Exercice 1 : **Suivi de compte bancaire**

L'objectif de cet exercice est un programme de suivi de solde bancaire. Ce programme s'appuie sur trois fonctions :

- fonction « solde » : affiche le solde du compte
- fonction « debit(somme à retirer) » : retire la somme du compte, et affiche le solde modifié
- fonction « credit(somme à ajouter) » : ajoute la somme au compte, et affiche le solde modifié

Le corps du programme est décrit ci-dessous :

```
1  Solde=1000
2  n='init'
3  while n!=0:
4      n=int(input('faite votre choix : '))
5      if n==1:
6          solde()
7      elif n==2:
8          debit(int(input('quel débit ?')))
9      elif n==3:
10         credit(int(input('quel crédit ?')))
11     elif n==0:
12         print('fin de programme')
13     else:
14         print('choix non valide')
```

Partie 1 : Compréhension du programme

Q 1: Quelles sont les variables utilisées par ce programme ?
Q 2: Décrire les opérations réalisées à la ligne 8.
Q 3: Quelle est l'utilité de la structure conditionnelle ici ?
Q 4: Quelle est l'utilité de la structure « while » ?
Q 5: Quelle est la différence entre « Solde » et « solde » ?

Partie 2 : Finalisation

Q 6: Rédiger les fonctions « solde », « credit », « debit ».
Q 7: Vérifier le bon fonctionnement en exécutant le programme.

Exercice 1 : **Manipulation de liste**

Dans une console, exécuter ligne par ligne les instructions suivantes :

```
L1=[1,2,3]
L2=L1
id(L1),id(L2)
L1=L1+[5]
id(L1),id(L2)
```

Q 1 : A la fin, les variables L1 et L2 ont-elles le même identifiant ?
Q 2 : Expliquer quelle variable a changé d'identifiant, et quand.

Exercice 2 : **Effet de bord**

Préciser si à la fin de l'exécution des programmes ci-dessous, la variable `L` a été modifiée.

Cas 1
```
def ordre (M):
    M.sort()
    N=M
    return(N)

print(ordre(L))
```

Cas 2
```
def ordre (M):
    M.sort()
    N=M.copy()
    return(N)

print(ordre(L))
```

Cas 3
```
def ordre (M):
    N=M.copy()
    N.sort()
    return(N)

print(ordre(L))
```

Exercice 3 : **Tracés de polygones**

Dans un repère orthonormé $(O,\vec{x},\vec{y})$:

- Donner les coordonnées des 4 sommets d'un carré dont un sommet est confondu avec l'origine O, et un côté de longueur 10 est confondu avec l'axe $(O,\vec{x})$ (il y a 4 solutions, n'en retenir que celle pour laquelle les coordonnées des sommets sont positives ou nulles).
- Tracer ce carré en utilisant `matplotlib`.
- Généralisation : écrire une fonction qui trace un polygone régulier à n côtés de longueur 10, où n est l'argument de la fonction.

Exercice 4 : **Simulation géométrique d'un système bielle manivelle**

On se place dans le cas d'un compresseur à air. Le mouvement d'entrée est la rotation de la manivelle, le mouvement de sortie est la translation alternative du piston. (cf. Annexe - modélisation et paramétrage).

L'objectif est de décrire les mouvements en fonction du temps. Cela n'est pas possible dans une approche numérique basée sur des listes. Le temps sera donc défini uniquement pour certains instants suffisamment proches pour assimiler la liste à un intervalle continu.
On souhaite donc définir des instants "t" toutes les millisecondes pendant une période de 50 ms.

☞ Créer une liste *temps* qui contiendra les valeurs successives du temps.

Partie 1 : Définition de l'angle Alpha α: fonction à deux arguments

La vitesse de rotation d'entrée *omega* égale à 1500 tr/min.

✍ Définir α en fonction d'*omega* et de *temps.*

☞ Créer une liste *Alpha* contenant les valeurs de α aux instants étudiés.

Partie 2 : Calcul de Theta θ et de Lambda λ

☞ définir les longueurs *L1* (50mm) et *L2* (80mm)

☞ Créer deux listes, à partir des équations donnés en annexe, *Theta* et *Lambda* en fonction du temps.

Remarques :

- Les fonctions trigonométriques sont fonctions d'angle en radians.
- Les fonctions `acos()` et `asin()` ne sont pas chargées par défaut. Pour les charger, taper l'instruction :

```
from math import acos,asin
```

Partie 3 : Affichage du graphe Lambda en fonction de Alpha

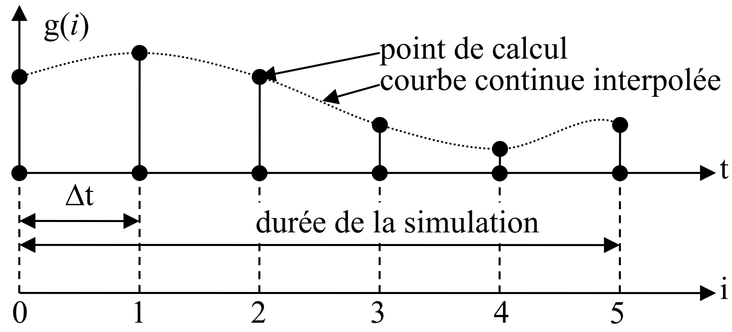
☞ Afficher la courbe de *Lambda* en fonction de *Alpha*

✍ Le mouvement est-il sinusoïdal ? Relancer le programme avec *L1*=60 et *L1*=20. Commenter.

