Projet SU-DOKU

Le projet sera intégralement écrit en **Python**.

Vous utiliserez seulement les bibliothèques standards de Python. Par exemple :

- Vous utiliserez le module **random** pour choisir la grille dans la base de données.
- Vous utiliserez la bibliothèque **tkinter** pour tout le graphisme et l'interface homme-machine.
- Vous utiliserez **SQLite3** pour la base de données.

Une partie de Su-doku » se déroulera de la manière suivante :

- Le joueur choisi le mode de jeu
  - Soit mode invité (pas de statistiques conservées)
  - Soit en mode inscrit. Une inscription est proposée la 1ère fois si le joueur est nouveau, sinon le joueur entre sont pseudo et éventuellement son mot de passe (la gestion d'un mot de passe est facultative)
  - Facultatif : à la demande du joueur, vous affichez les règles du jeu.
- Le joueur choisi la difficulté du jeu, c'est-à-dire :
  - Vous proposerez à minima 3 niveaux : facile, moyen ou expert.
  - On peut demander à refaire une grille effectuée auparavant (ou pas).
  - Facultatif : choisir la durée totale de réflexion pour finir la grille (éventuellement infinie).
  - Facultatif : durée totale de réflexion mesurée ou pas.
- Choix de la grille
  - L'ordinateur choisit au hasard dans la base de données une grille du niveau demandé.
  - Facultatif : l'ordinateur peut construire un « clone » d'une grille déjà existante.

    Un clone est une grille obtenue à partir d'une autre :
    - En permutant les 9 chiffres.
    - En permutant les groupes de 3 lignes (ou 3 colonnes) appartenant à un même carré.

        - En permutant les lignes (ou colonnes) appartenant à un même carré.
    - L'avantage du clone est d'être de même difficulté que l'original mais sans donner une impression de déjà-vu.
- Le joueur remplit la grille en cliquant sur les chiffres puis sur les cases de la grille et enfin valide son choix. Cette Interface homme-machine est uniquement à la souris, c'est à dire sans utilisation du clavier.
- Le joueur peut abandonner à tout moment et obtenir la solution.
- À tout moment le joueur peut revenir en arrière sans tout perdre depuis le début s'il s'aperçoit qu'il a commis une erreur.
- A la demande du joueur, l'ordinateur peut indiquer les chiffres qui ne respectent pas les règles du jeu.
  Soit en signalant qu'il y a un problème, soit en indiquant plus précisément le problème.
- Facultatif : si le temps de réflexion est dépassé la partie s'arrête.
- Si le joueur est en mode inscrit, le programme met à jour les statistiques du joueur dans la base de données.
  Ces statistiques concerneront à minima :
    - La grille qui a été cherchée :
        - Identifiant dans la base.
        - S'agissait-il d'un clone ? Si oui, lequel ? (facultatif)
    - Le résultat à ce jeu :
        - Résultat (réussite ou échec)
        - Temps de réflexion total (facultatif)
        - Note donnée à la performance (facultatif)
    - Vous conserverez tous les résultats d'un même joueur.
    - Facultatif : pour un joueur donné, vous afficherez avec une belle présentation, les résultats des différentes parties effectuées.
- Prolongements (facultatifs)
    - Créer un moyen de rentrer une grille personnelle dans la base de données.
    - Trouver la solution d'une grille et vérifier que la solution est bien unique.
    - Trouver un moyen de déterminer la difficulté d'une grille automatiquement à partir de critères objectifs.

CONSIGNES POUR LE PROJET

- Pour chaque module, indiquer le nom de la personne responsable et la date de la version.
- Conseil : il est prudent d'enregistrer régulièrement votre travail et de ne pas systématiquement écraser vos versions antérieures qui seront conservées sur un support sûr surtout lorsqu'elles sont fonctionnelles.
- Utiliser des noms de variables et de fonctions qui soient descriptifs.

  Par exemple au lieu d'écrire « x = a + b » , choisissez plutôt :

  « nombre_fruit = nombre_pomme + nombre_poire ».
- Suivez au maximum les recommandations de présentation données par : https://pep8.org/
- Décrire dans le doc_string ce que fait chaque fonction.
- Ecrire des commentaires lorsque c'est utile. Les commentaires doivent aider à la compréhension de votre logique. Par exemple :

« compteur += 1 # Ajoute 1 à compteur » Ici, le commentaire ne sert à rien !

« mot == mot[: :- 1] # Teste si mot est un palindrome » Est un commentaire plus utile.

- Regroupez les fonctions qui vont ensemble dans le même module, en revanche séparez dans des modules différents les entités différentes. Par exemple faire un module gérant l'affichage du plateau de jeu, un autre qui s'occupe de la partie logique, etc.
- Mettre dans un module séparé les asserts qui vérifient le bon fonctionnement de vos fonctions.