Exercices d'informatique - Série n° 2

Cours 4INOC01 Série distribuée le 28.9.2017

1. La fonction *sleep()*.

Déterminer l'action de la fonction sleep() du module time en exécutant le script suivant:

```
1 import time
2 x=raw_input("Tapez un texte: ")
3 time.sleep(12.34)
4 print x
```

2. Animation.

Rédiger un script Python qui affiche le texte "Programmer avec Python" en colonne. Les lettres doivent apparaître toutes les 0.5 secondes.

3. Processus légers ou *threads*.

Dans le script donné ci-dessous, deux classes sont créées à partir de la classe Thread() du module threading. Déterminer l'effet de la méthode start() et des méthodes getName(), isAlive() et _Thread__stop(). **NB**: Pour arrêter un script dont des processus ne sont pas terminés, taper dans un terminal la commande `killall python`.

```
1  import threading, time
2  class job1(threading.Thread):
3      def run(self):
4          for l in "Programmer avec Python":
5              time.sleep(0.2)
6              print l#, self.atester.isAlive()
7  class job2(threading.Thread):
8      def run(self):
9          for n in xrange(0,20,1):
10             time.sleep(0.1)
11         print "Fin"
12         #self.astoper._Thread__stop()
13 j1=job1()
14 j2=job2()
15 j1.atester=j2
16 j2.astoper=j1
17 j1.start()
18 j2.start()
19 #print j1.getName()
20 #print j2.getName()
```

4. Chronomètre.

Créer un chronomètre avec Python. Le chronomètre s'enclenche et s'arrête en appuyant sur la touche "enter". Le temps est donné en minutes, secondes et dixièmes. L'exécution du script donne par exemple:

```
Start:
Stop:
1 ' 7 '' 0
```

5. La valse des monstres.

Créer une classe Monstre contenant une méthode constructeur attribuant à chaque monstre un attribut couleur, un attribut coordonnees et un attribut canevas. La méthode constructeur crée une ligne dans le canevas selon l'attribut coordonnees. La classe Monstre doit également contenir une méthode deplacement(x,y) qui translate le monstre en plaçant le premier point de coordonnees au point (x,y).

Créer également un processus Objet qui instancie un objet de la classe Monstre et qui le fait tourner en rond sur le cercle d'équation $(x-200)^2+(y-200)^2=100^2$ dans le sens trigonométrique à partir d'un angle initial self.init.

Le script donné ci-dessous doit produire deux monstres qui tournent en rond (voir figure 1).

```
import Tkinter
cadre=Tkinter.Tk()
cadre.geometry("600x400")
canevas=Tkinter.Canvas(width=600,height=400)
canevas.place(x=0,y=0)
bernard=Objet()
bernard.cadre=canevas
bernard.couleur='red'
bernard.forme=[50,100,10,10,50,50,100,10,50,100]
bernard.init=0
bernard.start()
gustave=Objet()
gustave.cadre=canevas
gustave.couleur='blue'
gustave.forme=[50,10,10,100,50,50,100,100,50,10]
gustave.init=pi/4
gustave.start()
cadre.mainloop()
```

