Académie LYON

JOURNÉES DE L'INSPECTION – RÉFORME DU LYCÉE 2019

ATELIER PYTHON

académie Lyon

MINISTÈRE
DE L'ÉDUCATION NATIONALE,
DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

1. Présentation de Python

Pourquoi Python ?

Python est un langage de programmation:

- *Facile à apprendre*
- *Simple à lire*
- *Riche en fonctionnalités (modules)*
- *Très utilisé (mathématiques et supérieur)*

Java

```java
public class Main {
  public static void main(String[] args) {
      System.out.println("hello world");
   }
}
```

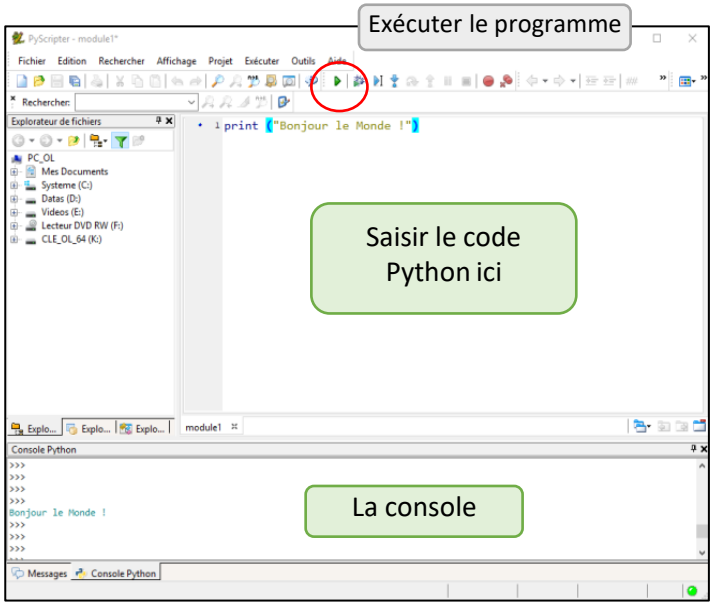
Python

```python
print("hello world");
```

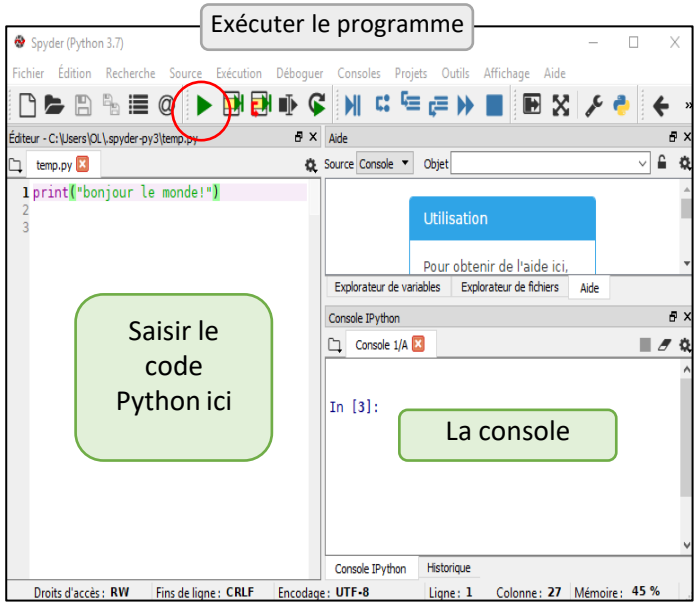
1. Présentation de Python

Où et comment saisir un programme ?

Avec EduPython



Avec Anaconda



1. Présentation de Python

Programme

Ensemble d'instructions (ou ordres) compréhensibles par l'ordinateur

```
# ceci est un commentaire (non interprété par Python)

nom = input("Entrez votre nom")
print("Bonjour "+nom+", comment allez vous ?")
if nom=="Einstein":
        print("Vous êtes un grand physicien alors !")
        nom2="Newton"
        print("Peut-être connaissez-vous "+nom2+" ?")
else:
        print("Enchanté !")
```

Les instructions respectent une « grammaire » propre à chaque langage

Une mauvaise instruction entraine une « syntax error »

(voir page 4 du livret: les erreurs fréquentes)

