

Fiche pour l'enseignant

Mission Space Lab

Phase 1 : Introduction à la programmation Python



Dans cette première phase de préparation, les élèves acquièrent les compétences minimales en programmation nécessaires pour mener à bien le défi de la Mission Space Lab: estimer la vitesse de la Station Spatiale Internationale à partir de photos prises à son bord en programmant le mini-ordinateur Astro Pi.



Avec le soutien financier de



Mission Space Lab

Phase 1 : Introduction à la programmation Python

Guide de l'enseignant

Dans cette première phase de préparation, les élèves acquièrent les compétences minimales en programmation nécessaires pour mener à bien le défi de la Mission Space Lab : estimer la vitesse de la Station Spatiale Internationale à partir de photos prises à son bord en programmant le mini-ordinateur Astro Pi.

Table des matières

VUE D'ENSEMBLE DE L'ACTIVITÉ	3
INTRODUCTION	4
INSTALLATION DE THONNY	6
ACTIVITE AVEC GOOGLE COLLAB	10
PRÉSENTATION	10
SAUVEGARDER SON TRAVAIL	11
EXPORTER LE CODE ET SAUVEGARDE DU NOTEBOOK	11
MISE EN PLACE EN CLASSE.....	11
COMMENTAIRES & REPONSE AUX EXERCICES	12
SECTION 2 : LES VARIABLES	12
EXERCICES DE LA SECTION 2.1	12
SECTION 4 : LES FONCTIONS	13
EXERCICES DE LA SECTION 4.1	14
EXERCICES DE LA SECTION 5.4.....	14
LIENS	16
PROJET MISSION SPACE LAB	16
RESSOURCES PYTHON	16
IDE THONNY	16

Vue d'ensemble de l'activité

Public

S3 à S6

Matières

Informatique

Durée

1 à 2 périodes

Résumé

Dans cette activité, une introduction aux éléments les plus importants du langage Python est présentée. Cela permet d'acquérir les notions minimales pour affronter les activités suivantes de Mission Space Lab.

Objectifs d'apprentissage

-

Matériel

- Ordinateur
- Accès à Internet
- Compte Google (Gmail ou autre)

(facultatif)

Métiers STEM en lien

- Informaticien
- Chercheur
- Ingénieur
-

Auteurs : La Scientotheque (ESERO Belgium)

Date de publication : Septembre 2024

INTRODUCTION

Mission Space Lab

Le projet Mission Space Lab, proposé par ESERO aux professeurs du secondaire, vise à éveiller l'intérêt des élèves pour les sciences spatiales et l'exploration de l'espace. Il s'agit d'une initiative concrète qui permet aux jeunes de se confronter à la recherche scientifique en conditions réelles, à bord de la Station spatiale internationale (ISS).

Ce projet poursuit plusieurs objectifs pédagogiques ambitieux :

- **Sensibiliser** les élèves aux merveilles de l'univers et aux enjeux de l'exploration spatiale.
- **Permettre** aux jeunes de réaliser des expériences scientifiques concrètes en microgravité, un environnement hors du commun.
- **Développer** leurs compétences en programmation, analyse de données et résolution de problèmes.
- **Favoriser** la collaboration et le travail en équipe pour mener à bien un projet scientifique complexe.

C'est pourquoi, le projet est séparée en trois phases distinctes :

1. **Une introduction à la programmation Python** : elle permet de construire des fondations solides pour entamer les prochaines phases tout en développant des compétences utiles.
2. **Une première estimation de la vitesse de l'ISS** : Cette activité propose un parcours guidé pour l'estimation de l'ISS permettant de construire une base solide pour la suite.
3. **Des activités d'approfondissement** : elles visent une meilleure compréhension de la problématiques par une étude sur papier et l'exploration de pistes de solutions pour l'amélioration de l'estimation de la vitesse de l'ISS

La première phase et lien avec les phases ultérieures

L'introduction à la programmation Python dans le projet Mission Space Lab, bien que constituant la première phase, joue un rôle crucial dans la réussite globale du projet et s'inscrit parfaitement dans sa cohérence pédagogique.

