

DM info : jeu du pendu

Le but de ce DM est de créer un jeu du pendu sur Python.

Le jeu devra se présenter comme suit. Quand on lance le programme, l'ordinateur choisi un mot de 7 lettres au hasard (parmi une liste de mots pré-enregistrés). Le joueur proposera une lettre et l'ordinateur devra dire si la lettre est bien dans le mot choisi. Si c'est le cas, l'ordinateur explicitera la (ou les) place(s) de la lettre présente dans le mot.

Le joueur aura un certain nombre d'essais (disons 10) afin de trouver le mot.

Si le joueur trouve le mot le programme affiche 'gagné' et s'arrête. Si le joueur ne trouve pas le mot le programme affiche 'perdu' et s'arrête.

Exercice 1. 1. On suppose que l'on dispose d'une liste `dictionnaire` contenant tous les mots du dictionnaire. (Pour vos testes, créer une liste `dictionnaire` contenant trois mots : 'coucou', 'olivier', 'matrice')

Ecrire une fonction `mots_de7lettres` qui retourne une liste ne contenant que les mots de 7 lettres du `dictionnaire`.

2. Ecrire une fonction `choix_mot`, qui choisit un mot de 7 lettres aléatoirement. (On pourra utiliser la fonction `len` qui prend en argument une chaîne de caractères et qui retourne sa taille, (comme pour les listes))

3. Ecrire une fonction `transform` qui prend en argument une chaîne de caractères `S` et retourne une liste dont chaque entrée est une lettre de la chaîne `S`.

4. Créer une fonction `test_lettre` qui prend en argument un mot `M` et une lettre `a` et retourne la (ou les) position de la lettre `a` dans le mot `M`. Si `a` n'est pas dans le mot, la fonction retournera la liste vide.

Exemples : `test_lettre('olivier', 'i')` -> [2, 4] et `test_lettre('olivier', 'w')` -> []

5. Ecrire une fonction `reponse` qui prend en argument deux chaînes de caractères. L'une `M` correspondant au mot que l'on doit trouver et l'autre `P` correspondant à la proposition du joueur. Si le joueur propose une seule lettre alors la fonction `reponse` retourne la (ou les places) de la lettre dans le mot `M`, si le joueur propose un mot (donc plusieurs lettres) alors la fonction `reponse` retourne `True` si le mot est bon et une liste vide si le mot est mauvais. (La liste vide permet d'être dans le même cas que si on avait donné juste une lettre qui n'est pas dans le mot)

6. Ecrire une fonction `lettre_connue` qui prend en argument un mot `M` (correspondant au mot à trouver) et une liste `L` (qui correspond au position des lettres déjà trouvées) et qui retourne une chaîne de caractères où les lettres dont la position sont dans `L` s'affiche en claire et sinon sont remplacées par des '*'

Exemple : `lettre_connue('olivier', [1,2,5])` retourne '*li**i**' `lettre_connue('matrice', [0,1,2])` affiche 'ma*****'

7. Compléter le code suivant qui permet de jouer au jeu du pendu sans limite d'essais.

```

1 def pendu():
2     mot_a_trouver=.....
3     mot_propose=""
4     list_lettres_connues=[]
5     while mot_propose != .....:
6         l=lettres_connues(.... , ....)
7         # On affiche les lettres deja trouvees par le joueur)
8         print(l)
9
10    mot_propose = input('Donner_une_lettre_ou_une_proposition')
11    #on demande au joueur une nouvelle lettre ou une nouvelle proposition
12
13    rep=reponse(...., .... )
14    #on analyse la reponse.
15
16    if ...==.... : #si le joueur a trouve le bon mot
17        print( .... )# on le felicite
18        return #on arrete le programme
19
20    else: #sinon
21        if len(rep).....: #soit le mot n'est pas le bon ou la lettre n'est pas dans le mot
22            print('essaye_encore') #et on lui dit de reesayer
23
24        else:
25            print('il_y_a_ +..... +'_lettre_' + .... )
26            #on affiche le nombre de fois ou la lettre proposee
27            #apparaît dans le mot cherche
28
29            list_lettres_connues=....+....
30            #et on ajoute a la liste des lettres
31            #connues les nouvelles lettres.

```

8. Améliorer la fonction `pendu` pour que le programme s'arrête après 10 essais infructueux et affiche à chaque essais le nombre de tentatives restantes.