JEU DU PENDU

CASTRE Théo

09/24

Contexte

Le projet JEU DU PENDU a été réalisé dans le cadre d'une SAE. L'objectif principal était de concevoir un programme fonctionnel, simple et interactif en ligne de commande, tout en respectant des exigences pédagogiques précises, telles que l'utilisation de fonctions modulaires, la récursivité et la validation des entrées.

Table des matières

But du Projet

Le jeu du pendu est un jeu classique dans lequel un joueur doit deviner un mot caché en proposant des lettres une par une. Le but du projet est de :

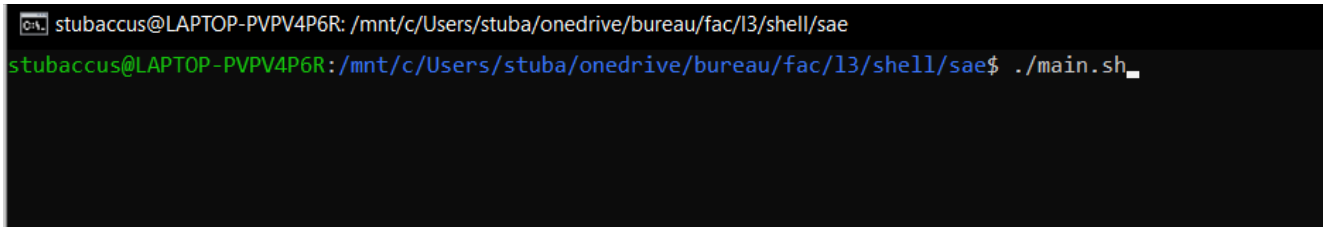
- Créer une expérience ludique et pédagogique qui permet d'appliquer des connaissances en programmation Shell.
- Structurer un programme en plusieurs fichiers pour favoriser la réutilisation du code.
- Respecter des contraintes techniques spécifiques, comme l'utilisation d'une fonction récursive et la gestion des fichiers.

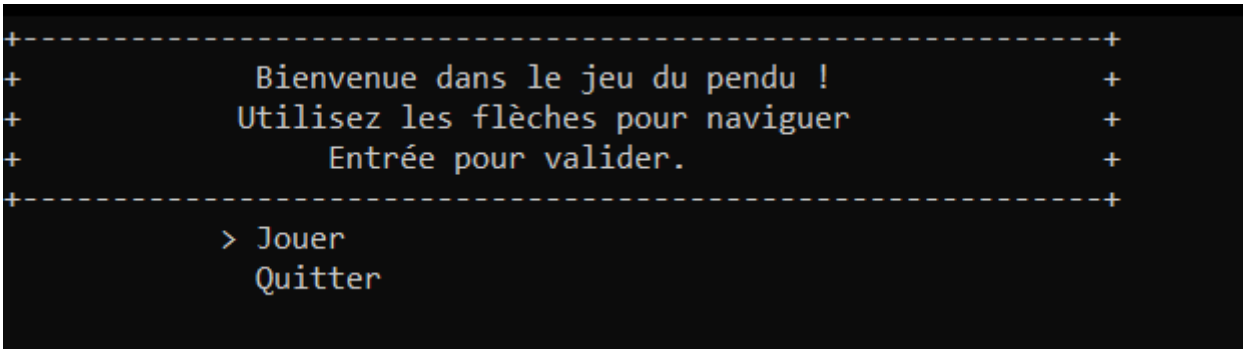
Algorithmes et Fonctions

1. get_random_word
    - Description : Sélectionne un mot au hasard dans le fichier input,txt
    - Paramètres : Aucun.
    - Retour : Une chaîne de caractères correspondant à un mot du fichier.
2. initialize_game
    - Description : Initialise le mot masqué avec des underscores _ représentant les lettres du mot à deviner.
    - Paramètres : Le mot à deviner.
    - Retour : Une chaîne de caractères masquée (ex. _ _ _ _ _).
3. check_letter
    - Description : Met à jour le mot masqué en fonction des lettres correctement devinées.
    - Paramètres : Lettre proposée, mot complet, état actuel du mot masqué.
    - Retour : Mot masqué mis à jour.
4. count_occurrences (fonction récursive)
    - Description : Compte le nombre d'occurrences d'une lettre donnée dans le mot.
    - Paramètres : Lettre, mot, index actuel dans le mot.
    - Retour : Nombre d'occurrences.
5. display_menu
    - Description :
      Cette fonction affiche le menu principal du jeu, permettant à l'utilisateur de choisir entre les options "Jouer" et "Quitter". Elle gère également la navigation dans le menu à l’aide des flèches directionnelles et valide la sélection avec la touche Entrée.
      Le menu est conçu pour offrir une expérience utilisateur interactive et intuitive, même dans un environnement en ligne de commande.
    - Paramètres : Aucun.
    - Retour :
    - `0` : si l'utilisateur sélectionne l'option "Jouer".
    - `1` : si l'utilisateur sélectionne l'option "Quitter".

