Programmer le jeu « cupidon » :

I- Récupération des images et du début de code :

- ✓ **Ouvrir un dossier** dans votre espace personnel situé sur le réseau ou bien sur une clef USB. Vous déposerez toutes les images ainsi que votre fichier python dans ce dossier. Il est important que tous les documents soient enregistrés dans le même dossier.

- ✓ **Récupérer sur classroom** les différentes images ainsi que le fichier python : **cupidon_1** dans lequel il y a le début du code.

- ✓ **Comprendre** et s'approprier le code pour pouvoir mieux faire la suite

II- Animer cupidon pour qu'il batte des ailes :

Dans cette partie, on veut construire la fonction **battement_ailes** ci-dessous en faisant appel aux images de cupidon de 0 à 3, s'il regarde vers la droite, ou de 13 à 16 s'il regarde vers la gauche (utiliser la variable globale **direction**). Il s'agira d'une fonction récursive, c'est-à-dire d'une fonction qui s'appelle elle-même. Ainsi, elle s'exécute en boucle.

```
def battement_ailes(i):
    """Fonction qui prend en paramètre un compteur.
    La fonction modifie l'image de cupidon en fonction du compteur pour qu'il batte des ailes"""


    # on rappelle la fonction battements_ailes au bout de 100 millièmes de secondes
    # pour que l'animation se produise en boucle
    toile.after(100, battement_ailes,i+1)
```

Remarques :

- ✓ Les différentes images sont dans la liste **images_cupidon** déjà créée dans le code du fichier **cupidon_1**.
- ✓ Les quatre images de direction « droite » (ou « gauche ») doivent changer à chaque appel de la fonction **battement_ailes** et tourner ainsi en boucle. On utilisera pour cela le compteur **i** et l'instruction **%** qui permet de récupérer le reste de la division euclidienne d'un entier par un autre.
  Pour que l'image de cupidon change, on utilisera l'instruction suivante :
  toile.itemconfigure(objet à changer , image = nouvelle image)
- ✓ Ne pas oublier dans le programme principal, d'appeler une première fois cette fonction pour initialiser l'animation.

III- Faire bouger cupidon à l'aide des quatre flèches du clavier :

Dans cette partie, on veut construire la fonction **pression_touche** ci-dessous, elle sera appelée dès qu'une touche du clavier sera enfoncée.

```
def pression_touche(event):
    """fonction appelée dès qu'il y a une pression sur une touche du clavier.
    Elle prend en paramètre l'évènement survenu """
    global x_cupidon, y_cupidon, direction
    # ces trois variables globales peuvent être modifiées dans cette fonction
    touche=event.keysym     # on récupère la touche appuyée
```

Remarques :

- ✓ Pour que la fenêtre récupère les « évènements » clavier et fasse appel à notre nouvelle fonction, il faut ajouter l'instruction suivante en fin de programme (juste avant le fenetre.mainloop() comme ci-dessous) :

```
fenetre.bind('<Key>', pression_touche)
# permet de réagir en cas de clic sur une touche du clavier
# la fonction pression_touche est alors automatiquement appelée
fenetre.mainloop()  # permet à la fenêtre "d'écouter" les évènements en boucle
```

- ✓ Le « nom » des quatre touches du clavier avec une flèches sont : *'Left'*, *'Rigth'*, *'Up'* et *'Down'* (Il faut les guillemets !).
- ✓ Pour déplacer un élément de la fenêtre, on utilisera l'instruction suivante :
  toile.coords(objet à bouger , nouvelles coordonnées)
  (ici il y a deux coordonnées : abscisse et ordonnée)
- ✓ On prendra soin que cupidon ne puisse pas sortir de la fenêtre.

IV- Tirer des flèches avec cupidon :

Pour que cupidon tire une flèche, on utilisera la touche espace qui se nomme *'space'*.

Dans cette partie, nous aurons besoin de quatre nouvelles variables globales : **fleches**, **action, fleche_gauche, fleche_droite** :

- ✓ **fleches** sera une liste contenant les informations concernant chaque flèche déjà lancée. Une flèche pourra être représentée par une liste contenant quatre éléments la caractérisant : sa direction, ses deux coordonnées, son image dans la fenêtre (voir ci-dessous). **fleches** sera donc une liste de listes.
  L'image de chaque flèche dans la fenêtre sera :
  toile.create_image(abscisse, ordonnée, image = image à afficher)
- ✓ **action** pourra prendre deux valeurs : *'tir'* ou *'passif'* suivant que cupidon soit en train de tirer ou non. Cette variable globale sera initialisée à *'passif'*.
- ✓ **fleche_gauche** et **fleche_droite** seront les deux images de flèches. Utiliser les deux instructions suivantes pour définir ces deux variables :

```
img_fleche_g=Image.open('Fleche1.png')
fleche_gauche =ImageTk.PhotoImage(img_fleche_g, master=fenetre)
```

Construire ensuite les fonctions suivantes :

```
def tirer(i):
    """fonction récursive qui prend en paramètre un compteur.
    La fonction va faire défiler les images de cupidon en train de tirer une flèche.
    Elle va également ajouter une flèche à la liste des flèches déjà tirées"""
    global action


def avancer_fleches():
    """fonction récursive qui permet de faire avancer les flèches qui ont déjà été tirées,
    la fonction supprime ensuite les flèches qui sont sorties de la fenêtre"""
```

Remarques :

- ✓ La fonction **tirer** ne sera récursive que le temps de faire défiler les 4 images de cupidon en train de tirer. Elle devra ensuite relancer le battement des ailes de cupidon.
- ✓ La fonction **tirer** sera appelée dans la fonction **battement_ailes** qui doit alors cesser sa « récursivité » le temps que cupidon tire sa flèche. On utilisera la variable globale **action**.
- ✓ Pour supprimer une flèche, on utilise l'instruction : **toile.delete(objet à supprimer)**. Pour supprimer un élément d'une liste, on utilise : **del(liste[indice de l'élément à supprimer])**.
- ✓ La fonction **avancer_fleches** sera appelée dans le programme principal. Elle s'appellera ensuite de façon récursive et de manière « infinie ».