Travail pratique # 2
Devine qui !

Date de remise : au plus tard le 4 novembre 2022 à 23h59
Pondération de la note finale : 10 %

1 Objectifs

Ce travail a comme principaux objectifs de valider votre compréhension de l'utilisation de **fonctions** en programmation, des **algorithmes complexes**, des **chaînes de caractères**, des **dictionnaires** et de la **lecture/écriture de fichiers**. Vous constaterez que ce travail demande en premier lieu une **compréhension du code** fourni, et par la suite la **réalisation de code** permettant de compléter le programme.

2 Travail à faire

On vous demande de créer une petite intelligence artificielle capable de jouer au célèbre jeu *Guess Who ?*, rebaptisé *Devine qui !* dans le cadre de ce TP.

Ce jeu se joue à deux (un sélectionneur et un devin) et ses règles sont simples. Étant donné un ensemble de personnages prédéterminé, le sélectionneur choisit un personnage de l'ensemble et garde son choix secret. Le devin doit alors poser une série de questions se répondant par oui ou non au sélectionneur quant aux caractéristiques du personnage choisi. Lorsqu'il ne reste qu'un personnage correspondant aux caractéristiques déterminées, le devin demande s'il s'agit bien du personnage sélectionné. Le but du devin est d'identifier ce personnage en posant *le moins de questions possible*, tandis que le sélectionneur a un rôle plutôt passif.

Dans ce travail pratique, le sélectionneur sera l'utilisateur humain du programme, tandis que le devin sera votre intelligence artificielle. En d'autres termes, le programme que vous écrirez demandera à l'utilisateur de choisir un personnage parmi les 24 personnages présentés subséquemment dans cet énoncé, puis lui posera des questions se répondant par oui ou non jusqu'à ce que le programme converge vers un seul personnage possible. Le programme sera conçu pour minimiser le nombre de questions posées.

2.1 Les personnages

Les personnages à deviner dans ce TP sont les mêmes que dans le jeu original, et sont tous représentés dans l'image suivante :

Évidemment, votre intelligence artificielle ne sera pas en mesure de déduire des caractéristiques directement à partir des images, mais elle possède une base de données sous forme de fichier texte (`personnages.txt`) dans lequel chaque personnage est encodé avec une série de caractéristiques. Par exemple, l'entrée de la base de données pour le premier personnage, Alex, est :

```
nom:Alex
genre:homme
accessoires:
cheveux:noirs
```

```
yeux:bruns
nez:petit
pilosite:moustache
```

Chaque personnage est donc identifié par 7 lignes :

- Nom : ne sert pas aux devinettes, mais uniquement à identifier le personnage à la fin.
- Genre : soit homme, soit femme. Le jeu date de 1979 après tout !
- Accessoires : peut être chapeau, bijoux, ou lunettes, une combinaison de plusieurs d'entre eux (séparés d'une virgule), ou aucun (le champ est alors laissé vide)
- Cheveux : il s'agit uniquement de la couleur, et non de la coupe. Les possibilités sont noirs, bruns, blonds, blancs ou roux.
- Yeux : soit bruns, soit bleus. Étrangement personne n'a les yeux verts !
- Nez : petit ou gros.
- Pilosité : peut être barbe, moustache, ou calvitie, une combinaison de plusieurs d'entre eux (séparés d'une virgule), ou aucun (le champ est alors laissé vide)

Un fichier python nommé `personnages.py` permet de transformer la base de données en un dictionnaire python. Vous pouvez déjà exécuter ce fichier individuellement pour voir à quoi ressemble le dictionnaire.

2.2 Le programme principal

Le fichier python `principal.py` sera le point d'entrée du programme, c'est-à-dire le fichier à exécuter pour jouer une partie. Une ébauche d'intelligence artificielle y est déjà commencée : un petit code qui sélectionne une caractéristique, pose une question, puis met à jour les personnages possibles selon la réponse, recommence jusqu'à ce qu'un seul personnage demeure puis écrit le nom du personnage. Toutefois, ce code appelle des fonctions qui ne sont pas encore programmées !