1. Mise en application immédiate

1. Lancer EduPython ou Spyder et charger le programme « inutile.py » (situé dans le dossier **exercices** de la clé USB)
2. Exécuter le programme. Comprendre ce qu'il fait
3. Remplacer dans le code « input » par « imput » et exécuter… SYNTAX ERROR !
4. Remettre « input » et saisir comme nom « Einstein ». Comprendre ce que fait le programme
5. Saisir maintenant « einstein ». Remarquer l'importance de la casse en Python.

```
# ceci est un commentaire (non interprété par Python)

nom = input("Entrez votre nom")
print("Bonjour "+nom+", comment allez vous ?")
if nom=="Einstein":
        print("Vous êtes un grand physicien alors !")
        nom2="Newton"
        date=1642
        print("Peut-être connaissez-vous "+nom2+", né en "+str(date)+" ?")
else:
        print("Enchanté !")
```

2. Bases de la programmation

Variable Sorte de “case” dans laquelle on stocke une donnée

Exemples en Python:

Type entier (int) ⟶	`A = 2`
Type texte (string) ⟶	`T = "Bonjour"`
	`N = int(input("Entrez un nombre entier"))`
Type réel (float) ⟶	`R = float(input("Entrez un nombre réel"))`
Type texte (string) ⟶	`Texte = str(A)`
Type entier (int) ⟶	`Somme = A + 4`
Type texte (string) ⟶	`NouveauTexte = T + str(A)`

Nom de variable:

- **Pas d’accents, pas d’espaces, pas d’instructions Python**
- **Respecter la casse**
- **Nom explicite (si possible)**

2. Bases de la programmation

Condition

Elle permet d'exécuter un bloc d'instructions en fonction du résultat (VRAI ou FAUX) d'une comparaison.

Exemple:

```
A = int(input("Entrez un nombre")
if A > 0:
        print("nombre positif")
elif A < 0:
        print("nombre négatif")
else:
        print("nombre nul")
```

Opérateurs de comparaison:

Algorithme en langage naturel	Instruction en Python
Si A = B	if A==B:
Si A différent de B	if A!=B:
Si A > B	if A>B:
Si A > ou égal à B	if A>=B:
Si A compris entre 0 et 2	if 0<A<2:
Si A = 2 et B = 4	if A==2 and B==4:
Si A = 2 ou B = 4	if A==2 or B==4:

Attention au double « égal »

Boucle

Elle permet de répéter plusieurs fois une suite d'instructions (appelée **bloc**)

On connaît le nombre de fois

Boucle « for »

```
for i in range(4):
    print ("Bonjour")
    print ("le monde")
```

```
for i in range(1,10):
    print (i)
```

i prend des valeurs de 1 à 9 (=10-1)

On ne connaît pas le nombre de fois

Boucle « while »

```
A=0
while A<9:
    A= A+1
    print (A)
```

Les instructions à exécuter dans la boucle doivent être alignées (on dit « indentées »)

Les blocs et l'indentation

En python, les blocs d'instructions suivant une condition ou une boucle doivent être INDENTÉES

```
for i in range(4):
    print ("Bonjour")
    print ("Hello")
    print ("Hallo")
    print ("Ciao")
print ("Salut")
```

Bloc d'instructions de la boucle if

```
Nom = input("Votre nom ?")
if Nom == "Albert":
    # l'utilisateur est Albert
    print ("E=mc2")
    print ("Relativité")
    print ("Physique")
else:
    # pas Albert
    print ("Bonjour")
    print ("Hello")
    print ("Ciao")
print ("Salut")
```

Les erreurs d'indentation sont fréquentes chez les élèves (et les programmeurs !)

(voir page 4 du livret: les erreurs fréquentes)

3. Mise en application n°1

Exercice 1

Objectif : S'approprier les fonctionnalités de bases en Python

1. Afficher « Bienvenue »
2. Demander à l'utilisateur de saisir une longueur d'onde en nm (mise dans une variable **longueurOnde** de type réelle)
3. Calculer la valeur de la fréquence correspondante:

frequence = $3.00x10^{8}$/ (***longueurOnde*** $x10^{-9}$)

Attention:
- La "virgule" se tape: . (point)
- Multiplier se tape: *

4. Afficher:

« La fréquence de l'onde dont la longueur d'onde est égale à » + **longueurOnde** + « nm vaut » + **frequence** + « Hz. »

la fréquence devra être affichée en notation scientifique avec 3 chiffres significatifs (voir livret page 9)

Aide: - Livret page 8 et 9
- $1x10^{9}$ se tape `1e9` en Python (ou `10**9`)
- pour transformer un nomnre en texte, utiliser: **str(nombre)**

3. Mise en application n°1

Exercice 2

Objectif : Traduire en langage Python l'algorithme ci-dessous permettant d'afficher la structure électronique

1. Définir la variable **numeroAt** = 20
2. Tant que **numeroAt** < 0 ou **numeroAt** > 18
   Demander à l'utilisateur un entier et le mettre dans **numeroAt**

