INF1 : Algorithmique et Programmation

Cours 7 :
Révisions et Compléments

Domitile Lourdeaux

Université de technologie de Compiègne

Printemps 2024

1 Turtle

2 Révisions

1 Turtle

2 Révisions

Module Turtle méthodes

Le module turtle permet de dessiner des graphiques à l'aide des fonctions suivantes :

- reset() : effacer l'écran
- goto(x, y) : aller à l'endroit de coordonnées x, y
- forward(n) : avancer de n
- backward(n) : reculer de n
- color(couleur) : retourne ou fixe la couleur trait ou remplissage ('green', 'red', 'blue', ...)
- fillcolor(couleur) : retourne ou fixe la couleur de remplissage
- left(angle) : tourner à gauche d'un angle donné (en degrés)
- right(angle) : tourner à droite
- begin_fill : commencer à remplir un contour fermé à l'aide de la couleur sélectionnée
- end_fill : terminer le remplissage
- width(epaisseur) : choisir l'épaisseur du trait
- up() : lever le crayon (et pouvoir avancer sans dessiner)
- down() : baisser le crayon

Module Turtle : Exemple

Module figures (avec une seule fonction pour l'instant)

```
from turtle import *

def carre(cote, couleur):
    "fonction qui dessine un carré de côté et de couleur donnés"
    color(couleur)
    begin_fill()
    for i in range(4):
        forward(cote)
        right(90)
    end_fill()
```

Utilisation (pour dessiner 5 carrés bleus)

```
from figures import carre
from turtle import forward, up, down

for i in range(5):
    carre(50, 'blue')
    up()
    forward(60)
    down()
```

1 Turtle

2 Révisions
Tableaux à deux dimensions : TP5
Annales

1 Turtle

2 Révisions
Tableaux à deux dimensions : TP5
Annales

Matrice magique : énoncé

TP N°5 : matrices magiques

On considère des matrices carrées de nombres entiers d'ordre NMAX.

Écrire un programme permettant de :

a) Calculer la moyenne des valeurs d'une matrice de ce type

b) Vérifier si une matrice carrée est magique (la somme des valeurs sur chaque ligne, sur chaque colonne et sur les deux diagonales est la même).

Exemple de matrice magique :

14	5	17
15	12	9
7	19	10

Remarque : dans un premier temps, on n'écrira pas la partie du programme correspondant à la saisie des valeurs de la matrice par l'utilisateur.

Matrice magique : première solution

Algorithm 1 Matrice Magique : For (élégante mais non-optimale)

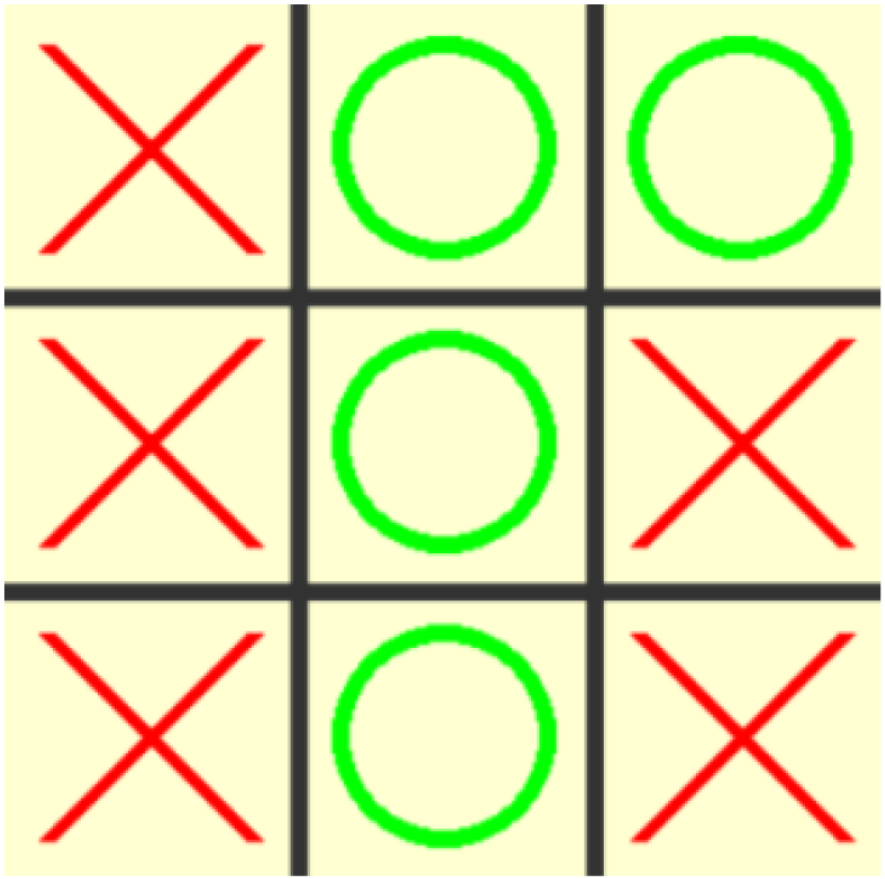
```
function matriceMagique (matrice : tab entiers) : booleen
    somme_ref ← sommeRef(matrice)
    set = []
    set = ajoutLignes, ajoutColonnes, ajoutDiagonales
    magique ← True
    pour chaque element elt ∈ set faire
        si somme_ref != sommeListe(elt): alors
            magique = False
        fin
    fin
    retourner magique
```

