**FICHE de PRÉSENTATION d’activités**

|  |  |
| --- | --- |
| ***Niveau***  | ***1ère spécialité*** |
| ***Séquence*** | ***Aspects énergétiques des phénomènes mécaniques*** |
| ***Titre de l’activité*** | ***Etude énergétique d’un système mécanique******à l’aide d’un programme Python*** |
| ***Type d'activité*** | ***Expérimentale et programmation en ½ groupe*** |
| ***Références au programme*** | Notions et contenus*Énergie potentielle. Cas du champ de pesanteur terrestre.**Établir et utiliser l'expression de l'énergie potentielle de pesanteur pour un système au voisinage de la surface de la Terre.**Énergie mécanique. Conservation et non conservation de l’énergie mécanique.* | Capacités exigibles***Capacité numérique :*** *Utiliser un langage de**programmation pour effectuer le bilan énergétique d’un**système en mouvement.* |
| ***Compétences mobilisées***  | ❑ Analyser/raisonner ❑ Réaliser |
| ***Mise en œuvre*** | Pré-requis:Quelques notions de python (variable, exécution d’un programme sous Edupython) |
| Durée : 1h |
| Contraintes matérielles : - salle informatique avec EduPython- vidéo lancer parabolique et ou chute libre sans Vitesse initiale- Logiciel de pointage : Latipro ou avimeca ou avistep |
| *Liens photos* | *aucun* |
| *Auteur* | **olivier.chaumette@ac-lyon.fr** |
|  **pour le GRD groupe lycée de l’académie de LYON** |

**Fiche élève : activité**

Nous allons faire une étude énergétique de la chute libre d’une balle de masse m = 200g à l’aide de LATISPRO.

***1. Positions de la balle : Compétence REALISER***

Ouvrir la vidéo « **Chute\_libre\_25fps.avi** ». Choisir l’étalon (la règle mesure 1 mètre) et prendre comme origine un point **au niveau du sol**. Pointer les positions de la balle.

***2. Exportation des données de pointage : Compétence REALISER***

Menu **Ficher** >> **Exportation**. **“Ajouter toutes les courbes”**. Vérifier que TXT est coché (dans “Format”) et que le **Choix des séparateurs** est **Virgule** (pour “Décimal”) et **Point Virgule** (pour “Entre les données”). Exporter dans votre dossier personnel.

***3. Etude énergétique à l’aide d’un programme Python: Compétence REALISER et ANALYSER***

**1.** Lancer l’éditeur Python (Edupython.exe) et ouvrir le programme **« 1ere\_spe\_Energie.py ».**

*Pour l’instant, le programme n’affiche rien. Vous devrez le modifier.*

*Les* ***différentes parties*** *du programme sont repérées par des commentaires précédés du symbole dièse : #*

**2. TRAVAIL 1 :** **Création des variables contenant masse et accélération de la pesanteur.** Suivre les consignes écrites en rose dans le programme au niveau de « TRAVAIL 1 ». Les variables devront avoir des noms explicites.

**3. TRAVAIL 2 :** **Création des tableaux contenant les différentes énergies.** Suivre les consignes en rose au niveau de « TRAVAIL 2 ». Il faudra créer 3 variables (une par type d’énergie). Les grandeurs que vous avez à votre disposition sont x , y , t et V (norme de la vitesse). Attention à la casse ! (c'est-à-dire la différence majuscule/minuscule)

**4. TRAVAIL 3 :** **Tracé des courbes expérimentales**:

Pour tracer la courbe représentant une grandeur Y en fonction d’une grandeur X, il faut taper en Python :

**plt.plot(X,Y,“kx”,label=“nom de la grandeur représentée en ordonnée”)**

*“kx” signifie que les points affichés seront noirs (“k”) et représentés par des croix (“x”)*

Dans le programme, en dessous des consignes correspondantes à TRAVAIL 3, taper le code permettant de tracer l’évolution de chaque énergie en fonction de Ia date t avec des points de couleur différente représentés par une croix sans être reliés (voir annexe).

**5.** Exécuter le programme (petit triangle vert en haut de l’écran). Une fenêtre vous permet d’aller chercher le fichier que vous avez exporté depuis LATISPRO. Les courbes doivent s’afficher dans une nouvelle fenêtre.

