1 TD 2 : Deux girafes en danger : utilisation de PyGame

(correction page ??)

Deux girafes sont en danger car un avion ne peut les éviter (voir figure 1). Votre mission est, si vous l'acceptez, de ralentir l'arrivée du bolide en frappant le plus vite possible alternativement les flèches gauche et droite. Le meilleur joueur est celui qui aura amené l'avion le plus bas possible. L'explosion de l'avion sera annoncée à grands bruits. L'objectif du TD est de découvrir le module *PyGame* qui permet facilement de programmer un jeu, comment savoir qu'une touche a été pressée, comment déplacer un avion sans altérer le fond de l'écran, comment jouer des sons, ... Voilà quelques questions abordées durant ce TD.

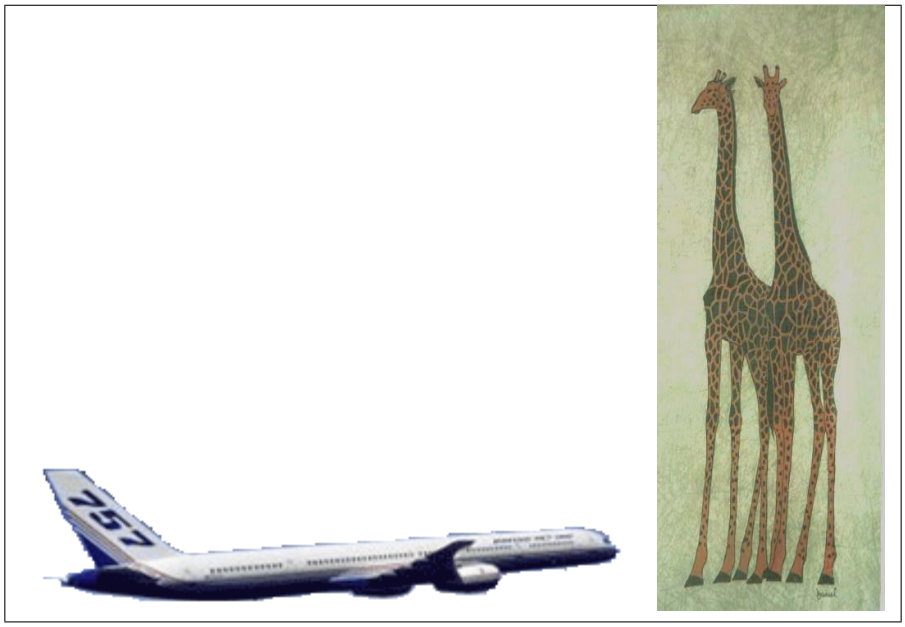

Figure 1 : *Deux girafes en danger à l'approche d'un avion.*

Le programme final nécessite quelques images et sons qu'il faut télécharger depuis l'adresse http://www.xavierdupre.fr/enseignement/td_python/python_td_simple/index.html.

1) Le module PyGame facilite la création d'un jeu. En quelques lignes, il est possible de créer une fenêtre et d'y placer une image, ce que font les instructions suivantes.

```
import pygame
pygame.init ()
image  = pygame.image.load ("girage.jpg")
sizeim = image.get_size ()
size   = (sizeim[0]*3, sizeim[1])
screen = pygame.display.set_mode (size)
screen.blit (image, (0,0))
pygame.display.flip ()
```

Plus précisément, l'image `girafe.jpg` est lue et conservée en mémoire, on récupère ses dimensions puis on crée une fenêtre de jeu trois fois plus large que l'image. Toutefois le programme précédent

s'exécute si rapidement qu'il est impossible de voir quoique ce soit. Pour cela, il faut appeler la fonction suivante à la fin du programme. D'après son nom, on peut deviner ce qu'elle est censée faire.

```
def attendre_clic (screen,x,y):
    while True:
        for event in pygame.event.get():
            if event.type == pygame.MOUSEBUTTONUP :
                return None
```

La première question du TD, sera de modifier le programme pour placer les deux girafes à l'autre bout de la fenêtre.

2) On voudrait maintenant afficher l'image `ciel.jpg` de façon à ce qu'elle apparaisse en dessous des girafes. Pourquoi ne pas faire un petit tour sur Internet pour découvrir le rôle de l'instruction `pygame.display.flip()`, la recherche des mots-clé *pygame* et *pygame.display.flip* sur *Google* devrait vous y aider.

3) Au tour de l'image `avion.PNG` d'apparaître. Le résultat est-il vraiment satisfaisant ? Ce serait sympathique de faire disparaître tout ce blanc, un nouveau petit tour par Internet et l'utilisation de la méthode `set_colorkey` avec comme argument $(255, 255, 255)$.

4) On sait maintenant afficher toutes les images, les placer là où on en a envie. Il ne reste plus qu'à les animer. Le code suivant permet de faire patienter le programme pendant 100 millisecondes.

```
    import time
    time.sleep (0.1)
```

On veut maintenant, à l'aide d'une boucle, faire bouger l'avion depuis le coin supérieur gauche jusqu'aux girafes, et lorsque l'avion arrive à l'autre bout de la fenêtre, le programme doit s'arrêter.

5) Maintenant, lorsque l'avion atteint les girafes, on veut afficher une explosion, celle de l'image `explosion.jpg`.

6) Et pour faire plus réaliste, une explosion sonore serait la bienvenue. Le code suivant permet de jouer un son :

```
son = pygame.mixer.Sound ("bomb.wav")
son.play ()
```

Plusieurs sons sont proposés, `bomb.wav`, `explosion.wav`, `explosion_2.wav`, `missile.wav`, `sheep.wav`, `toilet_flush.wav`, `toilet_flush_2.wav`. A vous de composer votre petite explosion en sélectionnant le son qui vous plaît ou ceux qui vous plaisent tout en sachant que le code suivant produira non pas deux explosions consécutives mais deux explosions simultanées.

```
son = pygame.mixer.Sound ("bomb.wav")
son.play ()
son = pygame.mixer.Sound ("bomb.wav")
son.play ()
```

7) Question facultative... le code suivant permet d'afficher du texte à l'écran.

```
    font = pygame.font.Font ("freesansbold.ttf", 15)
    text = font.render ("message en blanc à la position 100,100", True, (255,255,255))
    screen.blit(text, (100,100))
    pygame.display.flip ()
```

Internet ou le chargé de TD vous donneront le sens de chaque instruction. Libre à vous de vous en servir pour décorer votre fenêtre de jeu.

8) Tous les éléments graphiques ou sonores vous sont désormais connus, il ne reste plus qu'à faire en sorte que le programme réagisse en fonction des touches pressées par le joueur. La fonction suivante retourne une chaîne de caractères si une des touches suivantes est pressée : flèche haut, flèche bas, flèche droite, flèche gauche, ECHAP.

```
def attendre_touche () :
    for event in pygame.event.get():
        if event.type == pygame.MOUSEBUTTONUP : return "clic"
        elif event.type == pygame.KEYDOWN :
            if event.key == 275 : return "right"
            elif event.key == 276 : return "left"
            elif event.key == 273 : return "up"
            elif event.key == 274 : return "down"
            elif event.key == 27 : return "quit"
            else :
                pass
                #print "key ", event.key # on imprime quand on ne connaît pas le code des touches
        else :
            #print event
            pass
    return ""
```

Il ne reste plus qu'à insérer cette fonction dans le programme pour permettre au joueur de ralentir l'inexorable course de l'avion vers les deux girafes. A vous de décider quelle touches permettront d'influencer sa course.