Matrice magique : deuxième solution

Algorithm 2 Matrice Magique : For (optimale mais moins intelligible)

```
function matriceMagique (matrice : tab entiers) : bool
    somme_ref ← sommeRef(matrice)
    magique ← True
      i ← 1
    tant que magique AND i < longeur(matrice) faire
        somme ← sommeLigne(matrice[i])
        si somme_ref != somme alors
            magique = False
        sinon
            i ← i + 1
        fin
    fin
// Idem pour les colonnes et diagonales
retourner magique
```

Tic Tac Toe : grille



sum(liste 1)	sum(liste 2)	sum(liste 3)
= +1 +1 +1	= -1 -1 -1	= -1 +1 +1
= 3	**= -3**	**= 1**

grille[0][0]	grille[1][0]	grille[2][0]
grille[0][1]	grille[1][1]	grille[2][1]
grille[0][2]	grille[1][2]	grille[2][2]

Tic Tac Toe : algorithme

Algorithm 3 Tic Tac Toe : première Ligne gagnante

```
somme ← grille[0][0] + grille[0][1] + grille[0][2]
si somme == 3 alors
    game_over ← True
    winner ← 1
sinon
    si somme == −3 alors
        game_over ← True
        winner ← 2
    sinon
fin
```

Il existe deux façons de procéder :

- 1ère méthode (la plus simple) : énumérer tous les cas possibles (il y en a 6 en tout)
- 2ème méthode (plus poussée) : en passant par une boucle.

Tic Tac Toe : algorithme pour les lignes

Algorithm 4 Tic Tac Toe : Lignes

```
l ← longueur(grille)
game_over ← False
i ← 0
tant que i < l AND game_over == faux faire
    somme ← 0
    pour j ← 1 to l faire
        somme ← somme + grille[i][j]
    fin
    si somme == 3 alors
        game_over ← True
        winner ← 1
    sinon
        si somme == −3 alors
            game_over ← True
            winner ← 2
        sinon
            i ← i + 1
        fin
    fin
```

1 Turtle

2 Révisions
Tableaux à deux dimensions : TP5
Annales

Médian P21 : Scrabble (Question 1)

Enoncé

- Au jeu de Scrabble, les joueurs doivent former des mots à partir des 7 lettres dont ils disposent. On appelle jeu cet ensemble de 7 lettres
- Écrire en Python une fonction booléenne acceptable(mot, jeu) qui retourne True si un mot est acceptable et False sinon.

Indication

Un mot est dit acceptable si on peut le former à partir des lettres du jeu, même si ce mot n'existe pas en français. On considérera pour cela que jeu et mot sont constitués uniquement de lettres majuscules non accentuées.

