TP4 – Langage Python

Les sous-programmes (fonctions et procédures)

Pour tous les exercices vous définirez les différentes fonctions demandées dans un module de nom mDivers.py et vous écrirez votre programme principal dans un autre fichier.

Script 1 : légende

Une légende de l'Inde ancienne raconte que le jeu d'échecs a été inventé par un vieux sage, que son roi voulut remercier en lui affirmant qu'il lui accorderait n'importe quel cadeau en récompense. Le vieux sage demanda qu'on lui fournisse simplement un peu de riz pour ses vieux jours, et plus précisément un nombre de grains de riz suffisant pour que l'on puisse en déposer 1 seul sur la première case du jeu, deux sur la suivante, quatre sur la troisième, et ainsi de suite jusqu'à la 64e case.

Écrivez un script qui affiche le nombre de grains à déposer sur chacune des 64 cases du jeu ainsi que le poids du tas de sable. **Calculez** le nombre exact de grains (nombre entier) par case et pour chaque case le poids des grains déposés en sachant qu'un grain pèse 0.000036274 gr

Vous définirez une fonction de nom **`poids(nb,p)`** qui retournent le produit de **`nb * p`** avec **`nb`** pour le nombre de grains et **`p`** le poids d'un grain.

exemple d'affichage => Case : 16 Nombre de grains : 32768 Poids : 1.188626432 gr

Script 2 : minimum et maximum

Ecrire un script **`mini_maxi.py`** qui effectue une saisie de n entiers et qui affiche la plus petite et la plus grande des valeurs. Vous définirez trois fonctions, **`saisie(nb)`** qui effectue la saisie des n entiers dans une liste et qui retourne cette liste, **`min(a,b) et max(a,b)`** qui retournent respectivement la valeur mini et la valeur maxi des deux nombres entiers a et b, passés en paramètre.

Ensuite, vous ferez un appel à la fonction de saisie et un appel aux fonctions **`min(..) et max(...)`** afin de déterminer et d'afficher la plus petite et la plus grande valeur.

Script 3 : pair et impair

Ecrire un script **`pair_impair.py`** qui effectue une saisie de n entiers et qui affiche respectivement sur deux lignes, les nombres pairs et les nombres impairs .

Vous définirez :

- une fonction **`estPair(a)`** qui retourne vrai si le nombre entier **`a`** passé en paramètre est pair, faux sinon.
- une fonction **`affiche(liste)`** qui affiche la liste d'entiers passée en paramètre.

Ensuite, vous utiliserez la fonction de saisie définie au script 3 pour saisir les n valeurs et vous ferez un appel à la fonction **`pair(a)`** afin de ranger dans la bonne liste les éléments pairs et impairs. Enfin vous ferez un appel à la fonction **`affiche(liste)`** afin de voir le contenu de chaque liste.

Script 4 : lecture et écriture dans un fichier

Ecrire un script qui lit un fichier ligne par ligne (**`readline()`**) et qui écrit dans un autre fichier, toute les lignes dont le premier caractère est différent de **`'# '`**.
Vous définirez une fonction de nom **`filtre(car,source,destination)`**

- car : caractère à rechercher dans le fichier source
- source : le fichier à lire
- destination : le fichier à écrire

et une fonction de nom **`executeCde(cde,chaine)`** qui affiche utilisant la commande **`more`** du shell le contenu d'un fichier passé en paramètre.

PS : `readline()` lit une seule ligne à la fois et retourne une chaîne de caractères. Cette chaîne est vide en fin de fichier.

Script 5 : **Tirage au hasard de nombres entiers**

Savoir utiliser la fonction **`randrange()`** du module **`random`**. Cette fonction peut être utilisée avec 1, 2 ou 3 arguments. Avec un seul argument, elle renvoie un entier compris entre zéro et la valeur de l'argument diminué d'une unité. Par exemple, **`randrange(6)`** renvoie un nombre compris entre **0 et 5**.

Avec deux arguments, le nombre renvoyé est compris entre la valeur du premier argument et la valeur du second argument diminué d'une unité. Par exemple, **`randrange(2,8)`** renvoie un nombre compris entre **2 et 7**.

Si l'on ajoute un troisième argument, celui-ci indique que le nombre tiré au hasard doit faire partie d'une série limitée d'entiers, séparés les uns des autres par un certain intervalle, défini lui-même par ce troisième argument.
Par exemple, **`randrange(3,13,3)`** renverra un des nombres de la série **3, 6, 9, 12** :

```
for i in range(15):
  print random.randrange(3,13,3)  # affiche : 3 12 6 9 6 6 12 6 3 6 9 3 6 12 12
```

Ecrire un script de nom **`jeuCarte.py`** qui tire au hasard des cartes à jouer. Le nom de la carte tirée sera affiché par son nom.

Le programme affichera par exemple :

```
Frappez <Enter> pour tirer une carte : Dix de Trèfle
Frappez <Enter> pour tirer une carte : As de Carreau
Frappez <Enter> pour tirer une carte : ...
```

Script 6 : Lecture du contenu d'un répertoire et écriture de ce contenu sous la forme d'ancres dans une liste, puis dans un fichier html. Ouvrir le fichier **`lect-dossier-ecrit-liens.py`** et le compléter. Cet exercice a été fait partiellement en cours le 04/12.

Script 7 : **Tracé d'une spirale avec le module Turtle**

Ecrire un script **`spirale.py`** qui dessine une spirale. Pour cela vous définirez un sous-programme de nom **`deplace(x,y)`**, x et y sont deux paramètres correspondants à un déplacement.

Ensuite, vous utiliserez 2 boucles **`for`** imbriquées et un appel à la fonction **`deplace(..,..)`**.
A l'exécution du programme, vous devez obtenir le dessin d'une spirale.

Script 8 : Le **module Tkinter : exemple**

Ce module permet de créer des interfaces graphiques à l'aide composants appelés widgets. Voici un exemple d'utilisation simple. Le but de cet exemple est de tracer une droite dans une fenêtre.

```
from Tkinter import *

def traceDroite():
    "Tracé d'une ligne dans le canevas can"
    # coordonnées de la ligne
    c.create_line(10,190,190,10,width=2,fill="dark green")

#------ Programme principal -------
# Création de la fenetre :
fen = Tk()

# création des widgets (composants) :
c = Canvas(fen,bg='dark grey',height=200,width=200)
c.pack(side=LEFT)

btn1 = Button(fen,text='Quitter',command=fen.quit)
btn1.pack(side=BOTTOM)

btn2 = Button(fen,text='Tracer une ligne',command=traceDroite)
btn2.pack()

fen.mainloop() # boucle en attente d'événements
fen.destroy() # destruction (fermeture) de la fenêtre
```

Ecrire un script de nom **`droites.py`** qui a pour interface graphique une fenêtre avec 3 boutons.
Les actions possibles sont :

- le bouton « Tracer une ligne » trace une droite à chaque clic sur ce bouton
- le bouton « Autre couleur» permet de changer la couleur de la droite (liste de couleurs)
- le bouton « Quitter» quitte en fermant la fenêtre