Jeu en POO :
Space Invaders

Situation :

Votre vaisseau spatial rencontre une flotte de vaisseaux extraterrestres qui s'avance vers lui. Il doit détruire ces vaisseaux en tirant des balles pour les faire exploser.

Objectifs :

- Prendre en main la bibliothèque Pygame.
- Comprendre les éléments d'un code proposé.
- Utiliser la programmation orientée objet pour créer les éléments d'un jeu.

GitHub Classroom :

Les codes de démarrage à modifier sont disponible sur GitHub classroom.
Si ce n'est pas déjà fait, créer un compte GitHub puis choisissez votre nom dans la liste de noms disponibles à cette adresse :
Vous obtenez un espace personnel où vous pourrez déposer tous vos codes (avec un système de versioning pour conserver toutes les versions une à une). C'est là que je pourrais observer vos codes et votre avancée.

Impératifs techniques :

Toutes les fonctions et méthodes développées dans ce projet devront comporter des **spécifications claires**. Pour rappel, les spécifications permettent de savoir ce que fait une méthode grâce à l'emploi de triples guillemets.

Exemple :

```
def ajoute(x: float, y: float) -> float:
    """ ajoute fait la somme d'un flottant x et d'un flottant y """
    return x+y
```

ou

```
def complex(real = 0.0, imag = 0.0):
    """Crée un nombre complexe.

    Paramètres d'entrée :
    real -- la partie réelle (default : 0.0)
    imag -- la partie imaginaire (default : 0.0)
    """
```

Mission 1 : Comprendre le code proposé

Le fichier “space_invader_1.py” propose quelques éléments de code.

En vous aidant de la documentation du site de pygame : https://www.pygame.org/docs/ , répondez aux questions suivantes dans votre compte rendu.

1) Création de la fenêtre :

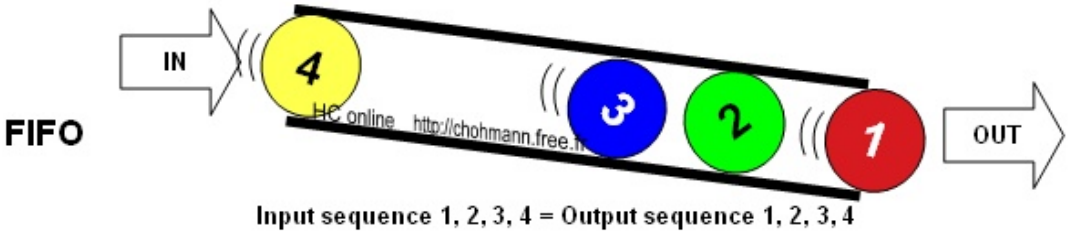
- Que permet de faire la fonction **`pygame.display.set_mode()`** ? (ligne 10)
- Que permet de faire la fonction **`pygame.display.set_caption()`** ? (ligne 11)

- Que permet de faire la fonction **`pygame.image.load()`** ? (ligne 13)
- Quelle instruction permet d’afficher une image sur l’écran ? (ligne 18 et suivante)
- Quels sont les formats d’images supportés par pygame ?

Corriger les erreurs du fichier.

2) Gestion des événements :

Un événement peut être une entrée clavier (soit l’appui, soit le relâchement d’une touche), le déplacement de votre souris, un clic… Il faut savoir que chaque événement créé est envoyé sur une file (ou queue), en attendant d’être traité, comme lorsque l’on fait la queue au supermarché.



La ligne 23 crée une boucle qui va parcourir cette file d’événements, puis tester chaque événement enregistré en interrogeant son type (lignes 24, 29, 31 et 33)

Quelques questions :

- A quel événement correspond le type **`pygame.QUIT`** ? (ligne 24)
- A quel événement correspond le type **pygame.KEYDOWN** ? (ligne 29)

- Quel est le type d’événement correspondant à l’appuie sur la flèche droite ?
- Quel est le type d’événement correspondant au relâchement d’une touche ?

- A quoi sert la variable running définie ligne 16 ?

Mission 2 : Déplacement du vaisseau

Le vaisseau du joueur se déplace horizontalement, en bas de l’écran (800 x 600) à 500 pixels du haut. Ce déplacement est réalisé par les flèches droite et gauche.
Quand la flèche droite est pressée, le vaisseau part vers la droite et ne s’arrête qu’au bord de l’écran. Il change de sens lorsque la touche gauche est pressée.
Visualiser la vidéo SpaceInvadersMission2.m4v

Ouvrir le fichier “space_invader_2.py”. Dans ce fichier, nous allons devoir importer les objets créées dans le fichier “space.py” sur lequel vous allez travailler.

- Ajouter une instruction dans l'en-tête de "space_invader_2.py" afin d'importer space.py comme un module.