- Exemples : ELEVE, WVB ou BLE sont des mots acceptables avec le jeu BEWEELV

Conseils :

Pour écrire cette fonction, on pourra remarquer qu'un mot est acceptable si le nombre d'occurrences de chacune des lettres du mot est inférieur ou égal au nombre d'occurrences de chacune des lettres du jeu. On pourra utiliser la méthode count(car)d'une chaine de caractères, qui retourne le nombre d'occurrences du caractère car dans cette chaine.

- Par exemple, si le mot est ELEVE, mot.count('E') retourne 3.

Médian P21 : Scrabble (Question 1)

Solution 1

```
############################################################
#                    Exo Scrabble
############################################################
print('************************************')
print('********** Scrabble * **************')
print('************************************')

print('********************')
print('*** Question 1  ***')
print('********************')

def acceptablev1 (mot,jeu):
    possible = True
    for lettre in mot:
        if jeu.count(lettre) < mot.count(lettre):
            possible = False
    return possible
print(acceptablev1("ELEVE", "EELEVAM"))
print(acceptablev1("ELEVE", "EELEVAE"))
```

Remarque :

Cette version n'est pas optimale. On pourrait s'arrêter dès que la condition n'est plus vérifiée

Médian P21 : Scrabble (Question 1)

Solution 2

```
print('********************')
print('*** Question 1  ***')
print('********************')

def acceptablev2 (mot,jeu):
    possible = True
    i = 0
    while i < len(mot) and possible:
        if jeu.count(mot[i]) < mot.count(mot[i]):
            possible = False
        i += 1
    return possible
print(acceptablev2("ELEVE", "EELAVM"))
print(acceptablev2("ELEVE", "EELAVE"))
```

Médian P21 : Scrabble (Question 2)

Enoncé

- Écrire une fonction nbPoints(mot)qui retourne le cumul du nombre de points des lettres du mot mot

Valeur des lettres (version française du jeu)

- A,E,I,L,N,O,R,S,T,U : 1 **point**.
- D,G,M : 2 **points**.
- B,C,P : 3 **points**.
- F,H,V : 4 **points**.
- J,Q : 8 **points**.
- K,W,X,Y,Z : 10 **points**.

Médian P21 : Scrabble (Question 2)

Solution

```
print('********************')
print('*** Question 2  ***')
print('********************')
def nbPoints(mot):
    points = 0
    for lettre in mot :
        if lettre in 'EAINORSTUL':
            points = points + 1
        elif lettre in 'DMG':
            points = points + 2
        elif lettre in 'BCP':
            points = points + 3
        elif lettre in 'FHV':
            points = points + 4
        elif lettre in 'JQ':
            points = points + 8
        elif lettre in 'KWXYZ':
            points = points + 10
    return points

mot = "ELEVE"
print(f"Point pour {mot} : {nbPoints(mot.upper())}")
mot = "WHISKEY"
```

Médian P21 : Lancé de dés (1)

```
############################################################
#                    Exo dés
############################################################
print('************************************')
print('********** Lancés dés **************')
print('************************************')

# Question 1
#     Remplir un tableau avec la valeur du lancé de dé de 5 joueurs
import random
print('********************')
print('*** Question 1a ***')
print('********************')
DIM = 5
t = ['-']*DIM
print(t)
for i in range(DIM):
    t[i] = random.randint(1, 6)
print(t)

# ou
print('********************')
print('*** Question 1b ***')
print('********************')
DIM = 5
t = []
for i in range(DIM):
    t.append(random.randint(1, 6))
print(t)
```

Médian P21 : Lancé de dés (2)

```
# Question 2
#     Trouver le 1er joueur à faire un 6 dans ce panel de 5 joueurs
#     Afficher l'indice du 1er joueur à faire 6
print('******************')
print('*** Question 2 ***')
print('******************')
t = []
i = 0
b6 = False
while (b6 == False) and i < DIM:
    t.append(random.randint(1, 6))
    b6 = t[i] == 6
    i += 1
print(t)
if b6:
    print(f'Le premier joueur est le joueur {i}')
else:
    print("Personne n'a fait de 6")
```