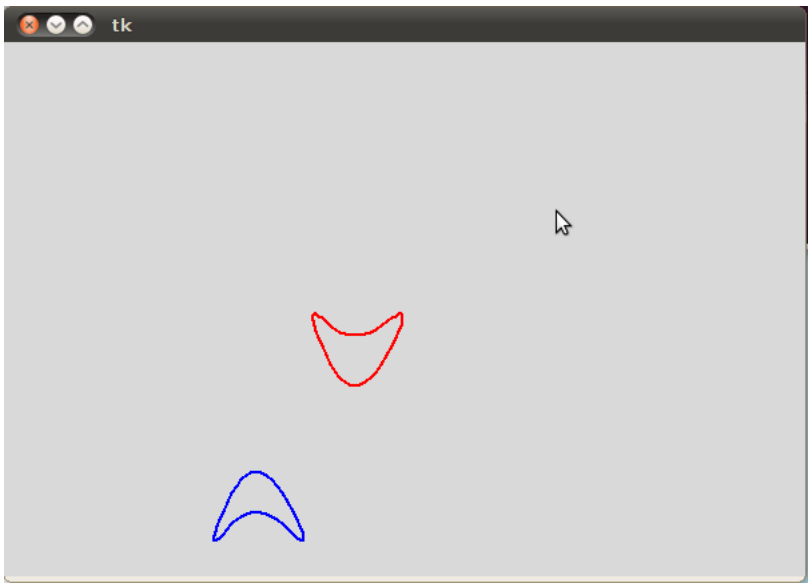

FIGURE 1. Exercice 5

6. Flocons.

Ecrire un script *Python* qui crée 100 "flocons", à raison de un flocon par dixième de seconde (voir figure 2). Chaque flocon est créé en haut du cadre et tombe à vitesse v. La vitesse v, la position horizontale et la couleur des flocons sont aléatoires.

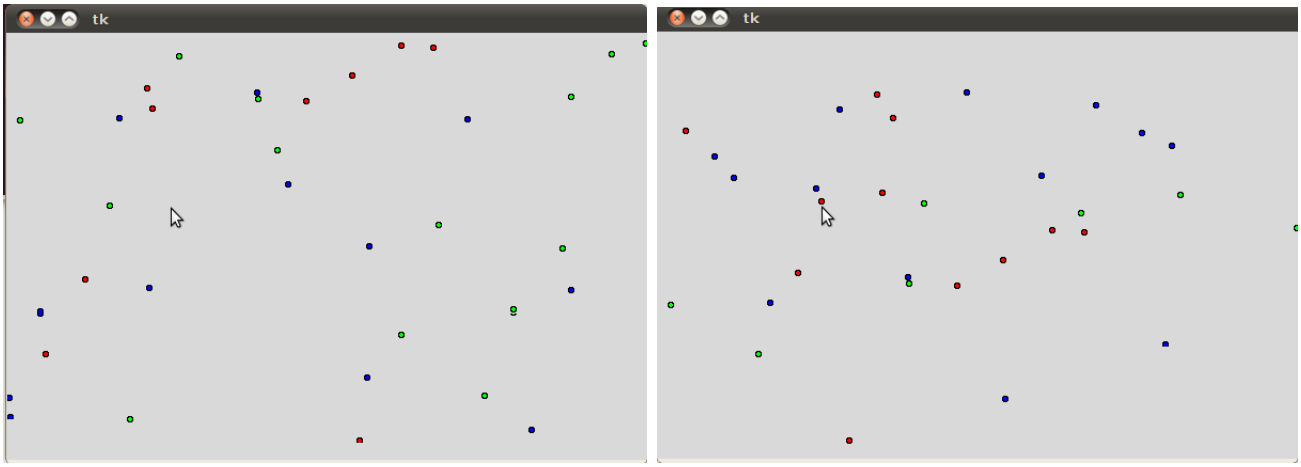

FIGURE 2. Exercice 6

7. Une particule.
Créer un code *Python* qui affiche un cadre carré à l'intérieur duquel se déplace une particule (voir figure 3). Elle se déplace le long de trajectoires parallèles aux diagonales du carré à vitesse constante. Quand elle touche un bord, elle rebondit selon la loi de la réflexion. La vitesse, la couleur et le point de départ de la particule sont aléatoires. Ajouter une option qui permet à l'utilisateur d'arrêter le mouvement en appuyant sur la touche `s` et de le faire redémarrer en appuyant sur une autre touche.