La programmation Python est le langage de programmation choisi pour développer le programme informatique qui sera exécuté sur l'Astro Pi à bord de l'ISS. Sans une connaissance de base de Python, les élèves ne seraient pas en mesure de comprendre les instructions nécessaires à la collecte et à l'analyse des données scientifiques.

La phase d'introduction à Python permet aux élèves d'acquérir les concepts fondamentaux de la programmation, tels que les variables, les structures de contrôle, les fonctions et les modules. Ces notions sont indispensables pour la compréhension et la mise en œuvre du programme informatique qui sera développé par la suite. L'apprentissage de la programmation favorise le développement de compétences transversales essentielles, telles que la logique, la résolution de problèmes, la créativité et l'adaptabilité. Ces compétences seront précieuses pour les élèves tout au long du projet Mission Space Lab et au-delà.

La première phase d'introduction constitue le socle sur lequel les élèves s'appuieront pour estimer la vitesse de l'ISS et mener des activités d'approfondissement. Commencer par une introduction à la programmation permet de susciter l'intérêt des élèves et de les motiver à s'engager pleinement dans le projet.

Outils mis à disposition

Il est possible de réaliser cette activité de deux manières différentes :

- **Avec Google Collab** : c'est un environnement de programmation en ligne accessible depuis un navigateur web et sans aucune installation. Son avantage principal est de permettre de mélanger texte et code. Le code de l'activité peut ainsi être progressivement écrit et testé dès qu'il est expliqué, évitant de d'éparpiller son attention sur plusieurs supports/environnements.
- **Avec Thonny** : c'est l'environnement de développement intégré (IDE) préconisé par l'ESA pour la réalisation de l'ensemble du projet. Il permet l'exécution de code localement sans connexion internet et propose un simulateur Astro Pi pour tester le code dans des conditions réalistes avant l'envoi à l'ESA. Une fiche pédagogique fournit les instructions

Caractéristique	Google Collab	IDE Thonny
Pas d'installation requise	✓	✗
Simulateur Astro Pi	✗	✓
Configuration non requise	✓	✗
Internet non requis	✗	✓
Fiche élève non requis	✓	✗
Compte Google non requis	✗	✓

La première et deuxième phase sont proposées sous forme de notebook Google Collab ou sous la forme d'une fiche pédagogique. Pour la première phase, il est plus intéressant de se focaliser principalement sur l'apprentissage de Python avec le notebook Google Collab pour ne pas partager son attention entre la fiche pédagogique et l'interface Thonny.

Pour la deuxième phase, il devient plus intéressant de s'habituer plus proprement à l'interface Thonny pour pouvoir se préparer à la réalisation du challenge qui en fera grand usage. Cependant, un notebook Collab est mis à disposition si votre objectif est de rapidement se familiariser avec la méthode d'estimation de la vitesse proposée avant de coder de manière plus conventionnelle.

Voici le lien vers le notebook Google Collab de cette première phase :

https://colab.research.google.com/drive/1F_sjj13imZNYCzcnPuMY5_sSve3lYNRH?usp=sharing

INSTALLATION DE THONNY

Installer Thonny sur un Raspberry Pi

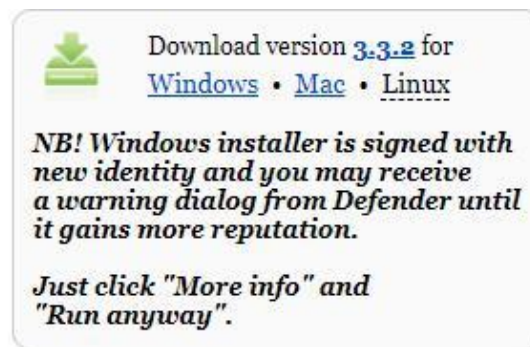
Thonny est déjà installé sur Raspberry Pi OS, mais peut avoir besoin d'être mis à jour pour avoir la dernière version.

