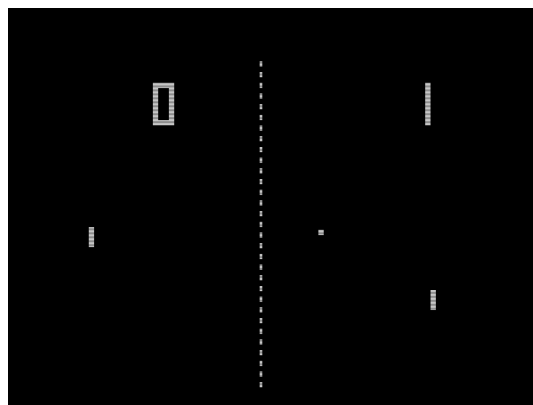


Un peu d'histoire

Parmi celles-ci, Atari sortira alors la première borne d'arcade Pong et suite au succès immédiat de celle-ci, produira alors la console de jeu éponyme.



Structure d'un jeu interactif

1. Déclaration des constantes ;
2. Initialisation de l'interface ;
3. Chargement des ressources ;
4. Initialisation des variables d'état ;
5. Boucle principale :
 - (a) Récupération des évènements ;
 - (b) Logique du jeu ;
 - (c) Affichage.
6. Fermeture de l'interface.

Le Pong Classique

Vous disposez du fichier `pong_classique.py`. Celui-ci contient une esquisse du jeu final respectant la structure ci-dessus.

Vos Missions

Mission 1

Compléter la partie du code correspondant à la gestion du déplacement vertical de la raquette de droite. On utilisera les flèches haut et bas (`K_UP` et `K_DOWN`).

Mission 2

Compléter la partie du code correspondant à la gestion du débordement vers le haut et le bas de chaque raquette.

Mission 3

Compléter la partie du code correspondant à la gestion du rebond de la balle sur la raquette de droite.

Mission 4

Compléter la partie du code correspondant à la gestion du rebond de la balle sur le sol.

Mission 5

Compléter la partie du code correspondant à la gestion de la collision de la balle avec les bords gauche et droit de la fenêtre.

ATTENTION : Il faudra dans ce cas que la balle reparte du milieu de l'écran.

Mission 6

Compléter la partie du code correspondant à la gestion du score.

Mission 7

Compléter la partie du code correspondant à la gestion de l'affichage du vainqueur.

On considèrera que le vainqueur sera le premier joueur à atteindre 11 Points.

Soyons créatifs

Mission 8

Modifier le code afin que la raquette de gauche soit bleue et celle de droite rouge.

Mission 9

Modifier le code afin que la vitesse de la balle augmente au cours du temps.

Mission 10

Modifier le code avec des fonctionnalités originales (raquettes, balles, effets...).

Un nouveau jeu de Pong : Le Pong Spatial.

Le jeu à développer ici est basé sur la chute d'astéroïdes sur la Terre.

La balle est désormais un astéroïde qui va s'écraser sur la Terre, les raquettes sont des vaisseaux qui peuvent se déplacer verticalement pour frapper les astéroïdes et les anéantir avant qu'ils ne frappent le sol de notre planète.

Les deux joueurs font désormais équipe pour protéger notre planète. Les astéroïdes arrivent de l'espace (haut de l'écran). Il y en aura 10 par partie (pluie d'astéroïdes). A chaque frappe d'un vaisseau la taille de l'astéroïde diminue, jusqu'à sa disparition ou son arrivée au sol. Le joueur peut changer la direction de cet astéroïde en fonction de l'endroit où celui-ci tape sur son vaisseau, alors que sa vitesse verticale, vers le bas, augmente graduellement au cours de la manche à cause de l'effet de la gravité terrestre.

Au début du jeu, on attribue à la planète un score de 10.

Si un astéroïde touche le sol avant d'avoir disparu ou s'il sort de l'écran (à gauche ou à droite) la manche se termine et le score de 10 attribué à la planète diminue de 1.

Si un astéroïde disparaît, le score de la planète ne change pas. Si, après la pluie des 10 astéroïdes, le score de la planète n'est pas 0, alors la planète est sauvée !