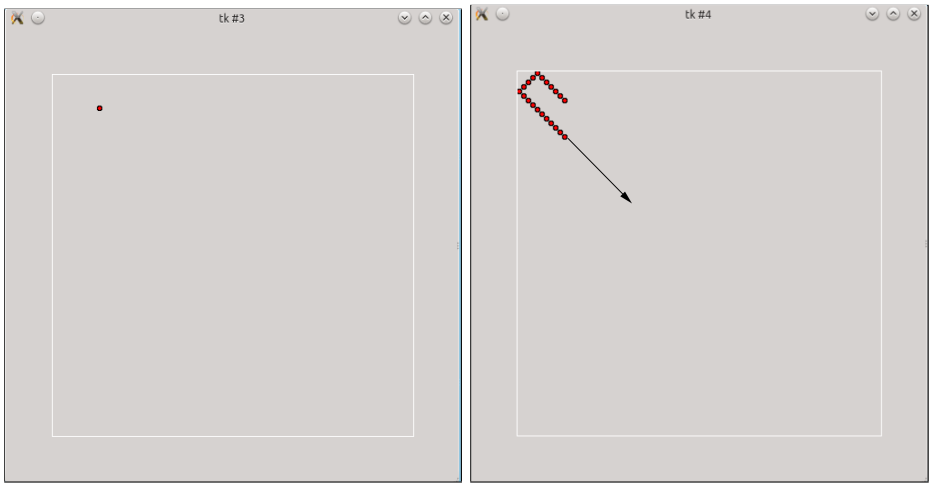

FIGURE 3. Exercice 7

8. Particules.
Rédiger un script *Python* qui crée 100 particules (chaque dixième de seconde). Les particules évoluent sans interagir dans une boîte carrée et rebondissent contre les parois selon la loi de la réflexion. Elles se déplacent à vitesse constante le long de trajectoires parallèles aux diagonales

du carré. La vitesse, la couleur et le point de départ de chaque particule sont aléatoires. Ajouter une option permettant de stopper le processus en appuyant sur la touche `s` et de le redémarrer en appuyant sur une autre touche.

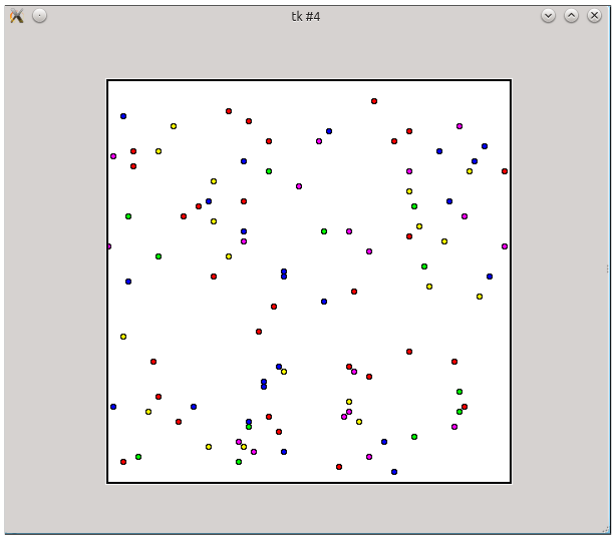

FIGURE 4. Exercice 8

9. Particules.
Rédiger un script *Python* qui crée 100 particules (chaque dixième de seconde). Les particules évoluent dans une boîte carrée et rebondissent contre les parois selon la loi de la réflexion. Elles se déplacent à vitesse constante le long de trajectoires parallèles aux diagonales du carré. La vitesse, la couleur et le point de départ de chaque particule sont aléatoires.

Les particules interagissent de la manière suivante: quand deux particules sont proches (environ moins de 10 pixels) elles se figent pour tout le reste du processus (voir figure 5). **Indication:** avant de bouger, chaque particule doit "tester" la position de toutes les autres particules.

Ajouter une option permettant de stopper le processus en appuyant sur la touche `s` et de le redémarrer en appuyant sur une autre touche.

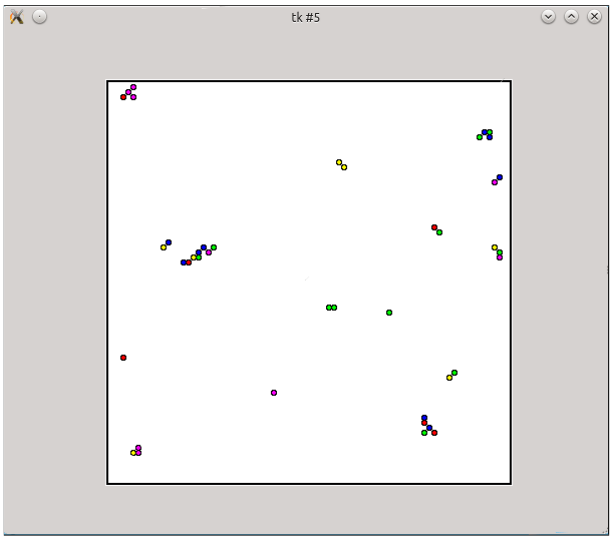

FIGURE 5. Exercice 9