TP 4 : Fonctions

Exercice 1 : Fonctions et repère orthonormé

Nous allons manipuler dans ce TP des points dans l'espace (en trois dimensions donc) et des vecteurs (au sens mathématique). Nous allons pour ce faire utiliser des **vecteurs** (au sens de GNU Octave ici) pour représenter ces structures dans un repère orthonormé `(O,i,j,k)` tridimensionnel.

Un point et un vecteur seront donc représentés par un tableau (vecteur Octave, colonne ou ligne à votre choix) durant ce TP.

Nota :

- *La plupart des fonctions demandées dans ce TP existent déjà sous GNU Octave !*
  *L'objectif de ce TP est de les **réécrire** en utilisant des structures propres à notre programmation.*
- *Vous utiliserez dans vos fonctions la notion de « **Point 3D** » et « **Vecteur 3D** », représentés dans GNU Octave par un tableau à 3 cases (v(1) représentant la coordonnée en x du « Vecteur 3D » v, etc.).*

1) Ecrire une fonction **`Pt = Demande_Point()`** qui renvoie un « point 3D » à partir des entrées de l'utilisateur (vous utiliserez **exceptionnellement** l'instruction `input` dans une fonction).
2) Ecrire une fonction **`Pt=Creer_Point(x,y,z)`** qui renvoie un « point 3D » à partir des coordonnées `(x,y,z)` données en paramètre.
3) Ecrire une fonction **`Affiche_Point(Pt)`** qui permet d'afficher un point (sous forme d'un « + ») dans l'espace (utiliser la fonction **`plot3`** pour afficher un point en 3D)

4) Ecrire une fonction **`V=Vectorise(Pt1,Pt2)`** qui définit le vecteur $\overrightarrow{AB}$ à partir des points `A` et `B`.
5) Ecrire une fonction **`PV=Produit_Vectoriel(V1,V2)`** qui renvoie le vecteur résultat du produit vectoriel de 2 vecteurs.
6) Ecrire une fonction **`PS=Produit_Scalaire(V1,V2)`** qui renvoie le produit scalaire de 2 vecteurs.
7) Ecrire une fonction **`N=Norme(V)`** qui calcule la norme (distance) d'un vecteur.

8) Ecrire une fonction **`Affiche_Triangle(Pt1,Pt2,Pt3)`** qui affiche un triangle dans un espace 3D à partir de 3 points donnés en paramètre.
9) Ecrire une fonction **`Eq=Equation_Plan(Pt1,Pt2,Pt3)`** qui, à partir de 3 points passés en paramètre renvoie l'équation du plan correspondant sous la forme de 4 coefficients `[a, b, c, d]` représentant l'équation `ax + by + cz + d = 0`. On stockera les 4 coefficients dans une seule et même variable (dans un *vecteur ligne* par exemple).
10) Ecrire une fonction **`Affiche_Plan(Eq)`** qui affiche l'équation du plan en 3D

Exercice 2 : Un script pour les utiliser toutes !

Ecrire un programme (script) qui permet :

1) à l'utilisateur de rentrer les coordonnées de 3 points dans l'espace.
2) de déterminer les coordonnées du centre de gravité `G` du triangle dont les sommets correspondent aux 3 points définis par l'utilisateur, de l'afficher ainsi que le triangle correspondant
3) de calculer et d'afficher le périmètre et l'aire du triangle dans une figure en 3D qui affiche elle-même le triangle.
4) de calculer l'équation du plan passant par les 3 points, d'afficher avec le plan dans l'espace avec l'équation du plan.

Rappels

- Caractéristiques du centre de gravité d'un triangle :

$$\overrightarrow{GA} + \overrightarrow{GB} + \overrightarrow{GC} = \vec{0} \text{ et } \overrightarrow{3AG} = \overrightarrow{AB} + \overrightarrow{AC}$$

- Surface du triangle ABC : $S = \frac{1}{2}\left\|\overrightarrow{AB} \wedge \overrightarrow{AC}\right\|$

Exemple de points à utiliser :

```
A (1,1,0)
B (5,1,0)
C (3,4,0)
```