettre fait partie du mot cherché, les tirets correspondants sont remplacés par la lettre. Sinon on ajoute un élément au dessin du pendu.
6. On reprend à l'étape 4 tant que :
   - soit il reste des lettres à trouver ;
   - soit le pendu est complètement dessiné (6 échecs).
7. Une fois la partie terminée, le joueur peut alors choisir de recommencer ou non une nouvelle partie.

Le rendu final devrait ressembler si possible à celui de la vidéo suivante :

2. Les fichiers nécessaires

Mise en place du dossier de projet

1.Vous commencerez par créer un dossier `NSI` dans votre dossier `Documents` de votre répertoire personnel.

2.Puis vous créerez un sous dossier `C01` dans le dossier `NSI`.

Ce dossier final devra donc avoir comme adresse absolue :

`P:\Documents\NSI\C01`

Fichier de mots

Pour mener à bien le projet, vous aurez d'abord besoin du fichier **des mots français.**.

Vous téléchargerez celui-ci et le copierez dans `P:\Documents\NSI\C01`

J'ai retiré de celui-ci tous les mots contenant des signes diacritiques autres que :

- les lettres accentuées ;
- les tirets ;
- les cédilles ;
- les lettres combinées (e dans l'o,...)

Les mots peuvent être en majuscule, en minuscule, ou toute autre combinanison de casse.

Base de code Python

Je vous donne le code ci-dessous qui doit être votre base de travail. Ce code sera sauvegardé dans un fichier `ProjetPendu.py` qui sera enregistré à l'aide du logiciel `Thonny` dans `P:\Documents\NSI\C01`

```
from random import choice


def choixMot(adresseFichier) :
    """ fonction ouvrant un fichier texte dont l'adresse absolue ou relative
    est passée en argument sous la forme d'une chaine de caractère
    et renvoyant une chaine de caractère issue d'une ligne aléatoire du fichier
"""
    with open(adresseFichier,'r', encoding='utf8') as file :
        mot = choice([m for m in file.readlines()]).replace('\n', '').strip()
    return mot

def formateMot(mot) :
    """ fonction transformant une chaine de caractères accentués
en une chaine de caractère latin strict (sans accents ni signes diacritiques).
La chaine renvoyée est en majuscule.

>>> formateMot('tRuC')
'TRUC'
>>> formateMot('Abécédaire')
'ABECEDAIRE'
>>> formateMot('')
''
>>> formateMot('où')
'OU'
>>> formateMot('garçONs')
'GARCONS'
>>> formateMot('àâäéèêëîïôöûùüç')
'AAAEEEEIIOOUUUC'
>>> formateMot('œil')
'OEIL'
>>> formateMot('Lætitia')
'LAETITIA'
"""
    ...

def genereTirets(motATrouver,lettresUtilisees) :
    """ fonction renvoyant une chaine de caractère correspondant
au mot à trouver pour lequel :
* les caractères non présents dans la chaine lettreUtilisees
sont remplacés par des _ (underscores) ;
* les tirest hauts "-" sont conservés ;
* tous les caractères sont suivis d'un espace, y compris le dernier.

>>> genereTirets("Bidules", "Ble")
'B _ _ _ l e _ '
>>> genereTirets("toto", "")
'_ _ _ _ '
>>> genereTirets("bananes", "bn")
'b _ n _ n _ _ '
>>> genereTirets("toto", "ot")
't o t o '
>>> genereTirets("pull-over", "plr")
'p _ l l - _ _ _ r '
>>> genereTirets("pull-over", "pulover")
'p u l l - o v e r '
"""
    ...


def compteRestantes(motATrouver,lettresUtilisees) :
    """ fonction renvoyant le nombre de lettres non encore trouvées
dans le mot, en connaissant les lettres déjà utilisées.
Un tiret haut "-" ne compte pas dans les lettres à trouver.
La valeur renvoyée est un entier

>>> compteRestantes("bananes","bn")
4
>>> compteRestantes("toto","to")
```

```
0
>>> compteRestantes("toto","")
4
>>> compteRestantes("","")
0
>>> compteRestantes("","z")
0
>>> compteRestantes("bidules","bidule")
1
>>> compteRestantes("pull-over", "plr")
4
>>> compteRestantes("pull-over", "pulover")
0

"""
    ...

def affichePendu(motATrouver, lettresUtilisees, nbEchecs) :
    """ fonction affichant à la fois la potence mais aussi le mot
à trouver sous sa forme de tirets
    """
    ...

def demandeJoueurLettre():
    """ fonction demandant une lettre latine non accentuée au joueur,
et renvoyant cette lettre en majuscule. La fonction redemande au joueur
tant que celui-ci n'a pas fourni une lettre correcte.
La lettre est renvoyée en par la fonction.
"""
    ...


def uneManche() :
    """ fonction déclenchant une manche de jeu. On entend par manche de jeu :
* le choix d'un mot dans le fichier 'liste_francais_modifiee.txt' ;
* le formatage de ce mot ;
* puis la répétitions de :
    * la demande d'une lettre au joueur ;
    * la mise à jour des lettres utilisée le cas échéant ;
    * la mise à jour de l'affichage

    jusqu'à ce que soit le mot complet ait été trouvé,
    soit que le dessin du pendu soit terminé (6 étapes).
    """
    ...

def presentation() :
    """ fonction affichant uniquement la présentation"""
    print("""
##############################################
#                                            #
#               Jeu du Pendu                 #
#                                            #
# 1ère NSI 2021-2022                         #
##############################################
""")


def main() :
    """ fonction principale du jeu, permettant d'effectuer plusieurs manches"""
    while True :
        presentation()
        uneManche()
        rep = input("Voulez-vous rejouer ? (o/n)")
        if rep.lower() not in ['o', 'oui', 'y', 'yes'] :
            break
    print("Au revoir !")

## La partie ci-dessous n'est effectuée que si vous déclenchez le programme
## en tant que programme principal (notion de modules, vue en terminale)

if __name__ == "__main__" :

    import doctest
    doctest.testmod()
    main()
```

