Chapitre 12

Course d'escargots

12.1. Nouveaux thèmes abordés dans ce chapitre

- Animation automatique
- Animation manuelle
- Événements

12.2. Règles du jeu

Quatre escargots font une course sur une ligne droite. Le premier arrivé a gagné !

On pourrait imaginer des faire des paris sur le vainqueur. Cela sera laissé en exercice. Le but de ce programme est de voir comment on peut faire bouger des objets à l'écran.

Ici, la petite difficulté est de faire en sorte que le vainqueur ne soit pas toujours le même... La solution est assez simple : on va tirer au hasard un nombre entre 1 et 4. L'escargot portant ce numéro avancera d'un cran, tandis que les autres resteront sur place.

12.3. Programme « Course d'escargots »

course.py

```
# La course d'escargots

from tkinter import *
from time import sleep
from random import randint

def go():
    global x1, x2, x3, x4
    y = 1000
    while x1<960 and x2<960 and x3<960 and x4<960:
        no = randint(1,4) # no de l'escargot qui va avancer
        if no==1:
            x1 += 1
            can.coords(esca1, x1, y1)
            y = y1
        elif no== 2:
            x2 += 1
            can.coords(esca2, x2, y2)
            y = y2
        elif no== 3:
            x3 += 1
            can.coords(esca3, x3, y3)
```

```
            y = y3
        else:
            x4 += 1
            can.coords(esca4, x4, y4)
            y = y4
        sleep(0.01)  # délai de 1/100 de secondes
        can.update()
    can.coords(vainqueur, 500, y)

def reinit():
    global x1, x2, x3, x4, y1, y2, y3, y4
    x1, y1  = 130, 75
    x2, y2  = 130, 225
    x3, y3  = 130, 375
    x4, y4  = 130, 525
    can.coords(esca1, x1, y1)
    can.coords(esca2, x2, y2)
    can.coords(esca3, x3, y3)
    can.coords(esca4, x4, y4)
    can.coords(vainqueur, 200, 800)   # non visible car en dehors du cadre.

x1, y1 = 130, 75
x2, y2 = 130, 225
x3, y3 = 130, 375
x4, y4 = 130, 525

fen = Tk()
fen.title("Course d'escargots")
can = Canvas(fen, width=1100, height=600, bg ='white')
can.pack(side=TOP, padx=5, pady=5)
b1 = Button(fen, text='Nouvelle course', width=15, command=reinit)
b1.pack(side=LEFT)
b2 = Button(fen, text='Partez !', width=15, command=go)
b2.pack(side=LEFT)
b3 = Button(fen, text='Quitter', width=15, command=fen.quit)
b3.pack(side=RIGHT)
# décor
can.create_line(1080, 0, 1080, 600, width=5, fill="red")
can.create_line(250, 0, 250, 600, width=5, fill="green")
can.create_line(0, 150, 1100, 150, width=5, fill="black")
can.create_line(0, 300, 1100, 300, width=5, fill="black")
can.create_line(0, 450, 1100, 450, width=5, fill="black")
# images
photo1 = PhotoImage(file='escargot1.gif')
esca1 = can.create_image(x1, y1, image=photo1)
photo2 = PhotoImage(file='escargot2.gif')
esca2 = can.create_image(x2, y2, image=photo2)
photo3 = PhotoImage(file='escargot3.gif')
esca3 = can.create_image(x3, y3, image=photo3)
photo4 = PhotoImage(file='escargot4.gif')
esca4 = can.create_image(x4, y4, image=photo4)
photo5 = PhotoImage(file='vainqueur.gif')
vainqueur = can.create_image(200, 800, image=photo5) # en dehors de l'image

fen.mainloop()
fen.destroy()
```

Ce programme n'est pas difficile à comprendre. On voit que les déplacements se font avec l'instruction `can.coord()`. En fait, on change simplement les coordonnées de l'objet.

Il ne faut pas oublier l'instruction `can.update()`, faute de quoi on ne verra que la situation au départ et à l'arrivée de la course. Essayez !

Exercice 12.1

Dans la version d'origine du programme, chaque escargot a la même probabilité de gagner. On voudrait maintenant tricher un peu, de sorte que plus le numéro de dossard est élevé, plus la chance de gagner est grande. Mais faites en sorte que chaque escargot puisse gagner quand même...

Exercice 12.2

Écrivez un programme qui simule la chute libre d'une balle (presque) parfaitement élastique, sans frottement. Il faudra gérer les rebonds contre le sol et les murs.

