TP 5 : Instructions conditionnelles et découverte des boucles *for*

0.1 Mise en route

1. Allumez l'ordinateur et ouvrez votre session.
2. Ouvrez l'application Edupython.
3. Au fur et à mesure du TP, vous ferez un compte rendu de vos observations et de vos réponses sur une feuille. Cette feuille sera **ramassée** par l'enseignant à l'issu du TP.

0.2 Colinéarité et déterminant

Objectif $n°1$ **:** *Créer une fonction permettant de vérifier si deux vecteurs sont colinéaires ou non*

Exercice 1. Soit $(O; I; J)$ un repère du plan et considérons les vecteurs

$$\overrightarrow{u} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}, \quad \overrightarrow{v} = \begin{pmatrix} 3 \\ -1.5 \end{pmatrix} \quad \text{et} \quad \overrightarrow{w} \begin{pmatrix} -5 \\ 2 \end{pmatrix}.$$

Partie A :

1. Sur votre compte rendu, étudier à la colinéarité (à l'aide du déterminant) des vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$. Faites de même avec les vecteurs $\overrightarrow{u}$ et $\overrightarrow{w}$.

Partie B :

L'objectif de cette partie est de **construire une fonction** permettant de retrouver les résultats de la partie A. Ouvrez un nouveau fichier et enregistrer le dans votre dossier personnel sous le nom ***TP5 exo1 NOM***.

1. Recopier dans **l'éditeur** et compléter le programme ci-dessous :

```
def determinant(a,b,c,d):
    det=...
    if ...:
        print('les vecteurs sont colinéaires')
    else :
        print('....')
```

2. Tester votre programme pour retrouver les résultats de la partie A

0.3 Découvertes des boucles *for*

Objectif $n°2$ **:** *Créer des fonctions dans lesquelles certaines instructions dépendent d'une boucle de répétitions* for

Exercice 2. Un nombre entier positif est dit premier si ses seuls diviseurs sont 1 et lui même.[1]. Par exemple, 2 et 5 sont des nombres premiers ; en revanche, 10 n'est pas un nombre premier parce qu'il est divisible par 1, par 2, par 5 et par 10.

Partie A :

1. Sur votre compte rendu, indiquez si 39 est un nombre premier.
2. Même chose, avec le nombre 23.
3. Expliquer votre démarche afin qu'elle soit reproductible sur d'autres nombres potentiellement plus grand. *Indication : utiliser la division euclidienne ; en python il faut remplacer l'opération diviser / par %.*

Partie B :

L'objectif de cette partie est de comprendre le fonctionnement du programme ci-dessous qui permet de vérifier si un nombre est premier ou non.[2] Ouvrez un nouveau fichier et enregistrer le dans votre dossier personnel sous le nom ***TP5 exo2 NOM***.

1. Recopier dans **l'éditeur** l'algorithme ci-dessous :

```
def prem1(n) :
    assert n>=1
    for i in range(2,n) :
        if n%i==0 :
            return False
    return True
```

2. Que se passe-t-il si vous testez le programme avec entier $n < 2$? En déduire le sens de la commande inscrite à la deuxième ligne.
3. Si $n = 6$ expliquer, en détaillant les calculs effectués par l'ordinateur, pourquoi le programme affirme qu'il ne s'agit pas d'un nombre premier. L'ordinateur vérifie-t-il que 6 est divisible par 3 ?
4. A l'aide du programme, vérifier les résultats obtenus à la partie A. Etudier ensuite la primalité du nombre 239.

1. Par convention, 1 n'est pas un nombre premier
2. En pratique, ce test est beaucoup trop simple pour être efficace sur des grands nombres

Exercice 3. En déposant à la banque une somme d'argent, il est possible de bénéficier d'intérêts. Ces derniers sont calculés à partir d'un pourcentage (taux d'intérêts) défini au préalable par l'établissement bancaire ou l'état ; les intérêts sont ensuite versés sur le compte bancaire en début d'année. Actuellement le taux d'intérêts du livret A est de 0.5%.

Partie A :

1. Si Yuriy dépose 10 000 euros sur son livret A en 2020, de combien disposera-t-il en 2021 ?
2. Proposer une formule permettant de calculer directement, à l'aide d'une multiplication et à partir du montant déposé en 2020, le montant disponible en 2021.
3. Quel argent possèdera Yuriy sur son livret A en 2022 (nous supposons que Yuriy laisse l'intégralité de l'argent, ainsi que les intérêts reçus, sur son livret A) ?

Partie B :

L'objectif de cette deuxième partie est de voir comment **introduire des boucles de répétitions** dans un programme. En particulier, il permettra de connaitre l'argent disponible, en tenant compte des intérêts successifs, sur un compte bancaire après n années. Ouvrez un nouveau fichier et enregistrer-le dans votre dossier personnel sous le nom ***TP5 exo3 NOM***.

1. Recopier dans **l'éditeur** et compléter l'algorithme ci-dessous

```
def livreta(n) :
        u=...
        for i in range(n) :
            u=1.005*u
        b=int(u*100)/100
        print('le montant disponible après', str(n), 'années est de', str(b), 'euros')
```

2. Si $n = 2$, détailler les calculs effectués par l'ordinateur dans la ligne 4 à la suite de la boucle *for*.
3. Quelle somme sera disponible au bout de 10 ans ?
4. Le taux d'intérêt du livret d'épargne populaire est de 1%. Que faut-il modifier dans le programme précédent pour calculer les intérêts au bout de n années si Yuriy dépose 5 000 euros sur un livret d'épargne populaire ?

L'essentiel des nouveautés de ce TP est résumé dans le tableau ci-dessous :

MÉMO

▶ Il est parfois utile dans un programme de répéter une ou plusieurs instructions un nombre défini de fois. Lorsque le nombre de répétitions est connu à l'avance, on utilise une boucle bornée *for*.

▶ **La syntaxe d'une boucle bornée**

Langage naturel	Langage Python
Pour variable **allant de** minimum **à** maximum instruction(s)	for variable in range() : instruction(s)

▶ La fonction range() permet d'énumérer le nombre de passages dans la boucle bornée. Elle peut être appelée de plusieurs façons :

- range(n), où n est un entier, fait prendre à la variable les valeurs entières de 0 à $n - 1$, donc n valeurs ;
- range(n, m), où n et m sont des entiers, fait prendre à la variable les valeurs entières de n à $m - 1$;
- range(n, m, k), où n, m et k sont des entiers, fait prendre à la variable les valeurs entières de n à $m - 1$, avec un pas de k.

Instructions	`for i in range(3) :` `  print(i)`	`for i in range(12,16) :` `  print(i)`	`for i in range(5,15,3) :` `  print(i)`

Affichage	La variable i prend les valeurs entières de 0 à 2, donc trois valeurs.
	`>>>`
	`0`
	`1`
	`2`
		La variable i prend les valeurs entières de 12 à 15.
		`>>>`
		`12`
		`13`
		`14`
		`15`
			La variable i prend les valeurs de 5 à 15, avec un pas de 3.
			`>>>`
			`5`
			`8`
			`11`
			`14`

▶ Il n'existe pas d'instruction pour définir la fin de la boucle. C'est l'indentation, c'est-à-dire le décalage vers la droite d'une ou plusieurs lignes, qui permet de marquer la fin de la boucle.