PYGAME:

Pygame est une bibliothèque libre multi-plateformes pour Python qui permet de créer des programmes en gérant tous l'aspect multimédia. Il permet de réaliser une grande variété de jeu 2D : gestion des images, des sons, interaction avec un clavier et une souris.

Pygame est composé de plusieurs modules:

Cdrom Accéder et contrôler les lecteurs CD audio.	**Cursors** Charger et compiler des images de curseur.	**Display** Configurer la surface d'affichage.
Draw Dessiner des formes simples comme des lignes et des ellipses sur des surfaces.	**Event** Gérer les évènements à partir de différents matériels d'entrée (clavier, souris…), ainsi que du fenêtrage.	**Font** Charger et faire un rendu des polices TrueType.
Image Charger, sauver et transférer des images sur des surfaces.	**Joystick** Gérer les joysticks.	**Key** Gérer le clavier.
Mixer Charger et jouer des sons.	**Mouse** Gérer la souris et son affichage.	**Movie** Lire des vidéos à partir de film en MPEG.
Music Jouer des pistes musicales en streaming.	**Overlay** Accéder à l'affichage vidéo avancé.	**Pygame** Fonctions de haut niveau pour le contrôle de Pygame.
Rect Conteneur flexible pour un rectangle.	**Sndarray** Manipuler des échantillons de données audio.	**Sprite** Objets de haut niveau pour la représentation des images de jeu.
Surface Objets pour des images et l'écran.	**Surfarray** Manipuler des images avec Numeric.	**Time** Manipuler les temporisateurs et le taux de rafraîchissement.
Transform Redimensionner et déplacer des images.		

Pour importer tous les modules, on utilise: from pygame import*.

1. Création d'une fenêtre et insertion d'une image de fond:

```
from pygame import*

#initialisation de tous les modules
init()

# creation de la fenêtre
fen=display.set_mode((800,501))

# charger une image
fond=image.load("ciel.jpg")

#coller l'image
fen.blit(fond,(0,0))

#rafraîchir l'affichage
display.flip()
```

2. Insertion d'une deuxième image:

Modifiez le programme en rajoutant la partie du code encadrée ci-dessous :

```
from pygame import*

#initialisation de tous les modules
init()

# creation de la fenêtre
fen=display.set_mode((800,501))

# charger une image
fond=image.load("ciel.jpg")

#coller l'image
fen.blit(fond,(0,0))

# charger une deuxième image
im=image.load("vaisso.png")

#coller l'image
fen.blit(im,(230,120))

#rafraîchir l'affichage
display.flip()
```

3. La boucle infinie :

Le langage Python étant un langage scripté, une fois que le traitement de tout votre script est effectué, le programme s'arrête. Il faut donc créer une boucle infinie pour empêcher que le programme s'arrête.

```
from pygame import*

#initialisation de tous les modules
init()

# creation de la fenêtre
fen=display.set_mode((800,501))

# charger une image
fond=image.load("ciel.jpg")

#coller l'image
fen.blit(fond,(0,0))

# charger une deuxième image
im=image.load("vaisso.png")

#coller l'image
fen.blit(im,(450,350))

#rafraîchir l'affichage
display.flip()
```

```
#Boucle infinie
boucle=True
while boucle:

    #liste d'evenements
    for evenement in event.get():

        #appui sur la croix
        if evenement.type==QUIT:
            boucle=False
quit()
```

4. Gestion des événements et mouvements d'images:

✓ Pour interagir avec pygame on utilise des évènements qui peuvent être un clic de la souris, un appui sur une touche etc….

✓ Ces évènements sont stockés dans une file qui s'appelle **event**. On peut avoir le type de l'événement grâce à la commande **event.type**.

✓ Avec des instructions conditionnelles, on peut gérer les actions correspondant aux différents événements.

✓ L'interception des différents événements se fait dans la boucle infinie (scrutateur d'événements).