Médian P21 : Lancé de dés (3)

```
# Question 3
#     Trouver le 1er joueur à faire un 6 dans un panel de 5 joueurs,
#     Recommencer les tours jusqu'à trouver le 1er 6
#     Afficher l'indice du 1er joueur à faire 6 et le numéro du tour
print('******************')
print('*** Question 3a ***')
print('******************')
b6 = False
tour = 0
while (b6 == False):
    tour += 1
    print(f'==== Tour {tour} ====')
    t = []
    i = 0
    while (b6 == False) and i < DIM:
        t.append(random.randint(1, 6))
        b6 = t[i] == 6
        i += 1
    print(t)
if b6:
    print(f'Le 1er joueur à faire un 6 est le joueur {i} au tour {tour}')
```

Médian P21 : Lancé de dés (4)

```
print('**************************')
print('***     Question 3b    ***')
print('*** Avec des fonctions ***')
print('**************************')
def Tour():
    t = []
    i = 0
    b6 = False
    while (b6 == False) and i < DIM:
        t.append(random.randint(1, 6))
        b6 = t[i] == 6
        i += 1
    return t
def LancesDes():
    b6 = False
    tour = 0
    while (b6 == False):
        tour += 1
        print(f'==== Tour {tour} ====')
        t = Tour()
        b6 = t[len(t)-1] == 6
        print(t)
    if b6:
        print(f'Le 1er joueur à faire un 6 est le joueur {i} au tour {tour}')

LancesDes()
```

Médian P21 : Lancé de dés (5)

```
# Question 4
#     Trouver le 1er joueur à faire un double 6 dans un panel de 5 joueurs
#     Afficher l'indice du 1er joueur à faire 6
print('********************')
print('*** Question 4a ***')
print('********************')
def Double6_1Tour():
    DIM = 5
    i = 0
    bDouble6 = False
    t = []
    while bDouble6 == False and i < DIM:
        lance = []
        for j in range(0, 2):
            lance.append(random.randint(1, 6))
        t.append(lance)
        bDouble6 = (t[i][0] == 6) and (t[i][1] == 6)
        i += 1
    print(t)
    if bDouble6:
        print(f'Le 1er joueur à faire un 6 est le joueur {i}')
    else:
        print("Personne n'a fait de double 6")

Double6_1Tour()
```

Médian P21 : Lancé de dés (6)

```python
#Question 4
#     Trouver le 1er joueur à faire un double 6 dans un panel de 5 joueurs,
#     Si personne ne gagne on refait un tour jusqu'à trouver un gagnant
#     Afficher Afficher l'indice du 1er joueur à faire 6 et le tour
print('******************')
print('*** Question 4b ***')
print('******************')
def Double6_nTours():
    DIM = 5
    bDouble6 = False
    tour = 0
    while (bDouble6 == False):
        t = []
        i = 0
        tour += 1
        print(f'==== Tour {tour} ====')
        while bDouble6 == False and i < DIM:
            lance = []
            for j in range(0, 2):
                lance.append(random.randint(1, 6))
            t.append(lance)
            bDouble6 = (t[i][0] == 6) and (t[i][1] == 6)
            i += 1
        print(t)

    if bDouble6:
        print(f'Le premier joueur à faire un double 6 est le joueur {i} au tour {tour}')
    else:S
        print("Personne n'a fait de double 6")
```

Médian P22 : Pair et Impair

Voir en ligne

Médian P23 : Chaines de caractères (1)

3ème partie : Chaines de caractères *(8 points)*

Soit les chaines de caractères suivantes :

S1 = "123456789"

S2 = "abcdefghi"

S3 = "Université de Technologie de Compiègne"

S4 = "portez ce vieux whisky au juge blond qui fume"

1. En utilisant le slicing Python extraire la chaine "5678" de s1.
2. En utilisant le slicing Python, comment extraire de la chaine s2 une lettre sur deux à l'envers et obtenir "igeca" ?
3. Ecrire le programme Python qui permet de créer une chaine de caractère s5 qui mélange des chaines de caractères de même longueur. En mélangeant s1 et s2, on obtient : "1a2b3c4d5e6f7g8h9i".
4. Modifier le programme précédent pour mélanger des chaines de longueurs différentes. En mélangeant s1 et s3, on obtient ; "1U2n3i4v5e6r7s8i9té de Technologie de Compiègne".
5. Ecrire un algorithme ou programme Python qui affiche les lettres majuscules de la chaine s3, ici "UTC"
6. Un pangramme est une phrase comportant au moins une fois chaque lettre de l'alphabet. Par exemple, s4 est un pangramme. Ecrire un algorithme ou programme Python qui affiche si une chaine de caractère est un pangramme ou non. On considère que toutes les lettes sont des minuscules.

