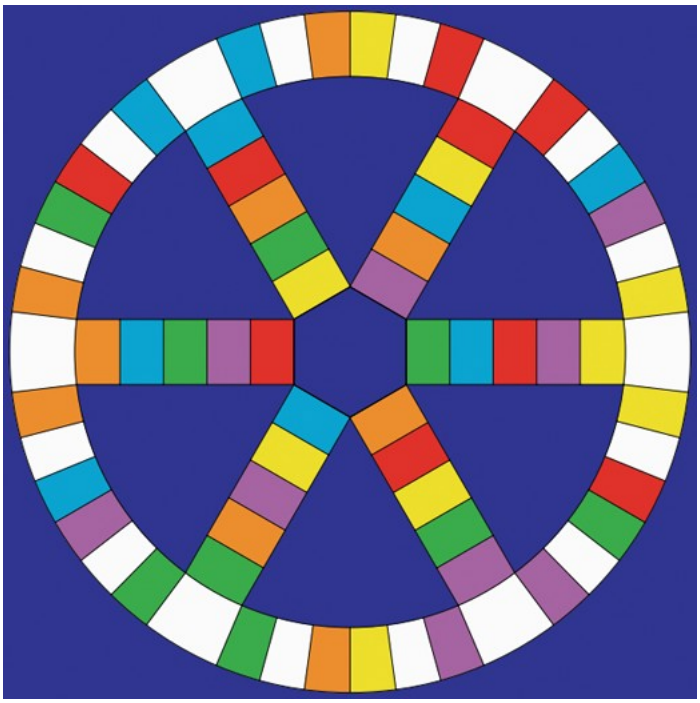
Projet : Trivial Poursuite



Introduction :

Le but du projet est à nouveau de programmer un jeu… Décidément !

Vous avez 4 enseignants qui se relaieront cette semaine pour vous aider, n'hésitez pas à les solliciter.

Le but est de réaliser un jeu du trivial poursuite : pour ceux qui ne connaissent pas le jeu, il s'agit d'un jeu de questions/réponses assez simple contenant 6 catégories de questions :

Règles simplifiées :

Seul les cases extérieures du jeu sont utilisées ; les joueurs commençant sur une des intersections au choix

Les joueurs à chacun leur tour lancent un dé : ils doivent se déplacer du nombre de cases égal au score de leur dé dans un sens ou dans l'autre pour rejoindre la case de leur choix :

- une case blanche, alors le joueur peut relancer le dé et se déplacer à nouveau.

- une case colorée : le joueur doit répondre à la question dans la catégorie que représente la couleur, s'il répond bien, il peut relancer le dé et se déplacer à nouveau.

- une case intersection : il s’agit d’une case colorée particulière qui permet d’obtenir un point dans la catégorie de la case associée.

Lorsqu’un joueur répond mal à une question, c’est au joueur suivant de jouer.

Pour remporter le jeu, les joueurs doivent récupérer un point dans chaque catégorie (dans les 6 intersections).

Règles complètes :

Toutes les cases du jeu sont utilisées : la case centrale est le point de départ. Lorsque tous les points ont été obtenus, il faut rejoindre la case centrale pour répondre à une dernière question.

Éléments à réaliser lors du projet :

Le projet va se dérouler en trois parties :

- la conception des fonctions et d’un modèle des données de votre choix pour faire fonctionner le tout sans graphismes. Vous stockerez dans votre programme, quelques questions / réponses afin de valider le fonctionnement du jeu.

- un affichage graphique.

- la connexion à une base de données qui vous permettra de disposer d’un nombre conséquent de questions / réponses.

Vous trouverez dans la suite de ce document le détail de chacune de ces étapes.

Dates importantes :

Mardi 23 juin au soir : livraison de la première partie du code

Vendredi 26 : interrogation orales de certains élèves (peu entendus ou pour précisions sur le code) pendant la journée

Vendredi 26 au soir : livraison du projet

Première partie :

Vous devez réaliser le jeu avec les règles simplifiées : celles-ci doivent permettre à un joueur de lancer un dé, de choisir une case vers laquelle se déplacer et de lui poser une question en fonction de la catégorie sur laquelle il est tombé.

Quelques indications pour le projet :

Une règle de programmation est à respecter dans votre code : votre programme principal ne doit être constitué que d'un seul appel à une fonction : la fonction jeu(). (cette fonction jeu fera office de programme principal). Cette fonction jeu utilisera ses propres variables et les appels aux fonctions que vous créerez pour faire tourner le programme

1re partie : plateau de jeu et mouvements sur le plateau :

Vous êtes libre de faire ce que vous souhaitez pour représenter les déplacements sur la plateau de jeu. Notez qu'il y a 42 cases organisées en cercle sur ce plateau.