**6. TRAVAIL 4 :** **Légendes du graphique.** Suivre les consignes en rose au niveau de « TRAVAIL 4 ».

**7.** Commenter ces courbes en termes de transfert d’énergie

**ANNEXE -** *Quelques options de mise en forme du tracé avec le module MATPLOTLIB utilisé par Python*

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

**Fiche professeur**

Une fiche expliquant comment exporter des données depuis AVIMECA ou AVISTEP est fournie à a fin de ce document. Cela permettra au professeur d’adapter la 1ère partie de l ‘énoncé au logiciel qu’il utilise.

L’auteur a défini préalablement, auprès des élèves, la vitesse Vx comme étant  (pour être cohérent par rapport à la définition mathématique). Cependant il a fait le choix d’utiliser la dérivée numérique centrée : Vx =  dans le code python car, pour un pointage, cette méthode donne de meilleurs résultats. Et elle reste, comme Vx = , simplement une méthode permettant d’approcher la vitesse instantanée (que l’on ne peut pas calculer à partir de points expérimentaux)

***ATTENTION :*** le fichier **import\_donnees\_meca.py** doit être présent dns le même dossier que le fichier python que complètent les élèves (car c’est lui qui permet de transformer le fichier exporté par le logiciel en tableaux numpy dans Python)

***Voici ci-dessous le code corrigé :***

# Programme Python permettant de calculer la norme de la vitesse et les énergies

# potentielles, cinétiques, mécaniques à partir d'un pointage.

# à destination des élèves qui peuvent modifier le code sous les parties roses

# Le code est volontairement simplifié pour qu'un élève puisse se

# l'approprier et surtout "coder"

# X,Y et t sont importés dans des listes Numpy

# L'auteur a fait le choix de calculer Vx, Vyx et V

# mais il est possible de le faire faire aux élèves

# ##########################################################################

# O. CHAUMETTE - Lycée JP SARTRE - 69500 BRON - olivier.chaumette@ac-lyon.fr

# version 2 - mars 2019

# ##########################

# importation de NUMPY (pour gestion tableaux et calculs) sous l'alias "np"

**import** numpy **as** np

# importation de PYPLOT (de MATPLOTLIB, pour le tracé de courbes) sous l'alias "plt"

**import** matplotlib**.**pyplot **as** plt

# modules permettant d'importer des données

# le fichier "import\_donnees\_meca.py" est indispensable

**from** import\_donnees\_meca **import** traiteDonnees

# Initialisation des tableaux numpy contenant x,y,t , la norme de V et ses coordonées

x**=**np**.**array**([])**

y**=**np**.**array**([])**

t**=**np**.**array**([])**

V**=**np**.**array**([])**

Vx**=**np**.**array**([])**

Vy**=**np**.**array**([])**

""" ---------------------------------

 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' (crééer 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**)**

Nbre\_Mesures**=**len**(**x**)**

# Calcul de la vitesse

# ################################

# cette partie peut être traitée éventuellement par les élèves

# on parcourt toutes les mesures (de la 2ème à l'avant-dernière) pour calculer Vx et Vy

# d'où ce "1" et "Nbre\_Mesures-1" (rappel: 1ère mesure = index 0)

**for** n **in** range**(**1**,**Nbre\_Mesures**-**1**):**

 # pour chaque mesure n, on ajoute au tableau Vx la coordonnée de V suivant X

 Vx**=**np**.**append**(**Vx**,(**x**[**n**+**1**]-**x**[**n**-**1**])/(**t**[**n**+**1**]-**t**[**n**-**1**]))**

 Vy**=**np**.**append**(**Vy**,(**y**[**n**+**1**]-**y**[**n**-**1**])/(**t**[**n**+**1**]-**t**[**n**-**1**]))**

 # On créée le tableau V, norme de la vitesse

 V=np.sqrt(Vx\*\*2+Vy\*\*2)

# ################################

# Modification des listes y et t pour que le nombre de points soit le même que V

# ################################

# ATTENTION ! le tableau V contiendra 2 éléments de moins que x, y ou t car, pour calculer V, on utilise le point d'avant et celui d'après.

# donc pas de vitesse en 0 et au tout dernier t.

# Afin de pouvoir ajouter des tableaux entre eux, il faut qu'ils aient le même nombre d'éléments. Il faut donc, dans y et t (x inutile pour Ep), enlever le dernier élément et le 1er.