Tâche 1

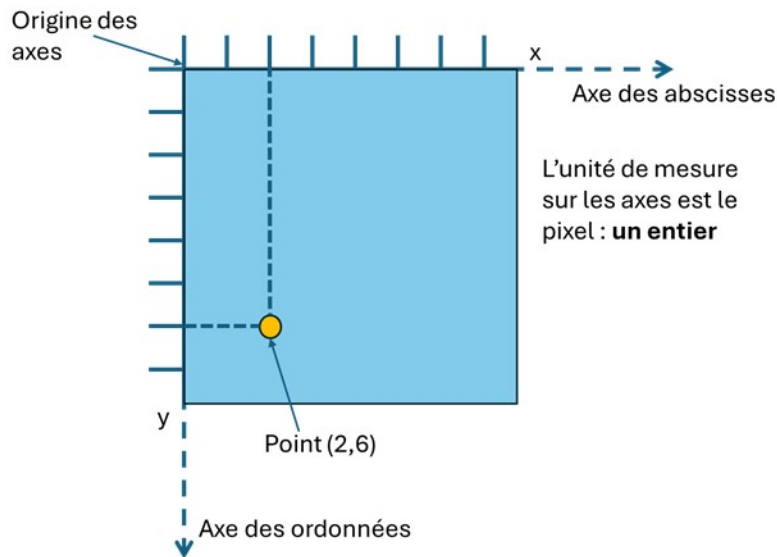
Modifier et compléter le code Python fourni pour gérer les différentes fonctionnalités du jeu du Pong Spatial.

Tâche 2

Automatiser l'obtention des meilleurs joueurs par un programme.

Tâche 1 : Le codage des fonctionnalités

Pour rappel avec Pygame, l'écran dans lequel se déroule le jeu est le suivant :



L'astéroïde qui apparaît dans l'écran de jeu a les attributs initiaux suivants :

- `asteroide_rayon = 40` (rayon du cercle initial de l'astéroïde en pixels);
- `asteroide_x = largeur // 2` (position du centre x);
- `asteroide_y = hauteur // 2` (position du centre y);
- `asteroide_speed_x = 4 * random.choice([-1, 1])` (vitesse aléatoire en x (-4 ou 4));
- `asteroide_speed_y = 2 * random.choice([-1, 1])` (vitesse aléatoire en y (-2 ou 2));

Sa vitesse verticale vers le bas augmente en fonction du coefficient de gravité terrestre qui a pour valeur 0.1 :

```
coef_grav = 0.1 # coefficient de gravite
```

Les vaisseaux spatiaux des deux joueurs sont modélisés et codés par des rectangles de couleurs différentes placés sur les côtés verticaux de l'écran, avec les attributs `vaisseau_largeur`, `vaisseau_hauteur`.

Le dictionnaire `etat` servira à stocker les valeurs des différentes variables qui sont modifiées au cours du jeu.

Un fichier de code Python à compléter vous est fourni, avec l'écran de jeu.

Lancer une première fois le jeu : vous allez constater que les rebonds sur les vaisseaux ne sont pas implémentés et que l'astéroïde ne vient pas de l'espace (haut de l'écran).

Vos missions

Mission 1

Modifier le code pour que l'astéroïde vienne de l'espace, sa position horizontale pourra être aléatoire.

Mission 2 : Arrière-plan avec les étoiles et nébuleuses

Écrire une fonction `dessin_etoiles_fond` qui prend en paramètre une liste de tuples de coordonnées (x,y) d'étoiles générées aléatoirement et permettant de faire apparaître sur le fond noir initial des cercles blancs représentant un ciel étoilé.

Mission 3 : Gestion d'un astéroïde

Déplacement, taille et affichage : compléter la partie du code concernée dans `pong_spatial.py` afin de gérer les rebonds sur les vaisseaux. La taille de l'astéroïde est divisée par 2 à chaque impact sur un vaisseau. On pourra utiliser les fonctions codées précédemment si nécessaire. Lorsque la taille de l'astéroïde atteint 5 pixels ou moins, celui-ci n'est plus dessiné).

Mission 4

Modifier le code pour prendre en compte le coefficient de gravité terrestre : ce coefficient appelé `coef_grav` sera appliqué à la vitesse l'astéroïde afin que celle-ci augmente lors de sa chute.

Mission 5

Dessiner à main levée la planète sauvée sur une feuille libre. Expliquer en quelques lignes, sous le dessin, comment construire le dessin et proposer un algorithme pour le réaliser sur une feuille.

Mission 6

Programmer l'image de la mission 5 avec turtle ou pygame. Sur la feuille libre, comparer et commenter votre production et votre dessin à la suite de votre algorithme. Quelles difficultés avez-vous rencontrées ? Pourquoi ?

