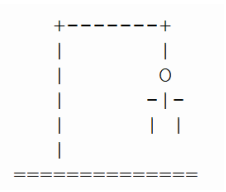
Jeu du pendu

L'activité va permettre d'implémenter le jeu du pendu.

1. Le jeu du pendu

Le but de l'activité est de compléter le programme en langage python, permettant de jouer au pendu.



La vidéo **Jeu du pendu en Python** donne un aperçu du résultat attendu (https://youtu.be/mDk-iWhrfC8).

2. À disposition

2.1. La liste des mots

La liste des mots, pour le jeu, est contenue dans un fichier texte (dico.txt).
La fonction ci-dessous permet de lire le fichier et créer une liste de mots.

```
def liste_de_mot():
    """création d'une liste de mot à partir du fichier dico.txt
    sortie : list_mot (list)
    """
    #ouverture du fichier dico.txt en lecture (r) et au format texte (t)
    #et enregistrement dans la variable fichierLu
    fichier_lu = open('dico.txt', 'rt')
    #lecture des lignes pour créer une liste
    liste = fichier_lu.readlines()
    #fermeture du fichier
    fichier_lu.close()
    return liste
```

2.2. Dessin du pendu

La fonction ci-dessous permet de dessiner le pendu.
Les triples guillemets """ permettent de délimiter une chaîne de caractères située sur plusieurs lignes (chaîne de caractères multi-lignes).

```
def dessin_pendu(nb):
    """renvoie une valeur du tableau correspondant à un dessin du pendu en fonction du nombre
rentré
    entrée : nb (int)
    sortie : tab (String)
    """
    tab=[
    """
       +-------+
       |
       |
       |
       |
       |
    ==============
    """,
    """
       +-------+
       |       |
```

```
    |       O
    |
    |
    |
==============
""",
"""
   +-------+
   |       |
   |       O
   |       |
   |
   |
==============
""",
"""
   +-------+
   |       |
   |       O
   |      -|
   |
   |
==============
""",
"""
   +-------+
   |       |
   |       O
   |      -|-
   |
   |
==============
""",
"""
   +-------+
   |       |
   |       O
   |      -|-
   |      |
   |
==============
""",
"""
   +-------+
   |       |
   |       O
   |      -|-
   |      | |
   |
==============
"""
]
return tab[nb]
```

2.3. Programme principal

Le programme principal assure la coordination entre toutes les fonctions ainsi que l'interface avec le joueur.

```
##nombre de tentative fausse
nb = 0
#création d'une liste de mot à partir d'un fichier
list_word = liste_de_mot()
#nettoyage dela liste des mots (\n)
list_clean = liste_de_mot_propre(list_word)
#choix du mot mystere
mot_mystere = choix_mot(list_clean)
#création du mot à trou à partir du mot mystère
mot_a_trou = creation_mot_vide(mot_mystere)
print('JEU DU PENDU')
print()
win = False
perdu = False
while (win != True) and (perdu != True) :
    print(convertir_liste_chaine(mot_a_trou))
    print()
    test_deja_lettre = True
    while test_deja_lettre == True:
        lettre_proposee = input('Proposes une lettre : ')
        print(lettre_proposee)
        test_deja_lettre = test_lettre_deja_proposee(liste_lettre_proposee, lettre_proposee)
        if test_deja_lettre == True:
            print('La lettre a déjà été proposée !')
    ajout_liste_lettre_proposee(liste_lettre_proposee, lettre_proposee)
    lettre_trouvee = test_lettre_dans_mot_a_trouver(mot_mystere, lettre_proposee)
    if lettre_trouvee == True:
        mot_a_trou = complete_mot_a_trou(mot_mystere, mot_a_trou, lettre_proposee)
        win = gagne(mot_mystere, mot_a_trou)
    else:
        print(dessin_pendu(nb))
        nb+=1
        if nb == 7:
            perdu = True
print()
if win:
    print(convertir_liste_chaine(mot_a_trou))
    print('BRAVO')
else:
    print('PERDU')
    print(f'Le mot à trouver était : {convertir_liste_chaine(mot_mystere)}')
```

3. Fonctions à compléter

3.1. Fonction `liste_de_mot_propre`

La liste crée à partir du fichier dico.txt contient après chaque mot les deux caractères correspondant à un retour à la ligne (**\n**).
Par conséquent, la fonction **liste_de_mot_propre** doit supprimer pour chaque mot de la liste les deux derniers caractères (**\n**) et retourner une liste propre.
La fonction répond à l'assertion ci-dessous :

```
>>> liste_de_mot()
['ANGLE\n', 'ARMOIRE\n', 'BANC\n', 'BUREAU\n',
OULOIR\n', 'DOSSIER\n', 'EAU\n', 'ECOLE\n', 'EN
UR\n', 'LAVABO\n', 'LIT\n', 'MARCHE\n', 'MATELA
CARD\n', 'PLAFOND\n', 'PORTE\n', 'POUBELLE\n',
```

```
assert liste_de_mot_propre(['LAPIN\n', 'GIRAFE\n']) == ['LAPIN', 'GIRAFE']
```

3.2. Fonction choix_mot

Comme il est plus facile de manipuler le mot mystère et le mot à trou sous forme de liste de charactères que de chaîne de caractères, la fonction **choix_mot** retourne sous forme de liste un mot pris au hasard dans la liste.
La fonction répond aux exemples ci-dessous :

```
>>> choix_mot(['LAPIN', 'GIRAFE'])
['L', 'A', 'P', 'I', 'N']

>>> choix_mot(['LAPIN', 'GIRAFE'])
['G', 'I', 'R', 'A', 'F', 'E']
```

3.3. Fonction création_mot_vide

La fonction création_mot_vide crée une liste de n caractères _ égale au nombre n de lettres constituant le mot.

```
>>> creation_mot_vide(['G', 'I', 'R', 'A', 'F', 'E'])
['-', '-', '-', '-', '-', '-']
```

3.4. Fonction complete_mot_a_trou

La fonction **complete_mot_a_trou(mot_mystere, mot_a_trou, lettre)** complète la variable **mot_a_trou** avec la **lettre** à l'endroit où elle se situe dans le **mot_mystere**.

```
>>> complete_mot_a_trou(['L', 'A', 'P', 'I', 'N'], ['-', '-', '-', '-', '-'], 'A')
['-', 'A', '-', '-', '-']

>>> complete_mot_a_trou(['L', 'A', 'P', 'I', 'N'], ['-', '-', '-', '-', '-'], 'B')
['-', '-', '-', '-', '-']
```