Afficher 'La configuration électronique de l'élément de numéro atomique' + **numeroAt +** 'est : '
si **numeroAt** <=2 alors:
afficher '1s' + numeroAt
si 2 < **numeroAt** ≤= 4 alors:
afficher '1s2 2s' + numeroAt-2
si 4 < **numeroAt** <= 10 alors:
afficher '1s2 2s2 2p' + numeroAt-4
si 10< **numeroAt** ≤ 12 alors:
afficher '1s2 2s2 2p6 3s' + numeroAt-10
si 12< **numeroAt** ≤ 18 alors:
afficher '1s2 2s2 2p6 3s2 3p' + numeroAt-12

Aide pour les conditions: Livret page 6

3. Mise en application n°1

Suite de l'exercice 2 pour ceux qui veulent aller plus loin

Objectif : poursuivre le programme précédent en affichant également le symbole de l'élément chimique et son nom.

1. Créer une liste des 18 premiers éléments sous la forme de texte ("H", "He", etc…)
2. Créer une liste avec le nom des ces éléments (c'est un peu fastidieux…)
3. Créer un affichage du type

'L'élément dont le numéro atomique est Z=' (mettre le numéro atomique) 's'appelle' (mettre le nom) ' et a pour symbole ' (mettre le symbole)
' Sa configuration électronique est : '

Aide pour les listes:
Livret page 6 et 7

3. Mise en application n°1

Exercice 3 (pour les très rapides)

Objectif : Ecrire un programme calculant les angles pour la réfraction

1. Demander l'angle d'incidence en degré, l'indice du milieu 1 et celui du milieu 2
2. Transformer l'angle en radian
3. Calculer l'angle de réflexion et l'angle de réfraction
4. Les exprimer en degré et afficher un texte clair.

Attention, il faudra gérer le cas de la réflexion totale…

Aide:
- Taper: `import numpy as np` en début de programme
- Le sinus de x se tape: `np.sin(x)` (x en radians)
- Π se tape: `np.pi`
- Arcsinus se tape: `np.arcsin(valeur)`

4. Les modules

Python et les modules

Le nombre de fonctions de base dans Python est limité.

Il est possible d'ajouter des nouvelles fonctions à Python en **important** des modules

Exemples de modules

Numpy	Rajoute des fonctions de calcul
Matplotlib	Permet de tracer des courbes
PyGame	Permet de programmer des jeux
Tkinter	Permet de créer des interfaces graphiques
Il existe un très grand nombre de modules	

Pour utiliser un module : `import Nom_Du_Module` À placer au début du programme

5. Le module Numpy

Python et le module Numpy

Numpy permet d'ajouter des fonctions mathématiques à Python ainsi que des tableaux très pratiques.

À placer au début du programme: `import numpy as np`

"alias" de numpy

```
print(np.pi)
print(np.sin(3e-2))
print(np.log10(3e-2))
```

Une liste (non exhaustive) des fonctions mathématiques se trouve page 10 du livret

Remarque: les puissances ne nécessitent pas numpy.

$10^{4,8}$ s'écrit : `print(10**4.8)`

Les tableaux de Numpy

Numpy permet de créer et manipuler simplement des tableaux de données

Exemple:

```
t = np.array([1,2,3])

x = 2*t

print(x[0])
print(x[1])
print(x[4])
```

x devient un nouveau tableau qui contient les valeurs de t fois 2

t	x	
1	2	← Index 0
2	4	← Index 1
3	6	← Index 2

Résultat à l'écran

```
2
4
Error: index out of range
```

Intérêt en physique-chimie

Un tableau Numpy = une grandeur

5. Mise en application du module Numpy

Exercice 4 (Numpy)

Objectif : Entrer des valeurs dans un tableau NUMPY et les manipuler

1. Importer le module NUMPY
2. Créer un tableau de dates nommé t avec les valeurs suivantes : 1,2,3,4,5 jusqu'à 10
3. Créer un second tableau nommé x valant 2 fois t
4. Créer un 3ème tableau nommé y valant (-1/2) de $t^2 + 4$
5. Afficher les 3 tableaux
6. Afficher le dernier élément du 2nd tableau
7. Afficher le 1er élément du 1er tableau

Aide: page 10 du Livret

Pour aller plus loin…

1. Créer un tableau t allant de 0 à 2 par pas de 0,04
2. Créer 2 listes contenant les équations horaires suivantes:

$$\begin{cases} x(t) = 2t \\ y(t) = -\frac{1}{2} g t^2 + 2t + 1 \end{cases}$$

Aide: page 10 du Livret et la fonction ARANGE de Numpy

6. Le module Matplotlib

Python et le module MATPLOTLIB

Le module Matplotlib permet de tracer facilement des courbes et nuages de points à partir de tableaux numpy