Rappelons les équations du mouvement, pour deux dimensions, dans le repère utilisé par *tkinter* (à savoir l'origine en haut à gauche de la fenêtre et l'axe y orienté vers le bas) :

$$x(t+\Delta t) = x(t) + v_x \cdot \Delta t$$
$$y(t+\Delta t) = y(t) + v_y \cdot \Delta t$$
$$v_y(t+\Delta t) = v_y(t) + 9.81 \cdot \Delta t$$

$(x(t)\ ;\ y(t))$ sont les coordonnées au temps t de l'objet
v_y et v_y sont les vitesses selon les axes x et y
Δt est le pas de temps (choisir une valeur « petite » : 0.01 ou moins)

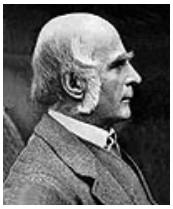
Exercice 12.3

Ajoutez une deuxième balle au programme de l'exercice 12.2 et gérer les chocs entre les deux balles, que l'on supposera (presque) parfaitement élastiques.

Pour la théorie sur les chocs élastiques, voir https://fr.wikipedia.org/wiki/Choc_élastique

Exercice 12.4

Une **planche de Galton** est un dispositif inventé par Francis Galton. Des clous sont plantés sur la partie supérieure de la planche verticale, de telle sorte qu'une bille lâchée sur la planche passe soit à droite soit à gauche d'un clou, avec la même probabilité. Dans la partie inférieure, les billes sont rassemblées dans des casiers en fonction du nombre de passages à gauche et de passages à droite qu'elles ont fait.

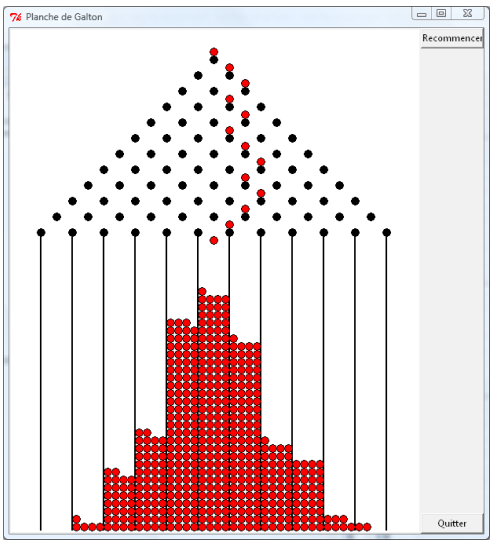

Francis **Galton**
(1822-1911)

Faites une animation d'une planche de Galton. Le mouvement des billes ne sera pas soumis aux lois de la physique, mais à celles des probabilités. Pour décider si la bille passera à droite ou à gauche du clou, utilisez le hasard.

On aimerait voir le chemin parcouru par chaque boule. Aussi, les boules laisseront une « trace », comme sur l'image ci-contre.

Les boules seront rangées dans les casiers par ligne de quatre boules.

Faites en sorte que l'on puisse choisir le nombre de rangées de clous.



Exercice 12.5

Modifiez l'exercice 11.6 pour faire une animation : on aimerait voir les bâtons de l'histogramme s'allonger sous nos yeux.

À chaque lancer, il faudra seulement modifier le bâton correspondant, et non pas redessiner tout l'histogramme.

Exercice 12.6 : l'horloge de la gare de Saint-Gall

Depuis 2018 se trouve sur la façade de la gare de Saint-Gall une horloge binaire géante :

Il est 07:20:40

La première ligne donne les heures, la deuxième les minutes et la dernière les secondes.
Programmez cette horloge de sorte qu'elle indique l'heure actuelle. On la verra donc fonctionner en direct.

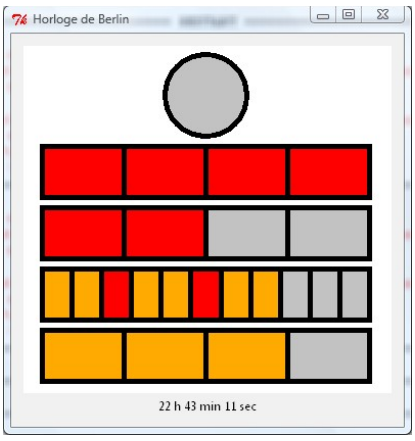
Indications

- Il faut importer le module `time`.
- L'instruction `h,m,s = time.localtime()[3:6]` récupère l'heure, les minutes et les secondes actuelles.

Exercice 12.7 : l'horloge de Berlin

La « Mengenlehreuhr » ou « horloge de Berlin » est la première horloge publique au monde qui a donné l'heure au moyen de rectangles lumineux.

Elle a été inventée par un certain Dieter Binninger et elle fut installée en 1975 à Berlin. Elle se trouve au Europa Center (Wittenbergplatz)



Chaque rectangle allumé indique qu'un certain temps s'est écoulé. Plus précisément :

- chaque lumière de la première ligne (celle du haut) représente 5 heures ;
- chaque lumière de la deuxième ligne représente 1 heure ;
- chaque lumière de la troisième ligne représente 5 minutes (les lumières rouges indiquent les quarts d'heure) ;
- chaque lumière de la dernière ligne représente 1 minute ;
- le disque au-dessus de l'horloge clignote : il est allumé les secondes paires et éteint les secondes impaires.