Note pour les points bonus : si le plateau de jeu est facile à représenter pour les règles simplifiées il est plus compliqué pour le règles normales : 6 chemins intermédiaires de 5 cases et une case centrale se rajoutent au plateau.

2ème partie : questions / réponses :

Réalisez une fonction **`poserQuestion(categorie)`** une liste de disctionnaires. Ces dictionnaires devront avoir la forme suivante :

```
{
     "q" : "texte question",
     "rc" : "réponse incorrecte",
     "ri_lst" : ["réponse incorrecte 1", "réponse incorrecte 2", "réponse incorrecte 3"],
}
```

Pour simplifier le programme pour le moment, les questions seront définies dans le programme principal en variable globale, juste avant l'appel de la fonction jeu().

Pour le moment, ne prévoyez qu'une ou deux questions par catégorie : le corpus de questions vous sera fourni à la fin du projet (jeudi soir sur moodle), sous la forme d'une base de données.

Respecter ce format vous simplifiera la tâche lorsque nous vous fournirons la base de données de questions.

3ème partie : nombre de joueurs et score des joueurs

Plusieurs joueurs jouent à tour de rôle et il faut savoir, plus que le nombre de points des joueurs, dans quelle catégorie de questions les joueurs ont eu le point).

Notez que la base de donnée fournie sera en Anglais (si vous souhaitez un peu plus de cohérence vous pouvez réaliser le jeu en anglais également).

Livraison du code de la première partie :

Vous devrez nous rendre mardi soir cette version du trivial poursuite. Votre fichier .py sera à poser sous moodle dans le dépôt prévu à cet effet.

Première partie (bonus):

Si vous souhaitez récupérer quelques points supplémentaires, vous pouvez modifier cette version pour prendre en compte le plateau des règles complètes. Vous pouvez alors nous livrer deux fichiers : la version avec règles simplifiées et la version normale.

Deuxième partie : affichage du plateau et mouvement des pions des joueurs

Plus d’information mercredi

Troisième partie : affichage du plateau et mouvement des pions des joueurs

Plus d’information jeudi soir

Annexe :

Aide pour l'écriture de code

Voici un mauvais exemple à ne pas reproduire :

```
import modulecool

def fonction1(param):
    maVariable = 1
    resultat = modulecool.nde(1, param)
    fonction2(resultat)
truc = "***"
def fonction2(param):
    print("Voici le résultat avec de jolies étoiles autour:")
    print(truc)
    print(param)
    print(truc)
fonction1(truc)
```

Plusieurs raisons à ça :

- un bout de programme principal traîne entre les définitions de fonction1 et fonction2 (à éviter : c'est le mal!)

- des fonctions qui s'appellent fonction, des variables qui s'appellent mavariable et des paramètres qui s'appellent param : c'est le meilleur moyen pour garder un programme illisible ces noms doivent avoir un sens.

- la fonction2 utilise la variable truc : il vaut mieux l'utiliser comme un paramètre de la fonction (et la renommer également)

- pas de commentaires !

- le code n'est pas aéré

Voici quelques règles simples à respecter pour éviter à vos correcteurs de saigner des yeux et de vous retirer des points de style :

1. placer vos imports en haut du code, suivi de vos def, suivi de votre programme principal. Si vous avez bien suivi, pour ce projet, le programme principal ne contiendra que quelques lignes.

2. gardez la même convention pour le nommage de toutes vos variables et noms de fonctions. La mode python et de séparer les mots par des souligné (underscore) : par exemple :
   - `def concatener_trois_chaines(...)`
   - `nombre_habitants = 127340`
3. commentez les lignes un peu compliquées par un commentaire (derrière un #) et commenter TOUTES vos fonctions avec une docstring (les trois guillemets). Voici un exemple pour ce dernier :

```
def reserver_un_gite(numero_client,numero_gite) :

"""crée une réservation pour le gîte et envoie une facture au client
       numero_client -- le numéro du client pour lequel la facture est créée
       numero_gite – le numéro du gîte réservé"""
```

4. soyez précis sur les noms de variables si possible (ce n'est pas toujours possible) : préférez **`valeur_entiere`** à **`valeur`** par exemple (et je ne veux pas voir de bidule ou de machin !)
5. Aérez votre code et essayez de respecter les mêmes règles tout au long de votre programme : par exemple utilisez un saut de ligne entre chaque fonction. Évitez d'avoir des sauts de lignes n'importe ou dans votre code, cela ajoute à la complexité de le lire.

   De même, sur une même ligne, vous pouvez aussi ajouter des espaces autour des opérateurs mathématiques, la deuxième ligne sera ici un peu plus lisible :
   - `(5+numerateur+i)*2/(10+denominateur)`
   - `(5 + numerateur + i) * 2 / (10 + denominateur)`