À placer au début du programme: `import matplotlib.pyplot as plt`

"alias" de pyplot

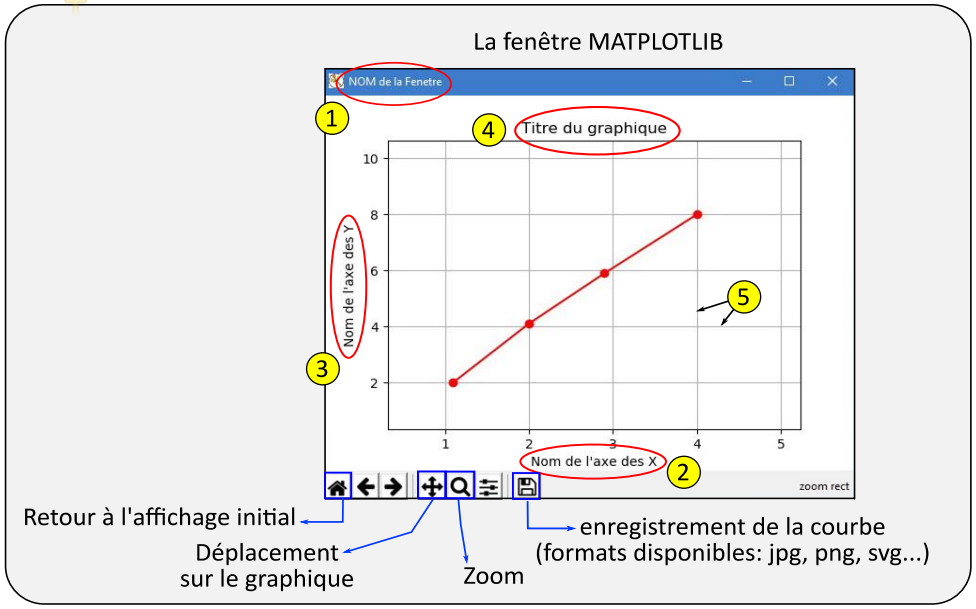
1

Instruction utiles:

```
1 plt.figure("NOM de la Fenetre")
2 plt.xlabel("NOM de l'axe des X")
3 plt.ylabel("NOM de l'axe des Y")
4 plt.title("TITRE du graphique")

5 plt.grid()
```

À placer à la fin:

```
plt.show()
```



6. Le module Matplotlib

Instruction pour tracer une courbe:

`plt.plot(`*abscisse*`,`*ordonnée*`,`*style*`)`

Peuvent être:
- des valeurs réelles
- des tableaux Numpy

- Type de point
- Couleur des points
- Relié ou non

Styles disponibles : à placer entre guillemets *dans l'ordre* : **1. Couleur** **2. Type de point** **3. Type de tracé**

Couleurs						Type de points tracés					Tracé	
r	b	g	k	m	c	o	.	x	+	v	-	--
Rouge	Bleu	vert	noir	magenta	cyan	Gros point	Petit point	Croix	Croix +	Triangle	Points reliés	Points reliés en pointillé

6. Mise en application du module Matplotlib

Exercice 5 (Matplotlib)

Objectif : Entrer des valeurs dans un tableau NUMPY et les manipuler

1. Reprendre le code que vous avez tapé pour l'exercice Numpy
2. Importer le module MATPLOTLIB en début de programme
3. Après la création des tableaux numpy, tracer y en fonction de x, en croix rouges.
4. Donner un nom au graphique
5. Donner des noms aux axes
6. Afficher la grille
7. Afficher la fenêtre Matplotlib

Aide: page 11 et 12 du Livret

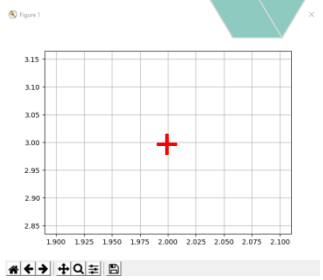
Pour aller plus loin…

1. Reprendre le code que vous avez tapé pour l'exercice Numpy « pour aller plus loin »
2. Importer le module MATPLOTLIB en début de programme
3. Tracer les équation horaires de deux couleurs différentes sans croix mais points reliés
4. Afficher le texte « x(t) » proche de la bonne courbe
5. Idem pour y(t)

