# Space Invader

Réalisation d'un petit jeu de tir en Python avec la librairie PyGame,
avec déplacement d'un vaisseau de gauche à droite via les flèches du clavier
Tir via la touche espace Avec un ennemi qui décent de l'écran et un système de score

## Technique installation de python et pygame

https://www.python.org/ python3 -m pip install -U pygame

installation de VSCode https://code.visualstudio.com/

- avec PyGame sur VSCode:
    - Press CTRL+SHIFT+P in visual studio
    - Choose "Preferences: Open Settings (JSON)"
    - Add "python.linting.pylintArgs": ["--generate-members"] below one of the lines (put a comma if necessary)
    - Save the .json file (CTRL+S)
- installation de python + mise à jour de PIP (python -m pip install --upgrade pip)
- installation de pyGame
    - pip install pygame

**pour trouver des images: https://www.flaticon.com/**

## Mise en place de la boucle de jeu avec PyGame

Le code de la boucle de jeu est le suivant.

```
import pygame
#Initialisation de pygame
pygame.init()

# Constantes du jeu
WIDTH = 800 # largeur de la fenetre de jeu
HEIGHT = 600 # hauteur de la fenetre de jeu

# Variable du jeu

# Fenetre de jeu
screen = pygame.display.set_mode((WIDTH,HEIGHT))
pygame.display.set_caption("Space Invader")
icon = pygame.image.load('PyGameModel/ufo.png')
pygame.display.set_icon(icon)

#boucle de jeu
```

```
running = True
FPS = 60
clock = pygame.time.Clock()
while running:
    clock.tick(FPS)
    screen.fill((100,0,0))
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
    pygame.display.update()
```

## Ajout d'un joueur (Player) vaisseau/fusée

ajout des attributs (propriétés) et de la fonction (methode) pour la fusée

```
# Mise en place du joueur "class" (player)
playerImg = pygame.image.load('PyGameModel/player.png')
playerX = 370
playerY = 480
playerXMove = 0
playerYMove = 0

def player(x,y):
    screen.blit(playerImg,(x,y))
```

affichage de la fusée dans la boucle de jeu:

player(playerX,playerY)

### ajout de la capture de la pression des touches

pygame.KEYDOWN pygame.KEYUP pygame.K_LEFT pygame.K_RIGHT

Voici le code à ajouter dans la boucle de jeu:

```
        if event.type == pygame.KEYDOWN:
            print("touche préssée")
            if event.key == pygame.K_LEFT:
                print("gauche")
                playerXMove = -1
            if event.key == pygame.K_RIGHT:
                print("droit")
                playerXMove = 1
        if event.type == pygame.KEYUP:
            print("touche relachée")
            if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
                playerXMove = 0
    playerX += playerXMove
    if playerX <= 0:
```

```
        playerX = 0
    if playerX >=736:
        playerX = 736
    player(playerX,playerY)
    # mise a jour de la fenetre
    pygame.display.update()
```

# Ajout de l'ennemi (Enemy)

## Ajout des "enemy" avec une position random

nous avons besoin d'importer le module random

```
import random
# ajout dans la section enemy
enemyX = random.randint(0,800)
enemyY = random.randint(50,150)
```

## Mouvement de l'enemy dans la boucle de jeu

```
    # ajout des bornes de l'écran
    if enemyX <=0:
        enemyXChange = 0.15
        enemyY += enemyYChange
    elif enemyX >= 736:
        enemyXChange = -0.15
        enemyY += enemyYChange
```

## Ajout du tir depuis le vaisseau

```
# ajout de la class
# Bullet (~Class)
bulletImg = pygame.image.load('bullet.png')
bulletX = 0
bulletY = 480
bulletXChange = 0
bulletYChange = 1
bulletState = "ready"  #ready on ne vois pas la balle, fire la balle est tirée

def fireBullet(x, y):
    global bulletState
    bulletState = "fire"
    screen.blit(bulletImg, (x + 16, y + 10))

# à ajouter au niveau du captage des touches
```

```
            # lancement de la balle
            if event.key == pygame.K_SPACE and bulletState is "ready":
                bulletX = playerX
                fireBullet(bulletX,bulletY)

# à ajouter au niveau de la boucle de jeu

    # Ajout du mouvement de la balle
    if bulletY <= 0:
        bulletState = "ready"
        bulletY = 480
    if bulletState is "fire":
        fireBullet(bulletX,bulletY)
        bulletY -= bulletYChange
```

### Detection de la colision entre la balle et l'enemy

nous allons rechercher la distance entre la balle et l'enemy
puis comparer cette distance aec une valeur de 20
equation de la distance D entre 2 points A et B A(xA,yA) et B (xB, yB)

D = (racine carré de) $((xB-xA)^2 + (yB-yA)^2)$ // pythagore

ajout d'une fonction isCollision() import math

```
# ajout d'une fonction pour vérifier si il y collision entre l'enemy et la balle
def isCollision(enemyX,enemyY,bulletX,bulletY):
    distance = math.sqrt(math.pow(enemyX - bulletX,2) + math.pow(enemyY -
bulletY,2))
    if distance < 20:
        return True
    else:
        return False

# à ajouter dans la boucle de jeu
    #collision
    collision = isCollision(enemyX,enemyY,bulletX,bulletY)
    if collision:
        bulletState = "ready"
        bulletY = 480
        score += 1
        print (score)
        enemyX = random.randint(0,800)
        enemyY = random.randint(50,150)
```

## Ajout d'une gestion de score

On ajouter une fonction d'affichage du score

```
score = 0
# Police de caratère
SCORE_FONT = pygame.font.SysFont('comicsans', 40)

# fonction affichage du score

def showScore():
    text = SCORE_FONT.render(str(score), 1, BLACK)
    screen.blit(text, (30, 30))
```

Dans la gestion de la colission on ajouter +1 à la variable score.

```
    #collision
    collision = isCollision(enemyX,enemyY,bulletX,bulletY)
    if collision:
        bulletState = "ready"
        bulletY = 480
        score += 1
        enemyX = random.randint(0,800)
        enemyY = random.randint(50,150)
```

ajouter dans la boucle de jeu l'appel à la fonction d'affichage du score.

```
    showScore()
```