2.3 Votre tâche

Pour faire fonctionner le programme principal, vous devez programmer les 6 fonctions du fichier `devine_qui.py`.

- `types_caracteristiques_ordre_aleatoire` : Cette fonction retourne une liste contenant les types de caractéristiques (genre, cheveux, yeux, etc.), mais chaque fois que la fonction est appelée, l'ordre de la liste doit être aléatoire. Ce hasard permettra de rendre le jeu plus dynamique ; l'ordinateur ne posera pas toujours les questions à score égal dans le même ordre.
- `valeurs_ordre_aleatoire` : Cette fonction retourne, pour un type de caractéristiques donné (cheveux, par exemple), la liste des valeurs possibles pour ce type (blonds, bruns, etc.). Encore une fois l'ordre doit être aléatoire pour rendre le jeu plus dynamique.

- `possede` : Étant donné un personnage et une caractéristique, cette fonction indique si le personnage possède la caractéristique. Par exemple, on retourne vrai pour Alex et moustache, mais faux pour Alfred et yeux bruns.
- `score_dichotomie` : Cette fonction attribue à une caractéristique (par exemple avoir un chapeau) un score qui dépend de combien de personnages ont ou n'ont pas cette caractéristique parmi les personnages *n'ayant pas encore été éliminés*. Le score est le plus élevé lorsqu'il crée une dichotomie (une division du groupe en deux) bien balancée (en d'autres termes, s'il y a environ autant de personnages ayant des chapeaux que de personnages n'en ayant pas), et au plus faible lorsque tous les personnages sont du même côté de la division. Autrement dit, le score mesure l'apport en information qu'on aurait en posant la question concernant cette caractéristique, et permet donc de réduire le nombre de questions.
- `selectionner_caracteristique` : Cette fonction permet de sélectionner la caractéristique présentant le meilleur score de dichotomie, en fonction de l'ensemble des personnages restants.
- `mettre_a_jour_hypotheses` : Cette fonction met à jour les personnages restants, en éliminant ceux qui n'ont pas la caractéristique sélectionnée (si l'utilisateur a répondu oui), ou qui ont la caractéristique (si l'utilisateur a répondu non).

Les détails de chaque fonction, incluant les arguments, les sorties attendues, et les appels à d'autres fonctions qu'il devrait y avoir à l'intérieur, sont inscrits dans la documentation à même le fichier. Des exemples de score de dichotomie y sont également.

Finalement, en exécutant directement le fichier `Devine qui!`, vous pourrez valider vos fonctions avec quelques tests unitaires que nous avons fait pour vous. Attention ! La réussite des tests ne certifie pas que tous les aspects de la fonction sont corrects. Ils sont là uniquement pour vous donner une bonne idée. Les correcteurs feront des tests plus sophistiqués.

3 Les fichiers fournis

Nous vous fournissons plusieurs fichiers :

- **correction.txt**
- **devine_qui.py**
- *personnages.py*
- *personnages.txt*
- *principal.py*
- *visages.jpg*

Les fichiers **en gras** sont les seuls que vous avez à modifier, et les seuls que vous devez remettre. Les fonctions du fichier **devine_qui.py** sont déjà définies et documentées, mais leur code est vide ! C'est à vous qu'il revient de les implémenter. Notez qu'il est interdit de

modifier la définition des fonctions (leur nom, arguments et valeurs de retour), tout comme il est interdit de créer d'autres fonctions. Comme d'habitude, un fichier texte **correction.txt** à remplir avec vos informations est également fourni.

4 Stratégie suggérée

Nous vous suggérons de commencer par (re)lire attentivement cet énoncé pour obtenir une vision de haut niveau de ce que votre programme doit réaliser. Consultez ensuite les fichiers fournis, en commençant par les fonctions déjà implémentées. Cela vous permettra de comprendre comment la tâche à réaliser sera implémentée dans votre code. Finalement, attaquez vous aux fonctions à programmer ; nous vous suggérons de programmer les fonctions dans l'ordre qu'elles vous sont présentées dans le fichier, c'est-à-dire les fonctions dans le haut du fichier en premier.