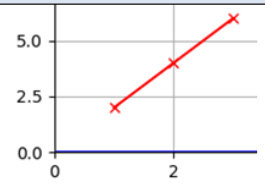
6. Le module Matplotlib

Autres exemples liés aux SPC:

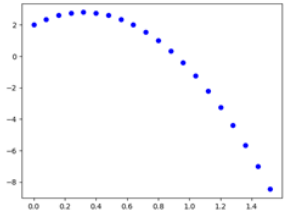
```
import matplotlib.pyplot as plt

plt.plot(2,3,"+r")
plt.grid()
plt.show()
```



Instructions en Python	Résultat à l'écran

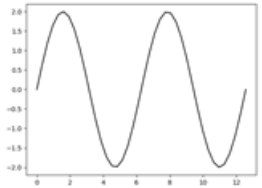
`I = np.array([1,2,3])`
`U = np.array([2,4,6])`
`plt.plot(I,U,"rx-")`
`plt.show()`
	

Instructions en Python	Résultat à l'écran

`t = np.arange(0,2,0.1)`
`X = 0.8*t`
`Y = -5*t**2 + 4*t + 2`
`plt.plot(X,Y,"bo")`
`plt.show()`
	

Attention, faute de frappe dans le livret page 12 →

Instructions en Python	Résultat à l'écran

`X = np.linspace(0,4*np.pi,40)`
`Y = 2*np.sin(X)`
`plt.plot(X,Y,"k-")`
`plt.show()`
	

7. Mise en application n°2 (si le temps le permet…)

Exercice 6 (Numpy)

Objectif : Entrer des valeurs dans un tableau NUMPY et les manipuler

1. Importer le module NUMPY
2. Créer un tableau avec les valeurs suivantes : -6.02E-1,-8.48E-1,-1.07,-1.39
3. Créer un second tableau avec les valeurs absolues du premier tableau
4. Afficher le premier et le second tableau
5. Afficher le dernier élément du 2nd tableau
6. Afficher le 1er élément du 1er tableau
7. Afficher tous les éléments du second tableau (en sautant une ligne entre chaque élément)

Exercice 7 (Numpy et Matplotlib)

Objectif : Afficher des données expérimentales

1. Ouvrir dans l'éditeur le fichier '**Exercice_7_matplotlib.py**'
2. Suivre les instructions en rose (avec EduPython) ou vert (avec Spyder)

Remarque: pour ne pas afficher y(x) et Ep(t) dans la même fenêtre, il faudra fermer la 1ère pour que la seconde apparaisse

Aide: page 10, 11 et 12 du Livret

7. Mise en application n°2 (si le temps le permet…)

Exercice 8 (Pour les plus rapides)

Objectif : Tracer une fonction périodique dans Numpy

1. Importer le module NUMPY et MATPLOTLIB
2. Demander une période T (réel) en seconde à l'utilisateur
3. Demander une amplitude Umax en volt à l'utilisateur
4. Créer un tableau numpy appelé « t » de 200 valeurs entre 0 et 4 périodes
5. Créer une fonction U sinusoïdale: $U = U_{max}.\sin(2\pi.t/T)$
6. Tracer U en fonction de t (avec grille, croix rouges, reliées ou non)

Aide: page 10, 11 et 12 du Livret

8. Activités élèves

Activité élève:

Tracé de la trajectoire et du vecteur vitesse en un point

Travail des élèves

- Pointage X et Y puis export au format texte

(Fiche fournie pour expliquer l'export sous: Avimeca, avistep, Latispro, Pymacavideo…)

- Tracé de la trajectoire
- Programmation des formules Vx et Vy au point 5
- Tracé du vecteur vitesse en M_5
- Généralisation en un point M_n

***Durée** (bilan de 2 tests en MPS):*

Pointage avec Latispro + export + Python**:* ***1h30

Attendus officiels:

Représenter les positions successives d'un système modélisé par un point lors d'une évolution unidimensionnelle ou bidimensionnelle à l'aide d'un langage de programmation.

Représenter des vecteurs vitesse d'un système modélisé par un point lors d'un mouvement à l'aide d'un langage de programmation.