fonction `choixMot(adresseFichier)`

Cette fonction **ne doit pas être modifiée !**

Elle prend en argument l'adresse **relative** ou **absolue** d'un fichier texte, et renvoie la chaine de caractères correspondant à une ligne de ce fichier, où :

- les éventuels espaces de début et de fin de ligne sont supprimés ;
- les caractères *retour chariot* (sauts de lignes) `\n` sont supprimés.

Dans le cadre de ce projet, le fichier `liste_francais_modifiee.txt` doit normalement être situé dans le même répertoire que `ProjetPendu.py`. Donc vous pouvez utiliser cette fonction de la manière suivante :

```
mot = choixMot("liste_francais_modifiee.txt")
```

Ainsi la variable `mot` contiendra un mot extrait aléatoirement du fichier.

3. Plan de travail

1. Vous commencerez par compléter la fonction `formateMot(mot)`, afin qu'elle renvoie une chaîne de caractères en **majuscule** dans laquelle tous les **signes diacritiques** ont été supprimés (à par les tirets des mots composés). Des **tests unitaires** sont donnés à titre d'exemple Vous pouvez éventuellement rajouter les votres.
2. Vous complèterez ensuite la fonction `genereTirets(motATrouver,lettresUtilisees)`, qui renvoie une chaîne de caractères correspondant à celle passée en premier argument `motATrouver`, pour laquelle les lettres **non présentes** dans la chaîne de caractère `lettresUtilisees`. De plus, chaque caractère de la chaîne finale **devra être suivi d'un espace**. Des **tests unitaires** sont donnés à titre d'exemple Vous pouvez éventuellement rajouter les votres.
3. Vous complèterez la fonction `compteRestantes(motATrouver,lettresUtilisees)` qui renvoie un entier correspondant au nombre de lettres restant à trouver dans `motATrouver` sachant la chaîne de lettres déjà utilisées `lettresUtilisees`.
4. Vous complèterez ensuite la fonction `demandeJoueurLettre()` et la rendrez *dumbproof* : cette fonction doit continuer à redemander au joueur de saisir une lettre tant que celle-ci n'est pas compatible avec les règles du jeu.
5. Vous complèterez ensuite la fonction `affichePendu(motATrouver, lettresUtilisees, nbEchecs)` qui affiche non seulement la potence, amis aussi le mot à trouver sous sa forme de tirets. Pour construire cette fonction, vous utiliserez une *f-string* multi-lignes telle que :

```
f"""

   _ _ _
  o     |
/ | \   |
 / \    |
        |
 _______|__
"""
```

6. A partir de toutes les fonctions précédentes, vous finaliserez le jeu en complétant la fonction `uneManche()`.

1. Une fois le jeu complété, vous devrez en outre compléter un **dossier personnel** d'une ou deux pages présentant :
2. ce que vous avez réalisé individuellement dans ce projet ;
3. les difficultés rencontrées et/ou les problèmes que vous n'avez pas pu résoudre ;
4. les aides qui vous on été apportées.
5. Vous pourrez enfin apporter des modifications et/ou améliorations au code, par exemple en :
   - ajoutant un compteur de score qui donne le nombre de réussites par rapport au nombre de parties jouées.
   - ajouter un niveau de difficulté, en changeant le nombre d'erreurs possibles ;
   - etc...

4. Grille de notation

intitulé	barême	Détails
fonction `formateMot`	2 pt	passage de tous les tests unitaires
fonction `genereTirets`	2 pt	passage de tous les tests unitaires
fonction `compteRestantes`	1 pt	passage de tous les tests unitaires
fonction `demandeJoueurLettre`	1 pt	*dumbproof*
fonction `affichePendu`	2 pt	Affichage correct
fonction `uneManche`	3 pts	On attend un jeu a minima fonctionnel
Noms des variables clairs	2 pts	On proscrira les noms de variable d'un seul caractère, sauf compteurs précis
Code commenté et clair	3 pts	Des explications minimales doivent être écrites pour expliquer votre code
Réalisation d'un dossier personnel	2pts	Rendu au format PDF ou ODT
Améliorations, qualité du code, etc...	2 pts	