Lancement du projet

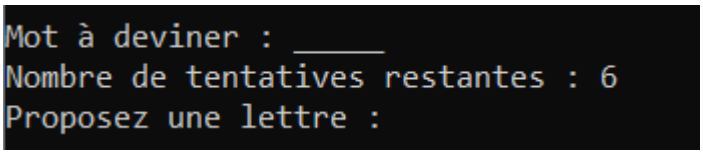
Il suffit de lancer le script main.sh sans paramètre. Ce script affiche un menu permettant de jouer au jeu ou alors de le quitter.





Lorsqu'une partie commence, le programme sélectionne un mot aléatoire dans le fichier input.txt, qui contient une liste de mots en français (un par ligne).

La fonction get_random_word est utilisée pour lire un mot au hasard. Elle s'assure de supprimer les espaces ou caractères superflus comme les sauts de ligne.
Ce mot devient le mot à deviner, et un équivalent masqué (par exemple _ _ _ _ _ pour un mot de 5 lettres) est initialisé grâce à la fonction initialize_game.



L'utilisateur propose une lettre à chaque tour.
Le programme valide l'entrée pour éviter les erreurs courantes :
Vérifie que l'utilisateur entre une seule lettre valide (ni chiffres, ni symboles).

```
Mot à deviner : _____
Nombre de tentatives restantes : 6
Proposez une lettre : 58
Erreur : veuillez entrer une seule lettre valide (a-z).
Proposez une lettre :
```

Empêche la répétition de lettres déjà proposées en vérifiant la liste des lettres précédemment saisies.

```
Proposez une lettre : o
Erreur : Vous avez déjà proposé la lettre 'o'.
Proposez une lettre :
```

Si l'entrée est incorrecte, un message d'erreur est affiché, et l'utilisateur est invité à réessayer.

Le test du mot avec une expression régulière (regex) consiste à comparer la lettre entrée par l'utilisateur avec chaque lettre du mot à deviner.
Si la lettre correspond à une ou plusieurs positions dans le mot, elle est révélée dans le mot masqué (grâce à la fonction check_letter).

```
La lettre 'l' apparaît 2 fois.
Mot à deviner : s__ll
Nombre de tentatives restantes : 5
Proposez une lettre :
```

Sinon, le programme considère cette tentative comme une erreur et réduit le compteur de tentatives restantes.