Mission 7

Écrire une fonction d'affichage de fin de jeu et de réussite éventuelle de la mission avec affichage de la planète créée.

Tâche 2 : Automatiser l'obtention des meilleurs joueurs par un programme.

A la fin de chaque partie les résultats ont été enregistrés dans un fichier `parties_pong_spatial.csv`.

Votre objectif

Créer un programme Python pour gérer et analyser les caractéristiques des joueurs du Pong spatial (scores, âges, niveaux de difficulté, etc.) à partir des fichiers CSV fournis.

Vous disposez pour cela de deux fichiers csv :

- le fichier `joueurs_Pong` rassemble les caractéristiques de chaque joueur inscrit au jeu :
 - `id_joueur` : un entier, identifiant unique du joueur ;
 - `nom` : une chaîne de caractères ;
 - `prenom` : une chaîne de caractères ;
 - `nom_joueur` : une chaîne de caractères au format première lettre du prénom accolée au nom complet ;
 - `sexe` : le caractère 'F' ou 'H' ;
 - `date_naissance` : une chaîne de caractères au format année-mois-jour ('aaaa-mm-jj') ;
 - `date_inscription` : une chaîne de caractères au format année-mois-jour ('aaaa-mm-jj').
- Le fichier `parties_Pong` rassemble les données de chaque partie
 - `id_partie` : un entier, identifiant unique de la partie ;
 - `id_joueur1` : un entier, identifiant unique du joueur ;
 - `id_joueur2` : un entier, identifiant unique du joueur ;
 - `id_gagnant` : un entier, indiquant l'identifiant du joueur gagnant ;
 - `score` : un entier, le score de la partie ("Facile" : 10 points, "Moyen" : 40 points, "Difficile" : 100 points) ;
 - `niv_difficulte` : une chaîne de caractères ("Facile", "Moyen", "Difficile").

Le programme que vous allez coder devra présenter et afficher un menu pour choisir la gestion et/ou l'analyse à effectuer avec les différents outils suivants :

Mission 1 : Lecture de fichiers csv

Écrire le code Python de la fonction `ouverture_lecture` ayant en entrée le nom du fichier et permettant de : lire un fichier CSV et de retourner une liste de dictionnaires.

Mission 2

`conversion_attributs` : cette fonction a en entrée une liste de dictionnaires et une liste des attributs qui doivent être présentés sous la forme d'entiers.

Écrire le code permettant de convertir en nombres entiers, les attributs présentés sous forme de chaîne de caractères (id, age, score, etc.) et qui renvoie une liste de dictionnaires.

Mission 3

`ajout_joueur` : cette fonction ayant en paramètres d'entrée la liste des dictionnaires de joueurs `liste_joueurs`, et les attributs du nouveau joueur (`nom_nouv`, `prenom_nouv`, `sexe_nouv`, `date_naissance_nouv`) ajoute ce nouveau joueur à la fin la liste des dictionnaires `liste_joueurs`.

Mission 4

`suppression_joueur` : cette fonction (ayant en paramètres d'entrée `nom`, `prenom`, `date_naissance` du joueur à supprimer, et les listes des dictionnaires `liste_joueurs`, `liste_parties`) supprime ce joueur de la liste des joueurs ainsi que les parties qu'il a faites de la liste des parties.

On prendra soin de :

- supprimer un joueur dans la `liste_joueurs` ;
- Supprimer automatiquement toutes les parties où il apparaît :
 - en `id_joueur1` ;
 - en `id_joueur2`.

Mission 5 : Enregistrement des listes de dictionnaires dans des fichiers csv

La fonction `ecriture_sauvegarde` ayant en paramètre d'entrée fichier (le nom du fichier avec son extension), `liste_colonnes` (les noms des colonnes), `liste_donnees` (la liste des dictionnaires) et enregistre les entêtes des colonnes et les données dans fichier.

Mission 6 : fusion des deux listes `liste_joueurs` et `liste_parties`

La fonction `fusion_tables` ayant en paramètres les noms des deux listes à fusionner créera une nouvelle liste de dictionnaires `liste_parties_avec_gagnants` contenant toutes les informations des parties ainsi que les informations des joueurs gagnants. Cette liste sera enregistrée sous un nouveau fichier `parties_avec_gagnants`.

Mission 7

`meilleurs_joueurs` : cette fonction renvoie la liste des 5 meilleurs joueurs avec leurs informations et leur score total.
Elle utilisera en entrée la liste fusionnée `liste_parties_avec_gagnants`.