- **Exemple 1**:clic sur la croix pour fermer la fenêtre

- **Exemple 2**:appui sur une touche pour déplacer une image

```
from pygame import*

#initialisation de tous les modules
init()

# creation de la fenêtre
fen=display.set_mode((800,501))

# charger une image
fond=image.load("ciel.jpg")

#coller l'image
fen.blit(fond,(0,0))

# charger une deuxième image
im=image.load("vaisso.png")
x=450  ; y=350

#coller l'image
fen.blit(im,(x,y))

#rafraîchir l'affichage
display.flip()

#Boucle infinie
boucle=True
while boucle:

    #liste d'evenements
    for evenement in event.get():

        #appui sur la croix
        if evenement.type==QUIT:
            boucle=False

        if evenement.type==KEYDOWN:
            if evenement.key==K_DOWN:
                y=y+5
                fen.blit(fond,(0,0))
                fen.blit(im,(x,y))
    display.flip()
quit()
```

1. Modifiez et complétez le code précédent comme indiqué ci-contre:
2. Complétez le programme pour que l'on puisse déplacer le vaisseau dans toutes les directions en utilisant les touches : ***K_UP , K_RIGHT , K_LEFT***

3. Ajoutez l'instruction suivante avant la boucle infinie et notez son rôle : ***key.set_repeat(2,2)***

❖ **Exemple 3**: clic et mouvement de la souris :

En vous inspirant des codes ci-dessous, modifiez ou complétez le programme précédent pour que le vaisseau :

1. se positionne à l'endroit du clic de la souris

2. Suit le mouvement de la souris dans la fenêtre

```
#Clic de la souris:
if evenement.type==MOUSEBUTTONDOWN:#type d'evenement clic souris
    if evenement.button==1:#bouton gauche
        print('(x,y)=',evenement.pos)#coordonnées du clic
        print('x=',evenement.pos[0])#abscisse du clic
        print('y=',evenement.pos[1])#ordonnée du clic

#Position de la souris dans la fenêtre:
if evenement.type==MOUSEMOTION:
     print('(x,y)=',evenement.pos)#coordonnees (x,y)
```

Défi :

Réalisez un programme qui permet :

- ✓ le déplacement automatique du vaisseau
- ✓ le changement de direction de celui-ci après appui sur une touche
- ✓ le rebond de celui-ci lorsqu'il atteint les limites de la fenêtre.

1. **Gestion du son:**

Pygame dispose de deux modules séparés permettant de jouer des sons : le module *Mixer* et le module *Music* .

Utilisation de Mixer:

✓ Mixer est le module général utilisé pour la gestion des sons , par exemple les bruitages d'un jeu vidéo avec Pygame.

```
from pygame import*

init()
#création de la fenêtre
fen=display.set_mode((500,500))
fen.fill((255,255,255))

#chargement de l'image
balle=image.load("ball.png")
x,y=210,210
fen.blit(balle,(x,y))

#création de l'objet sound
son=mixer.Sound("cri.wav")

#lecture du son
son.play()

display.flip()
```

Pour arrêter la lecture d'un son, on utilise la méthode "stop()"

```
#arrêt de lecture
son.stop()
```

Pour suspendre la lecture des sons, on utilise l'instruction suivante:

```
#pause de lecture
mixer.pause()
```

Pour reprendre la lecture des sons, on utilise l'instruction suivante:

```
#reprise de la lecture
mixer.unpause()
```

Pour arrêter la lecture de tous les sons, on utilise l'instruction suivante:

```
#arrêt de lecture de tous les sons
mixer.stop()
```

Utilisation de Music

✓ Music est un module particulièrement adapté pour la gestion des musiques (gros fichiers), car il possède des fonctionnalités différentes, comme le streaming (le chargement au fur et à mesure du fichier audio), qui permet de lancer la musique avant la fin de son chargement.

```
from pygame import*


init()
#création de la fenêtre
fen=display.set_mode((500,500))
fen.fill((255,255,255))

#chargement de l'image
balle=image.load("ball.png")
x,y=210,210
fen.blit(balle,(x,y))
display.flip()

#chargement d'un son
mixer.music.load("cri.wav")

#lecture du son
mixer.music.play()
```

✓ Pour arrêter la lecture de la musique, on utilise l'instruction suivante:

```
#arret de lecture
pygame.mixer.music.stop()
```

✓ Pour suspendre la lecture d'une musique, on utilise l'instruction suivante:

```
#pause de lecture
mixer.music.pause()
```

✓ Pour reprendre la lecture d'une musique, on utilise l'instruction suivante:

```
#reprise de la lecture
mixer.music.unpause()
```

Le défi:

1. Créer un programme qui lance un son quand on appui sur la barre Espace, qui le met en pause quand on la relâche, et qui le stoppe quand on appuie sur Entrée.