Dans “space_invader_2.py”, le joueur est initialisé par l’instruction **`player = space.Joueur()`** ,
Expliquer la présence du mot **`space`**.
De quelle classe l’objet **`player`** est-il l'instanciation ?
À première vue, quels sont les attributs de l’objet **`player`** utilisé dans le fichier “space_invader_2.py” ?
Quelles méthodes de l’objet **`player`** sont utilisées dans le fichier “space_invader_2.py” ? Précisez quelle méthode est un mutateur et quelle méthode est un accesseur.

Coder maintenant la classe Joueur dans le fichier “space.py” pour que tout fonctionne.

Mission 3 : Tir de balles

Le joueur dispose d’une balle attachée à son vaisseau, quand il tire avec la touche Espace, la balle monte verticalement, Quand elle atteint le haut de l’écran, elle réapparaît dans le vaisseau. Elle n’est pas disponible pendant qu’elle monte.
Visualiser la vidéo SpaceInvadersMission3.m4v

Le fichier “space_invader_3.py” contient le moteur de jeu permettant au joueur de tirer.
Il reste à **implémenter** une classe d’objets balle() à ajouter au fichier “space.py”.

Balle
tireur : Joueur depart : int hauteur : int image : pygame.image etat : string "chargee", "tiree"
bouger()

L'interface de la classe, écrite sous la forme d'un tableau regroupant attributs (en haut) et méthode (en bas) est la suivante :
La variable de type Joueur correspondant au tireur est transmise lors de l’initialisation de la Balle.

Elle possède 5 attributs :
- **`tireur`** = la variable Joueur qui a tiré la Balle
- **`depart`** = la position horizontale où la Balle a été tirée
- **`hauteur`** = la position verticale de la Balle
- **`image`** = l’image crée par pygame à partir du fichier “balle.png”
- **`etat`** = “chargee”, la Balle est prête ; “tiree”, la Balle a été tirée et se déplace verticalement.

Elle dispose d’une seule méthode[1] :
- bouger() pour se déplacer quand elle a été tirée

On aimerait bien que l'objet Joueur puisse tirer ! Ajoutez une méthode **`tirer()`** à l’objet Joueur.
Souvenez-vous, lorsque l'on tire, le vaisseau s'arrête.

Mission 4 : Apparition des ennemis :

Les ennemis apparaissent en haut de l’écran, dans une position aléatoire horizontalement.
Puis ils descendent verticalement.
Il y a deux types d’ennemis, qui ont des images et des vitesses différentes.

---

[1] Pour des raisons pratiques, nous décidons de ne pas encapsuler nos variables. Vous pouvez toutefois utiliser des getters et des setters pour corriger cela.

Comme dans la mission précédente, il s'agit de partir du fichier "space_invader_4.py" dans lequel le moteur du jeu est programmé.

Les modifications ont été effectuées ligne 20 à 23 (création des ennemis) et lignes 56 à 59 (déplacement et dessin des ennemis), sans respecter le principe de l'encapsulation.

Il reste à ajouter au fichier "space.py" une classe Ennemi dont on précise l'interface ci-contre.

Ennemi
depart : int hauteur : int type : int image : pygame.image vitesse : int
avancer()

Elle dispose d'une variable de classe NbEnnemis dénombrant le nombre d'ennemis affichés.

Mission 5 : Destruction des ennemis :

Quand un ennemi est touché par la balle, il disparaît et un nouveau vaisseau apparaît en haut de l'écran.La balle retourne dans le vaisseau du joueur. Le score du joueur est augmenté d'un point.

Seules les lignes 50 à 54 ont été ajoutées dans le fichier "space_invader_5.py" (remarquez à quel point la programmation est proche du langage naturel.

Il vous reste à ajouter, dans le fichier "space.py" :

- à la classe Balle une méthode **`toucher(ennemi)`**
- à la classe Ennemi une méthode **`disparaitre()`**
- à la classe Joueur un attribut score et une méthode **`marquer()`**

Pour résumer, les trois classes ont les propriétés suivantes (éventuellement d'autres que vous avez ajouté).

Joueur
position : int image : pygame.image sens : string "droite", "gauche", "O" score : int
deplacer() tirer() marquer()

Balle
tireur : Joueur depart : int hauteur : int image : pygame.image etat : string "chargee", "tiree"
bouger() toucher(Ennemi)

Ennemi
depart : int hauteur : int type : int image : pygame.image vitesse : int
avancer() disparaitre()

Mission 6 : Améliorations personnelles

Ce jeu n'est pas parfait ! A vous de choisir quelques améliorations de ce jeu et de les développer.

Exemple :

Pour les images du jeu, on pourra : soit choisir les images proposées, soit utiliser des images libres de droits, soit fabriquer ses propres images.
Le score devrait pouvoir s'afficher dans le jeu.
Le principe de l'encapsulation n'est pas respecté. Faites le !
Peut-on faire du multi-joueur ?