8. Activités élèves

La partie Python:

Choix du logiciel de pointage utilisé (**par le professeur**)

Le fichier "import_donnees_meca.py" doit être fourni aux élèves

```
""" ---------------------------------
        CHOIX du PROFESSEUR
---------------------------------- """
#  Logiciel utilisé à choisir parmi :
# 'pymecavideo'     (exporter un fichier TXT)
# 'latispro'        (fichier TXT créé par LATISPRO, avec décimale VIRGULE et séparation POINT VIRGULE)
# 'avimeca'         (créeer un fichier TXT en export)
# 'avistep'         (pointage exporté depuis AVISTEP:
# 'equation'        (équation paramétrées saisies dans le fichier "import_donnees")
# 'entrees'        (données de pointage saisies "à la main" dans le fichier "import_donnees.py)
Logiciel_Utilise = 'latispro'
""" ---------------------------------
        FIN du CHOIX du PROFESSEUR
---------------------------------- """

t, x, y = traiteDonnees(Logiciel_Utilise)
```

Le programme récupère du pointage 3 tableaux numpy: **x , y et t**

```
""" ##########################################
Ici commence le travail à réaliser par les élèves
########################################## """
# ####################################
# Tracé de la trajectoire
# ####################################
""" TRAVAIL 1:
Taper ci-dessous le code Python permettant de tracer l'ordonnée du point en fonction de son abscisse. """
```

Les élèves doivent tracer la courbe (**aide fournie**) Et compléter le code

Pour tracer la courbe représentant une grandeur A en fonction d'une grandeur B, il faut taper : **`plt.plot(B,A,"kx")`**
Pour le choix des couleurs et du dessins des points, voir annexe.

```
""" TRAVAIL 2:
taper, à la place des points entre guillemets, les légendes pour les axes,
le titre du graphique"""
plt.title("....................")          # titre du graphique
plt.xlabel("....................")          # nom de la grandeur sur l'axe X
plt.ylabel("....................")          # nom de la grandeur sur l'axe Y
```

8. Activités élèves

Informations apportées aux élèves

Le vecteur vitesse au point 5 s'exprime de la manière suivante :

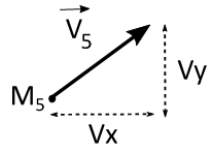
$$\vec{V_5} = \frac{\overrightarrow{M_4M_6}}{t_6 - t_4}$$

où M_4 et M_6 sont les noms des points n° 4 et 6
et t_4 et t_6 sont les dates de passage du système aux points n°4 et n°6

L'abscisse Vx du vecteur vitesse au point n°5 (par exemple) s'exprime :

$$Vx = \frac{x_6 - x_4}{t_6 - t_4}$$

où x_4 et x_6 sont les abscisses du système au points M_4 et M_6
et t_4 et t_6 les dates de passage du système aux points M_4 et M_4



Les élèves doivent créer Vx et Vy en M_5

```
Vx = (x[6]-x[4])/(t[6]-t[4])
Vy = (y[6]-y[4])/(t[6]-t[4])
```

```
""" TRAVAIL 3:
Taper ci-dessous le code python permettant:
    - de mettre dans une variable Vx l'abscisse du vecteur vitesse au point 5
    - de mettre dans une variable Vy l'ordonnée du vecteur vitesse au point 5 """
```

En Python, l'abscisse du système au point x_6 se tape : **`x[6]`**
La date t_6 au point 6 se tape : **`t[6]`**

Les élèves doivent tracer le vecteur V_5

```
draw_Vector(5,Vx,Vy,"b")
```

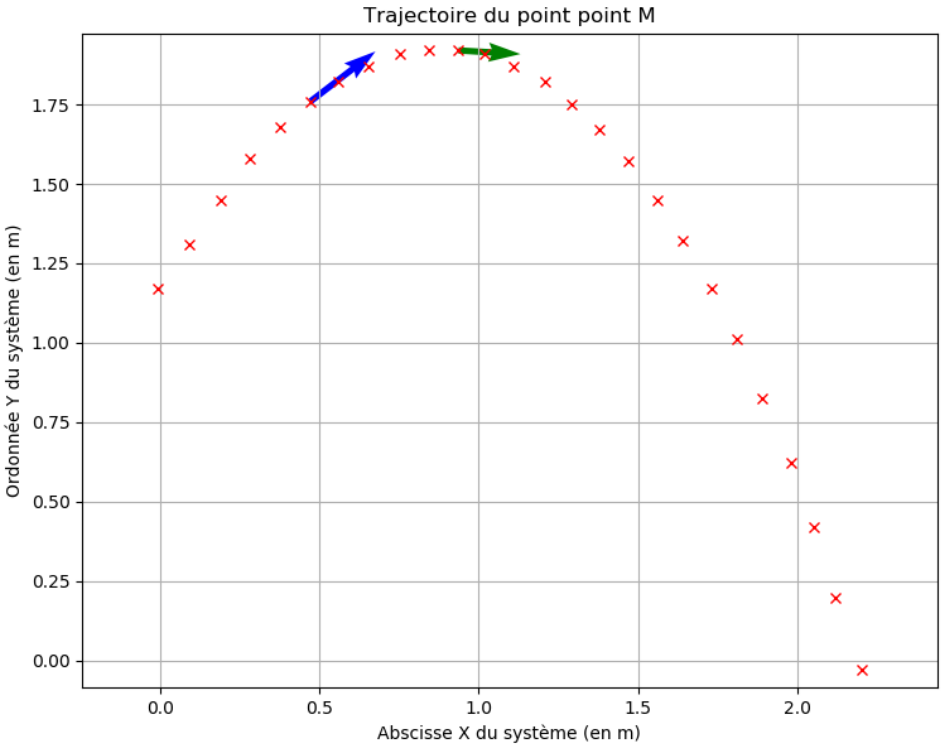
```
""" TRAVAIL 4:
Taper ci-dessous le code python permettant
de tracer le vecteur vitesse au point 5 en bleu"""
```