Ouvrez une fenêtre du terminal en cliquant sur l'icône située en haut à gauche de l'écran ou en appuyant simultanément sur les touches Ctrl+Alt+T. Dans la fenêtre, tapez la commande suivante pour mettre à jour votre système d'exploitation et Thonny :

```
sudo apt update && sudo apt upgrade -y
```

Installer Thonny sur d'autres systèmes d'exploitation

Sous Windows, macOS et Linux, vous pouvez installer la dernière version de l'IDE Thonny ou mettre à jour la version existante. Dans un navigateur Web, accédez à **thonny.org** (<https://thonny.org/>) Dans le coin supérieur droit de la fenêtre du navigateur, vous verrez des liens de téléchargement pour Windows et macOS, et les instructions pour Linux. Téléchargez les fichiers appropriés et exécutez-les pour installer Thonny.



Ouvrir Thonny

Ouvrez Thonny depuis votre lanceur d'application. Vous devriez obtenir une fenêtre qui ressemble à ceci :



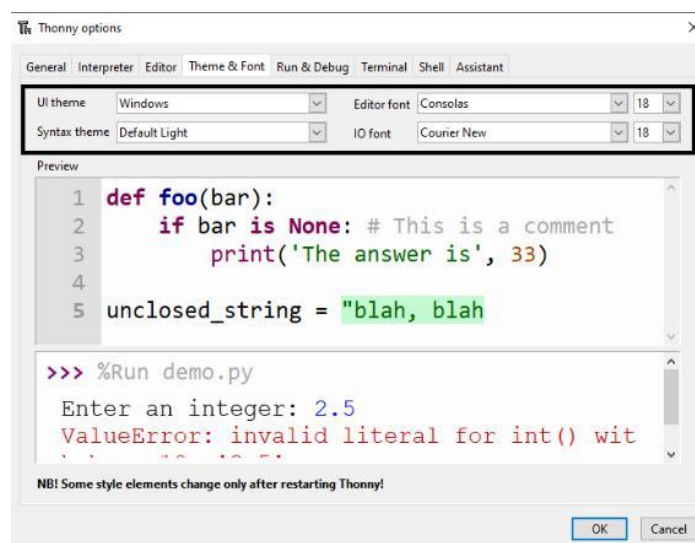
Vous pouvez utiliser Thonny pour écrire du code Python standard. Saisissez ce qui suit dans la fenêtre principale, puis cliquez sur le bouton Run (Exécuter) (il vous sera demandé d'enregistrer le fichier).

```
print('Hello World!')
```

Modifier le thème et la police dans Thonny

Thonny vous permet de modifier le thème et la police du logiciel. Cette fonctionnalité permet d'augmenter la taille de la police et de modifier les couleurs de l'arrière-plan et du texte en fonction de vos besoins. Pour modifier le thème et la police :

- Cliquez sur Tools (Outils) -> Options.
- Cliquez sur l'onglet Theme & Font (Thème et Police).
- Cliquez sur les cases déroulantes pour chaque option jusqu'à ce que vous trouviez les paramètres qui correspondent le mieux à vos besoins.

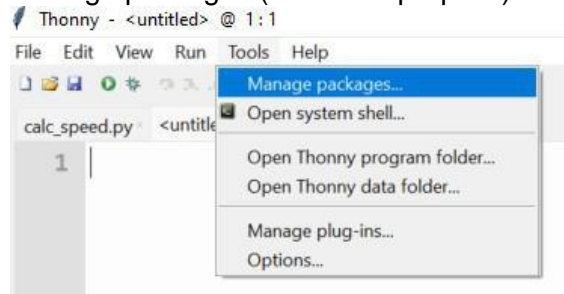


- Cliquez sur OK lorsque vous avez terminé.

