I11 - Programmation I : Python
TP 8

Semestre 1 2017

Géométrie 2D/3D

Dans les exercices suivants, on fera appel au module `pocketgl` vu dans le TP 7.

EXERCICE 1. **Module** `vecteur`

Le but de ce premier exercice est de créer un module python de manipulation de vecteurs. Un vecteur $u = (u_1, u_2, \ldots, u_n)$ sera représenté par un tuple de flottants.

1. Créer un fichier `vecteur.py` en entête duquel vous mettrez le nom du module, votre nom et la date.
2. Écrire deux fonctions `somme_vect(u,v)` et `diff_vect(u,v)` qui retournent respectivement la somme et la différence, coordonnée par coordonnée, des vecteurs `u` et `v`.
3. Écrire une fonction `prod_scal(u,v)` qui retourne le produit scalaire $u.v = u_1v_1 + \cdots + u_nv_n$ des vecteurs `u` et `v`.
4. En mathématique, une matrice est un tableau à deux dimensions. Par exemple:

$$\begin{pmatrix} 1 & 0 & 2 \\ 2 & 1 & \text{-}0.5 \\ 3 & 1 & 1 \end{pmatrix}$$

   Dans cette partie, nous appellerons matrice une liste de vecteurs de même longueur. Par exemple la matrice précédente sera représentée en colonnes par la liste `[(1,2,3),(0,1,1),(2,-0.5,1)]`. Le produit d'une matrice `m` avec un vecteur `u` est la liste des produits scalaires du vecteur `u` avec chacun des vecteurs formant la matrice `m` (rem: tous les vecteurs doivent être de même longueur).

   Écrire une fonction `prod_mat_vec(u,m)` qui retourne le produit matrice vecteur entre `u` et `m`. Par exemple `prod_mat_vec((1,1),[(1,0),(1,1)])` retournera `(1,2)`.

EXERCICE 2. **Module** `geometrie_2d`

1. Créer un fichier `geometrie_2d.py` en entête duquel vous mettrez le nom du module, votre nom et la date.
2. Écrire une fonction `mat_rotation_2d(theta)` qui retourne la matrice de rotation centrée en (0,0) et d'angle théta

$$\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.$$

3. Écrire une fonction `rotation_point_2d(p, c, theta)` qui retourne le point résultat de la rotation de centre `c` et d'angle `theta` du point `p`.

4. Écrire une fonction `dessine_polygone_2d( pol )` qui dessine à l'écran un polygone donné sous forme d'une liste de sommets `pol`.

5. Écrire une fonction `rotation_polygone_2d( p, c, theta )` qui retourne la listes des coordonnées des sommets de la rotation de centre `c` et d'angle `theta` du polygone `p`.

6. Écrire et exécuter le script suivant dans le même répertoire que vos deux modules précédents:

```
from geometrie_2d import rotation_polygone_2d, dessine_polygone_2d
from math import pi
from time import sleep

theta = pi/32
C=(250,250)
P=[(300,300),(200,300),(200,200),(300,200)]
init_window('Rotation 2D', 500, 500)
while True:
    P=rotation_polygone_2d(P,C,theta)
    dessine_polygone_2d(P)
    refresh()
    sleep(0.05)
    clear_screen()
main_loop()
```

Exercice 3. **Module** `geometrie_3d`

1. Créer un fichier `geometrie_3d.py` en entête duquel vous mettrez le nom du module, votre nom et la date.

2. Écrire trois fonctions `mat_rotation_3dx(theta)`, `mat_rotation_3dy(theta)` et `mat_rotation_3dz(theta)` qui retournent, respectivement, les matrices R_x, R_y, R_z de rotation centrée en (0,0,0) d'angle théta et d'axe respectifs x, y et z:

$$R_x = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) \\ 0 & -\sin(\theta) & \cos(\theta) \end{pmatrix}, R_y = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix}, R_z = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

3. Écrire une fonction `rotation_point_3dx(p, c, theta)` qui retourne le point résultat de la rotation de centre `c` d'axe x et d'angle `theta` du point `p` puis faire de même pour les axe y et z.

4. Écrire une fonction `dessine_cone_3d( cone )` qui dessine à l'écran la projection d'un cone donné sous forme d'une liste de points `base` et d'un sommet `sommet` (correspondant au 3 premières coordonnées). Pour la projection, on n'utilisera simplement que les coordonnées x et y.

5. Écrire une fonction `rotation_3d( obj, c, cardan )` qui retourne la listes des coordonnées des sommets de la rotation de centre `c` d'angle autour des axes x, y, z `alpha, beta, gamma` de la liste de points `obj`.

6. Écrire un script permettant de visualiser la rotation d'un cone quelconque sur le modèle de l'éxercice précédent. On pourra par exemple essayer avec le cone défini par les coordonées : `[ (230,230,200),(300,300,0),(200,300,0),(200,200,0),(300,200,0)]`.