Pour tracer un vecteur en un point, il faut utiliser :
`draw_Vector(`*numéro du point* **`,`***Abscisse du vecteur* **`,`***Ordonnée du vecteur* **`,"k")`**.

8. Activités élèves

Pour finir, on demande de taper le code pour une généralisation en un point M_n

```
n=10
Vx = (x[n+1]-x[n-1])/(t[n+1]-t[n-1])
Vy = (y[n+1]-y[n-1])/(t[n+1]-t[n-1])
draw_Vector(n,Vx,Vy,"g")
```



8. Activités élèves

Activité élève:
Tracé et modélisation de la caractéristique d'un conducteur ohmique

Travail des élèves

- Mesures de U et I
- Saisies de U et I dans un programme Python
- Tracé de la courbe U=f=(I)
- Choix d'un modèle parmi 3 proposés
- Tracé de la courbe modélisée

Attendus officiels:
Représenter un nuage de points associé à la caractéristique d'un dipôle et modéliser la caractéristique de ce dipôle à l'aide d'un langage de programmation.

Durée *(bilan de 2 tests en MPS avec des élèves n'ayant jamais programmé):*

- Proposition du protocole de mesure + mesures de U et I: **1h**
- Travail sous Python: **1h**

8. Activités élèves

La partie Python: L'endroit où saisir/modifier le code est clairement identifié

Partie en rose (si EduPython) ou vert (si Spyder)

Les élèves doivent parfois modifier le code

```
""" TRAVAIL 1:
    ----------
Entrer , ci-dessous, entre les crochets
I = np.array([0.1, 0.2, 0.3])
U = np.array([2, 4, 6])

""" TRAVAIL 2:
    ----------
    Suivre les consignes écrites en rose ci-dessous: """
# Titre Du graphique (OPTIONNEL, peut-être supprimé)
""" Entrer ci-dessous, entre les guillemets, le titre du graphique SANS ACCENTS !!!! """
plt.title("Titre du graphique")

# Légende des axes (OPTIONNEL, peut-être supprimé)
""" Entrer ci-dessous les noms des grandeurs pour chaque axe
(remplacer "abscisse" et "ordonnee") SANS ACCENTS !!!! """
plt.xlabel("Abscisse")
plt.ylabel("Ordonnee")

""" TRAVAIL 3:
    ----------
    Suivre les consignes écrites en rose ci-dessous: """
# ########################################
# Tracé de la courbe expérimentale
""" Taper ci-dessous le code permettant de représenter U en fonction de I
sous forme de croix rouges"""


""" Modi
plt.text
```

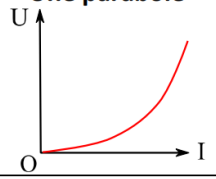
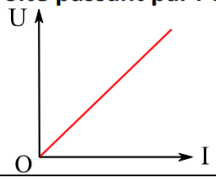
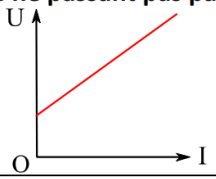
Les élèves doivent écrire l'instruction pour tracer la courbe (**aide fournie**)

Pour tracer la courbe représentant une grandeur Y en fonction d'une grandeur X, il faut taper : **`plt.plot(X,Y,"kx")`**

Pour le choix des couleurs et du dessins des points, voir annexe,

8. Activités élèves

Si la courbe ressemble à:

Une parabole	Une droite passant par l'origine	Une droite ne passant pas par l'origine
		
Il faut taper : `Umod = a*I*I`	**Il faut taper :** `Umod = a*I`	**Il faut taper :** `Umod = a*I + b`