Testez les fonctions individuellement, en utilisant des tests unitaires lorsque c'est possible. Exécutez finalement le programme principal fourni pour vous assurer que toutes les fonctionnalités sont implémentées correctement.

5 Modalités d'évaluation

Ce travail sera évalué sur 100 points, et la note sera ramenée sur 10. Voici la grille de correction :

Élément	Pondération
Le jeu se déroule fluidement et sans problèmes	10 points
L'intelligence arrive à deviner le bon personnage	10 points
L'intelligence devine le personnage en peu de questions	5 points
Fonction `types_caracteristiques_ordre_aleatoire`	5 points
Fonction `valeurs_ordre_aleatoire`	5 points
Fonction `possede`	10 points
Fonction `score_dichotomie`	20 points
Fonction `selectionner_caracteristique`	10 points
Fonction `mettre_a_jour_hypotheses`	10 points
Respect de la décomposition fonctionnelle demandée	5 points
Qualité du code (noms de variables, style, commentaires, documentation)	10 points

Notez qu'un programme qui n'est pas fonctionnel (qui ne s'exécute pas ou qui plante à l'exécution) pourrait recevoir une note de 0**.**

Votre programme doit être rédigé à même le fichier **devine_qui.py** fourni, que vous devez compresser dans une archive Zip avec le fichier **correction.txt**. Sur le site du cours, dans Contenu et activités, Capsules vidéo complémentaires, une **capsule vidéo** nommée *Démarrer et remettre un TP* a été mise à votre disposition pour vous montrer comment y arriver.

Assurez-vous que vous remettez tous les fichiers nécessaires à l'exécution de votre TP. Nous vous recommandons fortement de créer un **dossier** dans lequel vous mettrez les fichiers relatifs à votre TP, et **rien d'autre**. Nous ne pourrons pas donner de points à votre travail si vous remettez le mauvais fichier. Les **capsules** *Les fichiers sous Windows* et *Les fichiers sous Mac OS* devraient vous aider à bien vous organiser.

6 Ce que vous devez rendre

Vous devez remettre une archive `.zip` d'un **dossier**, contenant **uniquement** les fichiers suivants :

- `devine_qui.py`
- `correction.txt` : Le fichier de correction fourni avec l'énoncé, que vous devez compléter en y indiquant votre nom, votre NI, et le nombre d'heures que vous avez consacré à compléter votre TP.

Cette archive doit être remise via le site web du cours.

7 Remarques

Plagiat : Tel que décrit dans le plan de cours, le plagiat est strictement interdit. Ne partagez pas votre code source à quiconque. Une politique stricte de tolérance zéro est appliquée en tout temps et sous toute circonstance. Tous les cas détectés seront référés à la direction de la faculté. Des logiciels comparant chaque paire de TPs pourraient être utilisés pour détecter les cas de plagiat.

Retards : Tout travail remis en retard peut être envoyé par courriel à l'enseignant si le portail des cours n'accepte pas la remise. Une pénalité de 10% sera appliquée pour un travail remis le lendemain de la remise, puis une pénalité de 25% sera attribuée à un TP remis le surlendemain. Une note de 0 sera attribuée pour un plus long retard.

Remises multiples : Il vous est possible de remettre votre TP plusieurs fois sur le portail des cours. La dernière version sera considérée pour la correction.

Respect des normes de programmation : Nous vous demandons de prêter attention au respect des normes de programmation établies pour le langage Python, notamment de nommer vos variables et fonctions en utilisant la convention suivante : `ma_variable`, `base_de_donnees`, etc. Utiliser **PyCharm** s'avère être une très bonne idée ici, car celui-ci nous donne des indications sur la qualité de notre code (en marge à droite, et souligné).

Bonnes devinettes !