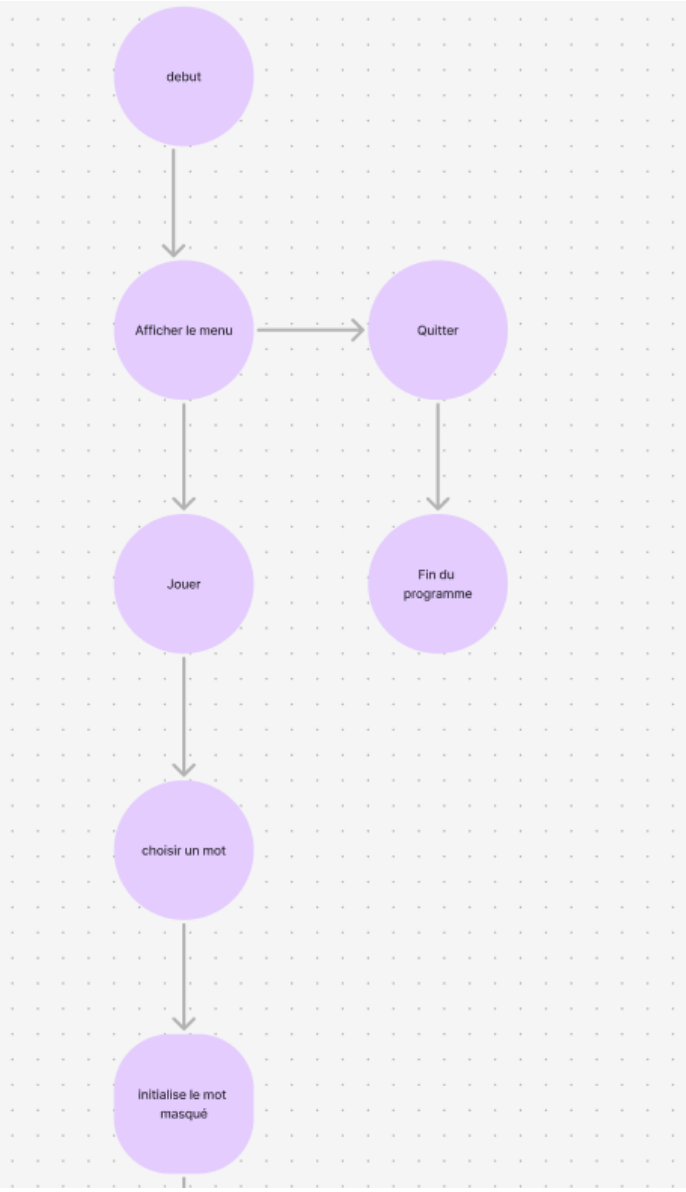
```
Mauvaise lettre !
Mot à deviner : _____
Nombre de tentatives restantes : 5
Proposez une lettre :
```

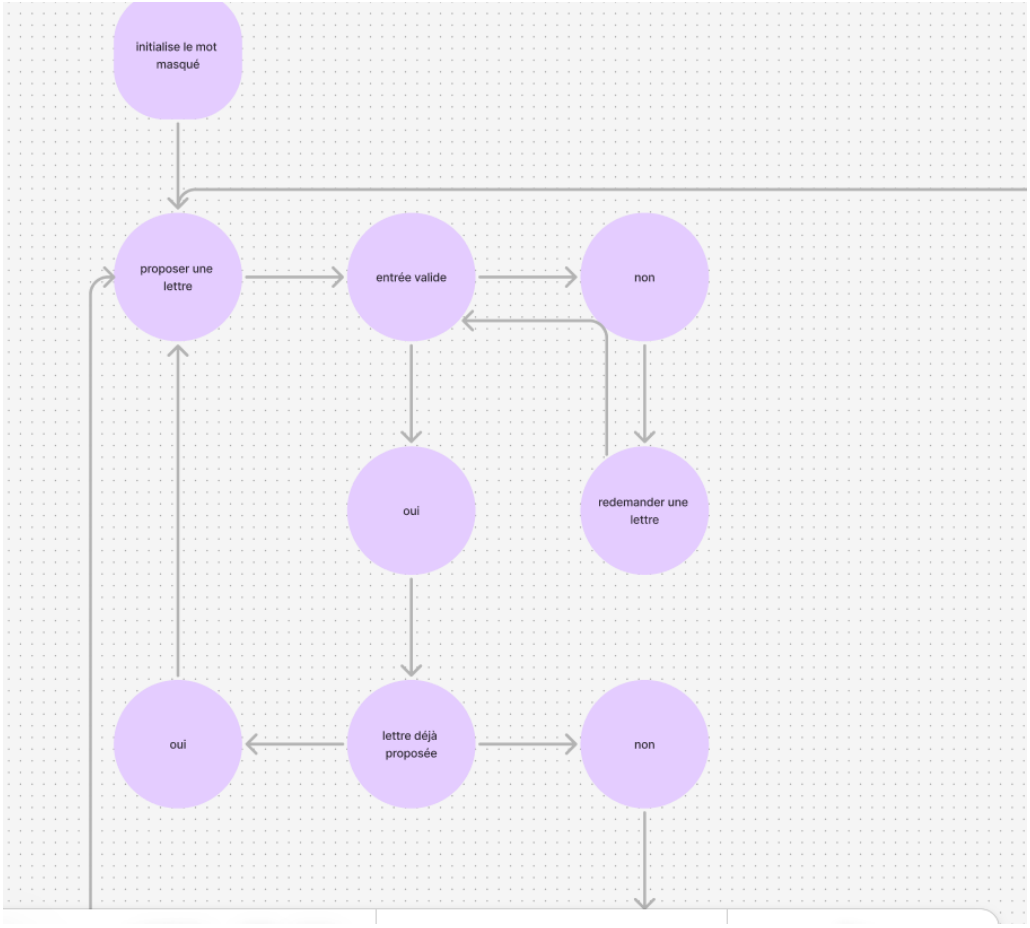
Chaque lettre proposée est stockée dans une liste. À chaque tour, le programme :
Vérifie si la lettre est déjà présente dans cette liste.
Si oui, il affiche un message comme : "Vous avez déjà proposé cette lettre !"
Si non, la lettre est ajoutée à la liste et testée sur le mot à deviner.
Ce mécanisme empêche l'utilisateur de perdre inutilement des tentatives en répétant les mêmes lettres.