Médian P23 : Chaines de caractères (2)

Remarque :

Même si l'accès aux caractères d'une chaine de caractères se fait comme avec les listes, contrairement aux listes, les chaines de caractères sont **immuables**.

Médian P23 : Chaines de caractères (3)

Questions 1 et 2 : en utilisant le slicing Python extraire la chaine "5678" de s1 et de s2 une lettre sur deux à l'envers et obtenir "igeca"

Médian P23 : Chaines de caractères (4)

```
#########################
# PARTIE 3
#########################
s1 = "123456789"
s2 = "abcdeFghi"
s3 = "Université de Technologie de Compiègne"
s4 = "portez ce vieux whisky au juge blond qui fume"
print('---------------------------')
print('------------ P3 -----------')
print('---------------------------')

# Q1 : Pour extraire la chaine "5678" (0.5)
print('******************')
print('*** Question 1 ***')
print('******************')
print(s1[4:8])

# Q2 : Chaine s2 à l'envers une lettre sur deux (0.5)
print('******************')
print('*** Question 2 ***')
print('******************')
print(s2[::-2])
```

Médian P23 : Chaines de caractères (5)

Question 3 : Ecrire le programme Python qui permet de créer une chaine de caractère s5 qui mélange des chaines de caractères de même longueur. En mélangeant s1 et s2, on obtient : "1a2b3c4d5e6f7g8h9i".

```
# Q3 : Mélanger les lettres (1.5)
print('********************')
print('*** Question 3 ***')
print('********************')
s5 = ""
for i in range (len(s1)):
    s5 = s5 + s1[i] + s2[i]
print(s5)
```

Médian P23 : Chaines de caractères (6)

Question 4 : Modifier le programme précédent pour mélanger des chaines de longueurs différentes. En mélangeant s1 et s3, on obtient ; "1U2n3i4v5e6r7s8i9té de Technologie de Compiègne".

```
# Q4 même chose si s1 et s2 de longeurs différentes (2)
print('******************')
print('*** Question 4 ***')
print('******************')
s5 = ""
if len(s1) < len(s3):
    min = len(s1)
    s6 = s3[len(s1):]
else:
    min = len(s3)
    s6 = s1[len(s3):]
for i in range(min):
    s5 = s5 + s1[i] + s3[i]
s5 = s5 + s6
print(s5)
```

Médian P23 : Chaines de caractères (7)

Question 5 : Ecrire un algorithme ou programme Python qui affiche les lettres majuscules de la chaine s3, ici "UTC"

```
# Q5 Créer acronyme à partir des lettres majuscules (1.5)
print('******************')
print('*** Question 5 ***')
print('******************')
acronyme = ""
for c in s3 :
    if c >= 'A' and c <= 'Z' :
        acronyme += c
print(acronyme)
```

Médian P23 : Chaines de caractères (8)

Question 6 : Ecrire un algorithme ou programme Python qui affiche si une chaine de caractère est un pangramme ou non. On considère que toutes les lettes sont des minuscules.

```
# Q6 : Ecrire un programme Python qui verifie si une chaine est un Pangramme
print('******************')
print('*** Question 6 ***')
print('******************')
pangramme = True
alphabet = "abcdefghijklmnopqrstuvwxyz"
i = 0
while i < len(alphabet) and pangramme:
    if not alphabet[i] in s4:
        pangramme = False
        #print(alphabet[i], "-", pangramme)
    else:
        #print(alphabet[i], "-", pangramme)
        i += 1
if pangramme:
    print("La chaine est un pangramme")
else:
    print("La chaine n'est pas un pangramme")
```

Questions...