# sauvegarde des listes y et t avant de les modifier (toujours utile...)

sauv\_y**=**y

sauv\_t**=**t

# on enlève le dernier élément de y et t

y**=**np**.**delete**(**y**,**len**(**y**)-**1**)**

t**=**np**.**delete**(**t**,**len**(**t**)-**1**)**

# et le premier:

y**=**np**.**delete**(**y**,**0**)**

t**=**np**.**delete**(**t**,**0**)**

""" ###########################################

Ici commence le travail à réaliser par les élèves

# Initialisation des variables pour l'exercice

""" TRAVAIL 1: masse et accélération de la pesanteur

 Rentrer ci-dessous les valeurs de m et g dans les unités du système international"""

m**=** 0.200

g**=**9.81

# #####################################

# Calculs des énergies

""" TRAVAIL 2:

Taper ci-dessous le code Python permettant de calculer les valeurs d'Ec, Ep et Em à chaque date.

Grandeurs disponibles:

 - t : date

 - x : l'abscisse du système à la date t

 - y : l'ordonnée donc l'altitude du système à t

 - V : norme du vecteur vitesse à la date t"""

Ec**=**0.5**\***m**\***V**\***V

Ep **=** m**\***g**\***y

Em**=**Ec**+**Ep

# #####################################

# Tracé

""" TRAVAIL 3:

code Python permettant de tracer Ec, Ep et Em en fonction de la date"""

plt**.**plot**(**t**,**Ec**,**'xr'**,** label**=**"Energie cinétique"**)**

plt**.**plot**(**t**,**Ep**,**'xb'**,** label**=**"Energie potentielle"**)**

plt**.**plot**(**t**,**Em**,**'xk'**,** label**=**"Energie mécanique"**)**

""" TRAVAIL 4:

taper, à la place des points entre guillemets, les légendes pour les axes, le titre du graphique"""

plt**.**title**(**"Variation de l'énergie en fonction du temps"**)**

plt**.**xlabel**(**"date t (s)"**)**

plt**.**ylabel**(**"Energie (J)"**)**

""" fin du travail """

# On affiche une grille

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

# On affiche le commentaire

plt**.**legend**(**bbox\_to\_anchor**=(**1.05**,** 1**),**loc**=**'upper left'**,**borderaxespad**=**0.**)**

# affichage de la figure bien adapté à la fenêtre la fenêtre

plt**.**tight\_layout**()**# on affiche la fenêtre

#plt.get\_current\_fig\_manager().window.state('zoomed') # <-- Affiche la fenêtre maximisée sous windows uniquement

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

**MODE D’EMPLOI des programmes Python de mécanique (formation 2019)**

*L’objectif de cette fiche est d’expliquer comment utiliser les programmes Python de mécanique proposés lors de la formation Académique de 2019.*

**1. Pourquoi cette notice ?**

Les programmes py de mécanique proposés nécessitent l’importation de données de pointage (abscisse, ordonnée et date). **Une ligne de ces programmes dit être modifiée par le professeur** en tenant compte du logiciel de pointage utilisé par les élèves.

**Attention, pour tous les programmes Python de mécanique, le fichier “import\_donnees\_meca.py” doit être présent dans le même dossier que le programme (car c’est lui qui permet d’importer les données)**

**2. Choix, par le professeur, de la manière d’importer des données dans le programme**

**La ligne à modifier se situe, dans le programme, au niveau du commentaire “CHOIX DU PROFESSEUR” :**

Dans la version par défaut, la ligne est la suivante : **Logiciel\_Utilise = 'latispro'**

S’il le souhaite, le professeur peut remplacer **‘latispro’** par un des choix proposés : **‘pymecavideo’** ou **‘avimeca’** ou **‘avistep’** ou **‘equations’** ou **‘entree’** (ATTENTION : bien respecter la casse et ne pas mettre d’accents)

**‘latispro’** ou **‘pymecavideo’** ou **‘avimeca’** ou **‘avistep’**: nom du logiciel utilisé.

**‘equations’** : le professeur saisit les équations horaires dans le fichier “import\_donnees\_meca.py” (voir explication § 7)

**‘entree’** : le professeur saisit les données « à la main » dans le fichier “import\_donnees\_meca.py” (voir explication § 8)



**3. Export des données depuis LATISPRO**

**1.** Une fois le pointage réalisé (Mouvement X, Mouvement Y et Temps seulement), choisir Menu **Fichier** >> **Exportation**.

