**FICHE de PRÉSENTATION d’activités**

|  |  |
| --- | --- |
| ***Niveau***  | ***Terminale spécialité*** |
| ***Séquence*** | ***AUCUNE (séance déconnectée faite en début d’année)*** |
| ***Titre de l’activité*** | ***Réactivation des savoir-faire en python nécessaires aux sciences physiques*** |
| ***Type d'activité*** | ***Programmation en ½ groupe*** |
| ***Références au programme*** | Notions et contenus | Capacités exigibles***Capacité numérique :*** *utilisation d’un langage de programmation* |
| ***Compétences mobilisées***  | ❑ Analyser/raisonner ❑ Réaliser ❑ Communiquer |
| ***Mise en œuvre*** | Pré-requis:- Notions de base de python, tableaux numpy, matplotlib… ou pas- Savoir se servir du mémento « Python pour le lycée » (fourni en pièce jointe). |
| **Durée : 1h** Cette séance se place en tout début d’année, soit sur une heure de TP soit en essayant de la faire exceptionnellement sur les heures classe entière. Elle a pour but de remobiliser les savoir-faire Python en faisant (re)découvrir l’éditeur python, la notion de tableau, l’ajout de données dans un tableau, le tracé de courbe.Cette séance doit permettre de gagner du temps lors des 1er TP utilisant Python car les élèves se seront ré-appropriés l’outil.La séance est volontairement « longue » pour que les élèves les plus rapides (ou en NSI) puissent avoir du travail pour 1h. La majorité des élèves arrivent à tracer la courbe et nommer les axes. |
| Contraintes matérielles : - salle informatique avec EduPython- notice « mémento Python pour le lycée » sur la paillasse (fournie en PJ) |
| *Liens photos* |  |
| *Auteur* | **olivier.chaumette@ac-lyon.fr** |
|  **pour le GRD groupe lycée de l’académie de LYON** |

**Fiche élève : activité**

Python est un langage de programmation. Il permet de donner une liste d'ordres à l'ordinateur. Cette suite d'ordres (appelés "instructions") s'appelle un programme. Python possède des instructions "toutes faites" très pratiques pour les scientifiques. Il permet de tracer des courbes, modifier rapidement les paramètres d'un calcul, tester des modélisations.

Son utilisation en sciences physiques n'a donc pas pour but d'apprendre à programmer mais de savoir utiliser un outil efficace et puissant (parmi d'autres que nous rencontrerons également: Latispro, Excel etc...)

**VOUS DISPOSEREZ TOUTE L’ANNÉE D’UN « MÉMENTO PYTHON POUR LE LYCÉE »**

**1. Premières lignes de code en python**

**1.** Ouvrir « **Edupython** », application qui va nous permettre de saisir les instructions de nos programmes python.

**2.** Menu **Fichier** >> **Sauvegarder sous…** et sauver le programme (vide pour l’instant) dans votre dossier personnel sous le nom « **00\_Introduction\_Python.py** »

**3.** Dans la partie à droite, nous allons saisir les instructions.

Nous allons commencer par écrire un commentaire : c’est une instruction commençant par un dièse (#) qui est ignorée par Python mais permet d’écrire des remarques en français dans notre code.

Taper :

**# Premier programme python de terminale spécialité**

**# par : (rajouter votre nom)**

**4.** Exécuter le programme (triangle vert en haut) : rien n’apparaît dans la console car nous n’avons tapé que les commentaires.

**2. Importation des « modules » permettant de tracer des courbes et faire des calculs**

Par défaut, Python ne possède pas les instructions permettant de tracer une courbe ou effectuer des calculs complexes. Il faut lui « importer » des « modules ». NUMPY permet de faire des calculs complexes et MATPLOTLIB permet de tracer des courbes.

Taper, tout au début de la partie droite (Attention à bien respecter majuscule/minuscule, c’est-à-dire la **casse**) :

**import numpy as np # ce module permet de faire des calculs**

**import matplotlib.pyplot as plt # ce module permet de tracer des courbes**

**3. Exemple de saisies de données « à la main » dans des tableaux**

Pour faire simple, un tableau correspond à une grandeur physique. Il peut contenir plusieurs valeurs.

Pour créer un tableau correspondant à la grandeur « T » (par exemple) et contenant les valeurs 1, 2 et 3, il faut taper :

**T = np.array([1,2,3])**

Créer 2 tableaux U et I contenant les valeurs suivantes (Attention, voir « erreurs classiques » ci-dessous) :

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***U en V*** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| ***I en A*** | 0 | 0,95x10-3 | 2,02x10-3 | 3,05 x10-3 | 3,96 x10-3 | 5,02 x10-3 | 5,94 x10-3 | 6,99 x10-3 | 8,02 x10-3 | 9,04 x10-3 |

***Erreurs classiques :***

- La virgule de nos nombres décimaux se tape avec un point.

- C’est bien une virgule qui sépare les éléments d’un tableau

- La puissance de 10 se tape « **e** » et pas de « chapeau ^ » : 2,1x10-3 se tape **2.1e-3**

- On ne tape pas les unités en python. On les fait figurer en commentaire : **# U en V # I en A**

**4. Exemples d’affichage des valeurs d’un tableau**

L’affichage en python se fait avec la fonction **print()**. Dans les parenthèses, on met ce que l’on veut voir afficher. Les guillemets sont réservés à des textes figés. Le résultat s’affiche dans la console.

**1.** Taper les instructions suivantes et, à côté de chacune d’entre elles, écrire un commentaire expliquant ce qu’elles font.

**print(U)**

**print(U[2])**

**print(I[2])**

**print("La 3ème valeur de la tension est:",U[2],"V")**

**2.** Taper les instructions permettant :

- d’afficher tout le tableau I

- d’afficher « La 8ème valeur de l’intensité vaut I= » … suivi de la valeur de I et de son unité.