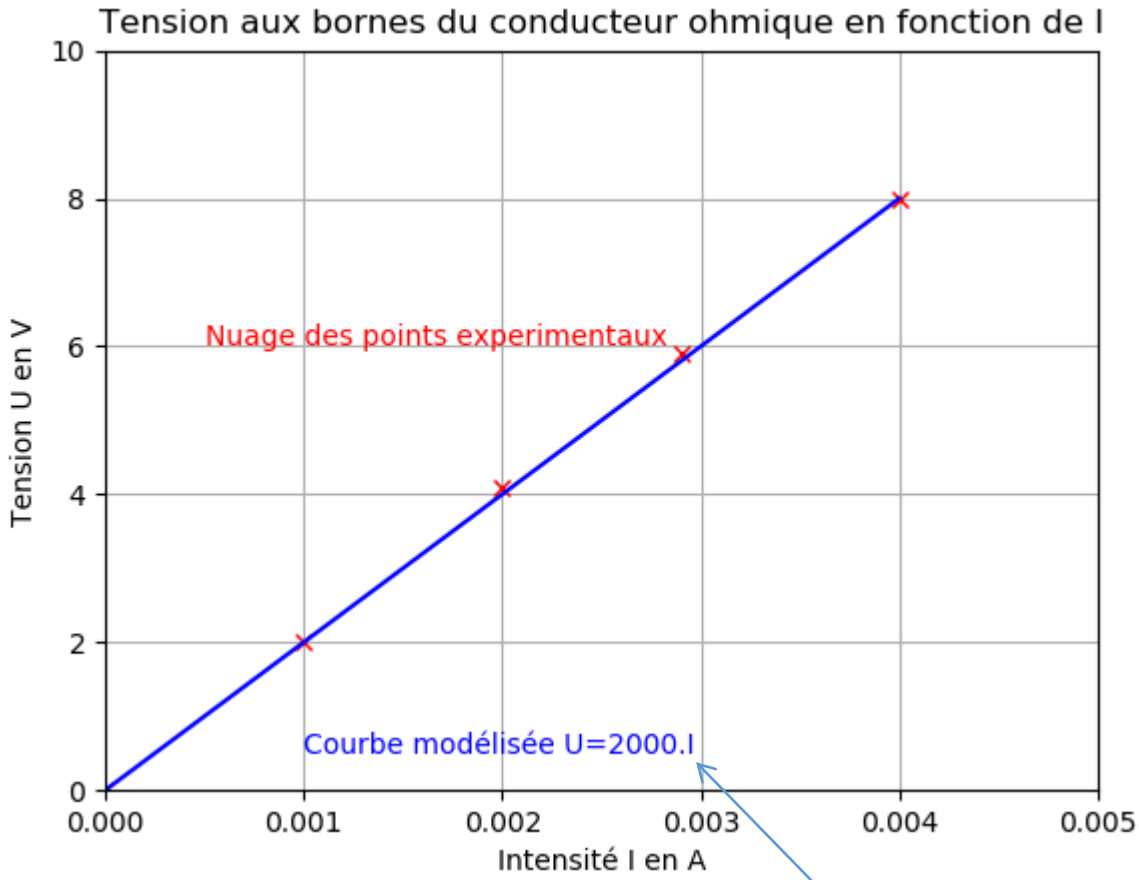
Les élèves analysent le nuage de points et tapent le code du modèle

Les élèves doivent écrire l'instruction pour tracer le modèle

```
""" TRAVAIL 4:
    ----------
    Suivre les consignes écrites en rose ci-dessous: """
# ######################################
# Tracé de la courbe modélisée

# Variables permettant d'ajuster le modèle:
a = 1000
b = 2

# Modèle choisi:
""" Taper ci-dessous le code du modèle qui correspond le mieux à votre cou


""" TRAVAIL 5:
    ----------
    Suivre les consignes écrites en rose ci-dessous: """
# ######################################
# Tracé de la courbe expérimentale
""" Taper ci-dessous le code permettant de représenter la courbe de mode
en bleu, points non affichés mais reliés par un trait"""


""" Modifier ci-dessous pour placer le texte aux coordonnées voulues et
plt.text (.3,2,"Courbe modélisée U="+str(a)+".I",color='g')
```

Enfin, ils font varier ***a*** pour que le modèle corresponde à leur nuage de points

8. Activités élèves



Comme U =R.I
Les élèves trouvent R

Olivier CHAUMETTE

Mathilde DIDIER-GLENAT
Jacques VINCE
Jean-Baptiste BUTET

Académie de Lyon
Version 1.3 – mai 2019