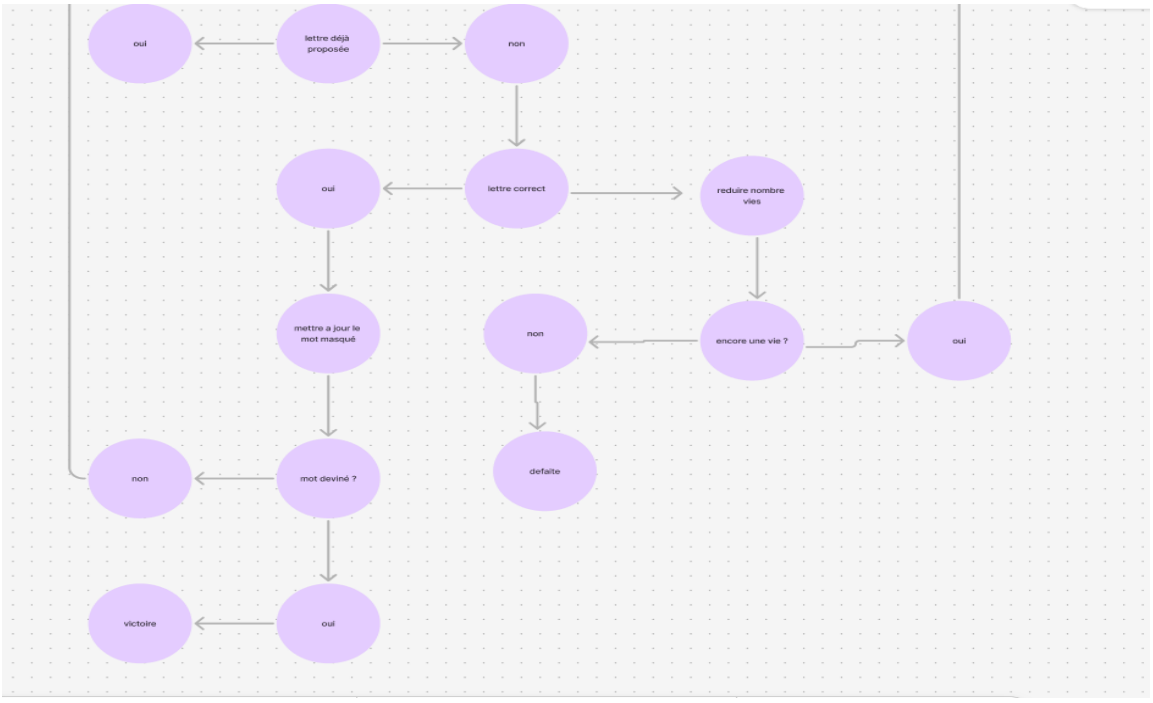
Enfin le jeu se termine dans deux situations :
Victoire : Si l'utilisateur révèle toutes les lettres du mot avant d'épuiser ses tentatives, il gagne. Un message de félicitations s'affiche.
Défaite : Si l'utilisateur n'a plus de tentatives restantes et que le mot n'est pas entièrement deviné, il perd. Le programme révèle alors le mot correct

```
La lettre 'e' apparaît 1 fois.
Mot à deviner : s_ell
Nombre de tentatives restantes : 5
Proposez une lettre : h
Félicitations ! Vous avez trouvé le mot : shell
```

Diagramme explicatif




initialise le mot masqué
proposer une lettre
entrée valide
non
oui
redemander une lettre
oui
lettre déjà proposée
non


oui
lettre déjà proposée
non
oui
lettre correct
reduire nombre vies
mettre a jour le mot masqué
non
encore une vie ?
oui
non
mot deviné ?
defaite
victoire
oui