**5. Saisie de formules**

En python, si on veut créer la grandeur R telle que R = U/I, il suffit de taper : **R = U/I**

Cela crée automatiquement un nouveau tableau « R» dont chacune des valeurs vaut le rapport des valeurs de U et I.

**1.** Taper dans python la formule donnant la puissance électrique P=U.I (le « multiplier » en python se tape **\***)

**2.** Faire afficher le tableau de puissance dans son ensemble.

**3.** Faire afficher : « La 2nde valeur de puissance est P= » … suivi de la bonne valeur de P et de l’unité.

**6. Tracé de courbes**

Pour tracer Y en fonction de X (où X et Y sont des tableaux), il faut taper : **plt.plot(X,Y)**

Il faut aussi, tout à la fin du programme, taper : **plt.show()** pour voir le graphique

Il est possible (voire conseillé) de rajouter un style : couleur, style de trait etc… Les styles possibles sont sur le memento.

Par exemple : **plt.plot(X,Y,"rx-")** trace Y en fonction de X en rouge, avec des « x » reliés.

**1.** Taper le code permettant de tracer U en fonction de I en bleu avec des « + » (non reliés). Ne pas oublier le plt.show()

**2.** Exécuter le programme puis **refermer la fenêtre « matplotlib »** pour pouvoir continuer le programme.

Pour nommer les axes, mettre une grille et un titre, il faut utiliser **plt.xlabel(), plt.grid(), plt.title()**

**3.** Utiliser le « mémento python» pour nommer les axes (grandeur et unité) et le graphique. Mettre une grille.

**7. Ajout d’une valeur à un tableau**

Pour ajouter la valeur « 8 » au tableau T, il faut taper : **T = np.append(T,8)**

**1.** Avant le tracé de la courbe (avant le « plot »), ajouter la valeur ***20 V*** au tableau U et la valeur ***19,9x10-3 A*** au tableau I.

**2.** Lancer le programme : la courbe tracée tient compte des nouvelles valeurs.

**8. Pour aller plus loin : une modélisation**

**1.** Calculer le coefficient directeur de la droite obtenu (« à la main »)

**2.** Créer une variable R qui vaut la valeur que vous venez de calculer (il suffit de taper : **R=** suivi de la valeur)

**3.** Créer un tableau Umod (qui signifie « U modélisé ») qui vaut R.I

**4.** Tracer Umod en fonction de I en rouge, sans croix mais relié.

**9. Encore un peu plus loin : légender les courbes**

Pour ajouter une légende aux courbes, il faut, dans chaque instruction plt.plot, rajouter **label="nom de la courbe"**

Et, juste avant le **plt.show()**, taper : **plt.legend()**

Créer une légende pour chaque courbe (« courbe expérimentale » et « courbe modélisée »)

**Fiche professeur**

Cette séance doit permettre de gagner du temps lors des 1er TP. Pour cela, les notions de base sont revues (utilisation de l’éditeur, exécution d’un programme etc…). Les erreurs « classiques » (indentation, parenthèses…) seront faite lors de cette séance par la plupart des élèves et ainsi, ils ne les commettront plus lors des TP.

Attention : La notion de boucle n’est pas abordée.

La séance part d’une « page blanche » dans Edupython.

1h suffit. Les plus rapides terminent. Les élèves les moins à l’aise avec l’outil tracent juste la courbe.

Il peut être bon de les encourager à télécharger Python chez eux pour finir l’activité et/ou s’entraîner.

# Premier programme python de terminale spécialité

# par : Olivier CHAUMETTE

**import** numpy **as** np # ce module permet de faire des calculs

**import** matplotlib**.**pyplot **as** plt # ce module permet de tracer des courbes

U**=**np**.**array**([**0**,**1**,**2**,**3**,**4**,**5**,**6**,**7**,**8**,**9**,**10**])**

I**=**np**.**array**([**0**,**.95e-3**,**2.02e-3**,**3.05e-3**,**3.96e-3**,**5.02e-3**,**5.94e-3**,**6.99e-3**,**8.02e-3**,**9.04e-3**,**9.99e-3**])**

**print(**U**)** # affiche le tableau U entier sans mise en forme

**print(**U**[**2**])** # affiche le 3ème élément de U. Remarquer le décalage entre l'index (2) et le nième élement (3ème)

**print(**I**[**2**])** # affiche le 3ème élément de I

**print(**"3ème valeur de la tension est:"**,** U**[**2**],**"V"**)** # affichage d'un texte suivi d'une valeur suivi d'un texte à nouveau. Séparation par des virgules

# exercice:

**print(**I**)**

**print(**"La 8ème valeur de l'intensité est: I = "**,** I**[**7**],**"A"**)**

# exercice formules:

P **=** U**\***I

**print(**P**)**

**print(**"La 2nde valeur de puissance est P="**,**P**[**1**],**"W" **)**

# ajout de valeurs aux tableaux

U**=**np**.**append**(**U**,**20**)**

I**=**np**.**append**(**I**,**19.9e-3**)**

# tracé de U en fonction de I

plt**.**plot**(**I**,**U**,**"b+"**,**label**=**"Courbe expérimentale"**)**

# axes, titre, grille

plt**.**xlabel **(**"I en A"**)**

plt**.**ylabel**(**"U en V"**)**

plt**.**title**(**"U en fonction de I"**)**

plt**.**grid**()**

R**=**1000 # calculé à partir de la courbe expérimentale "à la main"

Umod**=**R**\***I # formule de U modélisé

plt**.**plot**(**I**,**Umod**,**"r-"**,** label**=**"Courbe modélisée"**)** # tracé de Umod en fonction de I

plt**.**legend**()**

plt**.**show**()**