Programmez cette horloge de sorte qu'elle indique l'heure actuelle. On la verra donc fonctionner en direct.

12.5. Gestion manuelle des mouvements

Il est aussi possible de contrôler les mouvements d'un objet via le clavier ou la souris. Cela se fait grâce à l'instruction `bind`. Par exemple, `fen.bind("<z>", action)` appellera la procédure `action` quand la touche Z sera pressée. Comme vous le voyez, les événements (par exemple presser la touche Z) sont placés entre crochets et entre guillemets.

Événements du clavier

Toutes les touches peuvent être utilisées, entre autres :

<a>, <b>, etc.	lettres (en minuscules)
<BackSpace>, <Tab>, <Return>	
<Left>, <Up>, <Right>, <Down>	flèches
<F1>, <F2>, ..., <F12>	

Événements de la souris

<Button-1>	bouton gauche de la souris pressé
<Button-2>	bouton du milieu de la souris pressé (s'il existe)
<Button-3>	bouton droit de la souris pressé
<B1-Motion>	la souris est bougée pendant que le bouton gauche est pressé
<ButtonRelease-1>	le bouton gauche est relâché
<Double-Button-1>	double-clic sur le bouton gauche
<Enter>	le pointeur de la souris entre dans une fenêtre
<Leave>	le pointeur de la souris quitte une fenêtre

12.6. Programme « Space Invaders »

space_invaders.py

```
# Space invaders

from tkinter import *
from time import sleep
from random import randint

def move(h, v):
    # h : déplacement horizontal, v : déplacement vertical
    global x, y
    x, y = x+h, y+v
    can.coords(vaisseau, x, y)

def haut(event):
    move(0,-10)

def bas(event):
    move(0,10)

def gauche(event):
    move(-10,0)

def droite(event):
    move(10,0)

x, y = 130, 500
x2, y2 = 130, 225

fen = Tk()
fen.title("Space invaders")
can = Canvas(fen, width=1100, height=600, bg ='white')
can.pack(side=TOP, padx=5, pady=5)
b3 = Button(fen, text ='Quitter', width=15, command=fen.quit)
b3.pack(side=RIGHT)
# images
photo1 = PhotoImage(file='vaisseau.gif')
vaisseau = can.create_image(x, y, image=photo1)
```

```
photo2 = PhotoImage(file='ovni.gif')
ovni = can.create_image(x2, y2, image=photo2)
fen.bind("<Up>", haut)
fen.bind("<Down>", bas)
fen.bind("<Left>", gauche)
fen.bind("<Right>", droite)

fen.mainloop()
fen.destroy()
```

Exercice 12.8

Complétez le programme du § 12.6, pour en faire le « Space Invaders » de vos rêves.
Ajoutez par exemple une touche pour tirer sur l'ovni, faites bouger l'ovni (manuellement ou automatiquement), faites tirer l'ovni, changez les vitesses, etc.

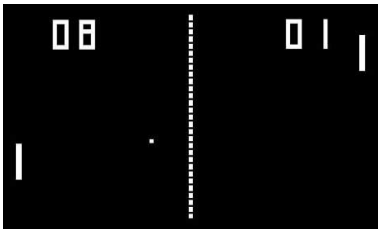
Exercice 12.9

En utilisant les images fournies sur le site compagnon, réalisez l'animation du personnage Laszlo, du jeu Nivalis. Utilisez les flèches du clavier pour bouger Laszlo.

Cet exercice est tiré du livre *Apprendre la programmation par le jeu*, par Vincent Maille, éditions Ellipse, 2013, page 173 (exercice M7).

Les images sont tirées du jeu Nivalis, qui a été créé par un jeune amateur nommé eXaHeVa.

Exercice 12.10



Pong est un jeu vidéo inspiré du tennis de table développé par Ralph Baer et son équipe en 1967. Après y avoir joué lors d'une première démonstration en mai 1972, Nolan Bushnell, créateur de la société Atari, en fait une version améliorée. C'est le premier jeu vidéo à connaître un succès populaire.

Programmez le jeu de Pong.

Vous trouverez une vidéo de démonstration sur YouTube :
https://www.youtube.com/watch?v=fiShX2pTz9A

12.7. Ce que vous avez appris dans ce chapitre

- Vous avez vu comment animer des objets à l'écran, automatiquement et manuellement.
- Il est possible de récupérer l'heure actuelle.
- L'instruction `bind` permet de gérer les événements du clavier et de la souris.