Remarque 1 : échantillonnage

En méthode numérique, les calculs ne se font pas en continu, mais à des instants réguliers, appelés **point de calcul**, référencés par un **index** (i). La grandeur calculée g(t) n'est donc calculée qu'à ces instants. On la note aussi g(i).

L'intervalle de temps séparant deux points (Δt) est appelé le **pas** ou la **période d'échantillonnage**. Un point de calcul d'index i est alors défini à un instant $t=i.\Delta t$



Remarque 2 : Paramètres de simulation

Pour toute simulation numérique, il est nécessaire de définir :

- le pas (période d'échantillonnage)
- la durée de la simulation (le nombre total d'index)
- un mouvement d'entrée, par sa nature et ses valeurs à chaque instant
- Les conditions initiales (positions initiales)

Dans notre exemple, les paramètres de simulation utilisés sont :

- Une période d'échantillonnage : 1ms,
- la durée : 50ms,
- le mouvement d'entrée : la rotation définie par α,
- la valeur d'initiale du mouvement d'entrée : $\alpha_0=0$.

Tous les autres mouvements, dont celui de sortie, peuvent être alors calculés.

Partie 4 : Analyse de la courbe : détermination de la course.

On cherche la course du piston, c'est-à-dire l'amplitude du déplacement de celui-ci. Elle peut facilement être déterminée en calculant : $\lambda_{max} - \lambda_{min}$

☞ Proposer une fonction, basée sur une boucle, permettant de déterminer la valeur maximale de Lambda. Cette fonction devra aussi renvoyer l'instant de ce maximum.

☞ Utiliser cette fonction pour déterminer la course du piston et la durée d'un aller de celui-ci.

Partie 5 : Détermination de la position où la vitesse du piston est maximale

Il peut être démontré que la vitesse du piston est maximale si $\overrightarrow{y_1}$ et $\overrightarrow{y_2}$ sont perpendiculaires.

☞ Compléter le programme précédant pour afficher l'angle $(\overrightarrow{y_1}, \overrightarrow{y_2})$ en fonction du temps. Comment évolue cet angle ?

- Proposer un algorithme dichotomique qui recherche une valeur dans une liste triée décroissante. Proposer une condition d'arrêt.
- Tester et valider le programme pour déterminer la position pour laquelle $\overrightarrow{y_1}$ et $\overrightarrow{y_2}$ sont perpendiculaires

Partie 6 : Détermination d'une trajectoire

- Déterminer l'expression des coordonnées en X et Y, dans le repère lié au bâti, du point A et du point C et D, respectivement situés au tiers et au deux tiers du segment [AB]
- Faire calculer ces coordonnées des points A, C et D en fonction du temps.
- Représenter dans un nouveau graphe la trajectoire de A par rapport au bâti.

On va maintenant ajouter la trajectoire des points C, D et B.

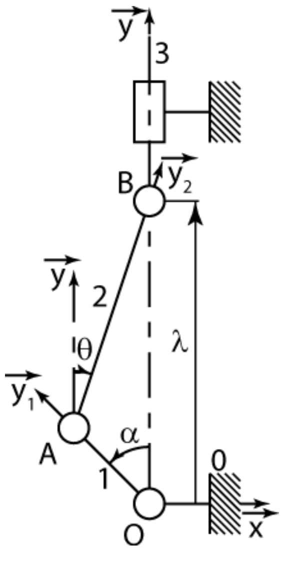
- Sans fermer cette figure, faire afficher la trajectoire des autres points.
- Enfin, sur une nouvelle figure, faire afficher le segment [AB] pour chacune des positions du temps.

Annexe : Système Bielle - Manivelle

Un système bielle – manivelle est un mécanisme de transformation du mouvement dont les applications sont très nombreuses.

La manivelle **1**, appelée aussi vilebrequin, a un mouvement de rotation par rapport au bâti **0**. La bielle **2** admet une rotation par rapport à la manivelle et par rapport au piston **3**. Ce piston a un mouvement de translation par rapport au bâti.
Le mouvement d'entrée est la rotation continue de la manivelle **1** par rapport à **0** (liée à un actionneur électrique). Le mouvement de sortie est la translation alternative du piston **3**.

On en déduit le schéma cinématique suivant :
Où :
$\alpha = (\vec{y}, \overrightarrow{y_1})$
$\theta = (\vec{y}, \overrightarrow{y_2})$
$\lambda = \overrightarrow{OB}.\vec{y}$
Et $l_1 = OA$; $l_2 = AB$



L'étude géométrique menée théoriquement donne :

$$\lambda = l_1 \cos\alpha + \sqrt{{l_2}^2 - {l_1}^2 \sin^2\alpha}$$

$$\theta = -\arcsin\left(\frac{l_1 \sin\alpha}{l_2}\right)$$

$$\dot{\lambda} = -l_1.sin\alpha.\dot{\alpha} - \frac{l_1^2.sin\alpha.cos\alpha}{\sqrt{l_2^2 - l_1^2 \sin^2\alpha}}\dot{\alpha}$$