**2. “Ajouter toutes les courbes”**. Vérifier que TXT est coché (dans “Format”) et que le **Choix des séparateurs** est **Virgule** (pour “Décimal”) et **Point Virgule** (pour “Entre les données”).

**4. Export des données depuis AVISTEP**

**1.** Une fois le pointage réalisé, choisir Menu **Résultats** >> **Tableau des valeurs**.

**2.** Dans la fenêtre Tableau des valeurs, choisir Menu **Fichier** >> **Enregistrer**.



**5. Export des données depuis AVIMECA**

**1.** Une fois le pointage réalisé, choisir Menu **Fichier** >> **Mesures >> Enregistrer dans un Fichier >> Format Texte (\*.txt)**.

**2.** Dans la fenêtre, vérifier que « **Tabulation** » est bien coché. Puis **OK**.

**6. Export des données depuis PYMECAVIDEO**

PyMecaVideo est un logiciel gratuit, libre et multiplateforme disponible à l’adresse : <http://outilsphysiques.tuxfamily.org/wiki/index.php?title=Pymecavideo>

Une fois le pointage réalisé, choisir Menu **Fichier >> Enregistrer les données…..**

**7. Choix EQUATIONS**

**1.** Charger le programme “**import\_donnees\_meca.py**” dans l’éditeur Python (EduPython ou Spyder par exemple)

**2.** Modifier la durée et les équations horaires dans le code suivant :

**elif origine=='equation':**

 **""" CAS 2:**

 **X ET Y ISSUES D'EQUATIONS HORAIRES DONNEES PAR LE PROFESSEUR**

 **(méthode rapide)"""**

 **tmax = 2 Saisir ici la durée maximale du temps en seconde**

 **Nbre\_Mesures = 20 Saisir ici le nombre de points voulus. L’intervalle de temps sera tmax/Nombre\_Mesures**

 **t = np.linspace(0, tmax,Nbre\_Mesures)**

 **x=2\*t Saisir ici l’équation horaire x(t)**

 **y=-4.9\*t\*t+2\*t+1 Saisir ici l’équation horaire y(t)**

**3.** Enregistrer le programme “**import\_donnees\_meca.py**” (sans changer son nom)

**8. Choix ENTREE**

**1.** Charger le programme “**import\_donnees\_meca.py**” dans l’éditeur Python (EduPython ou Spyder par exemple)

**2.** Modifier le contenu des tableaux x, y et t dans le code suivant :

**if origine=='entree':**

 **""" CAS 1:**

 **X ET Y ISSUES DU POINTAGE et entrées à la main"""**

 **x=np.array([0,0,0,0,0,0,0,0,0]) Saisir ici les valeurs de x (séparées par une virgule)**

 **y=np.array([9.86E-3,-5.92e-2,-1.48E-1,-2.76e-1,-4.24E-1,-6.02E-1,-8.48E-1,-1.07,-1.39])**

**Saisir ici les valeurs de y (séparées par une virgule)**

 **t=np.array([0,0.033,0.067,0.1,0.133,0.167,0.2,0.233,0.267]) Saisir les valeurs de t**

**3.** Enregistrer le programme “**import\_donnees\_meca.py**” (sans changer son nom)

**9. En cas d’erreurs d’importation**

Il peut arriver que l’exportation par les logiciels enregistre des données absurdes ou fausses. Ces données font que Python renvoie une erreur.

Dans ce cas, le professeur peut ouvrir le fichier exporté avec le bloc-note Windows et supprimer « à la main »les lignes à l’origine de l’erreur.

***Exemple d’un fichier Latispro incorrect :***

Depuis le bloc-note, supprimer cette ligne « à la main » et enregistrer le fichier

**Temps;Mouvement X;Temps;Mouvement Y**

**0;0,00902378999179662;0;1,18229227704207**

**NAN ;NAN ;NAN,NAN**

**0,04;0,0986757295206844;0,04;1,32995429508965**

**0,08;0,20414859955467;0,08;1,47761631313723**