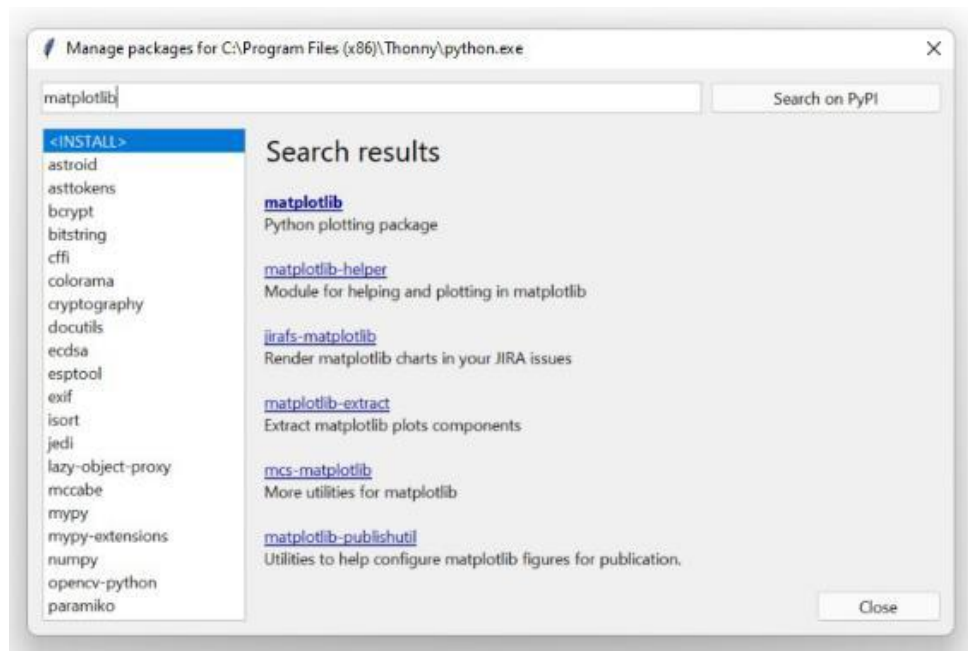
Attention : Privilégiez les polices simples et claires. Si vous utilisez une police de type écriture manuelle, cela peut rendre la lecture et le débogage difficiles.

Installer des paquets Python avec Thonny

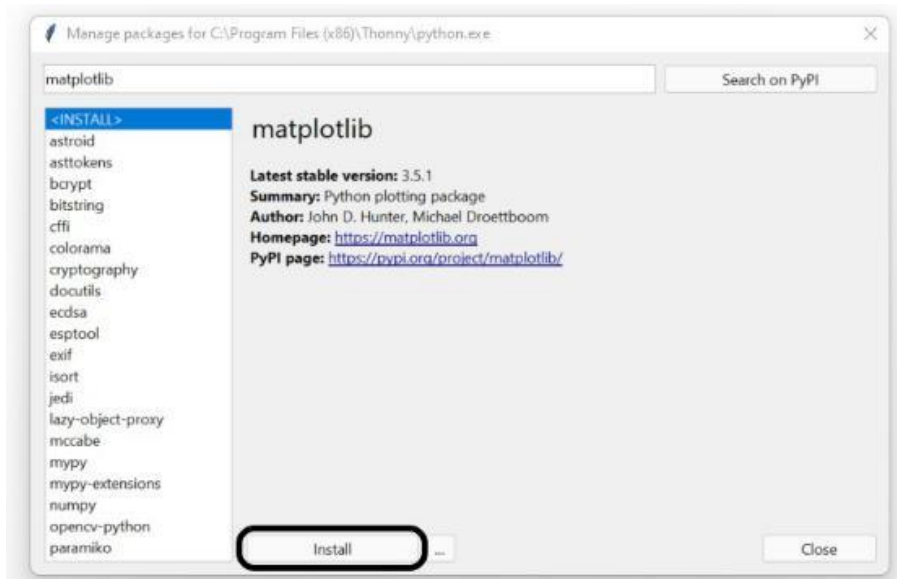
Ouvrez Thonny depuis votre menu d'applications. Cliquez sur Tools (Outils) dans la barre de menu, puis sélectionnez Manage packages (Gérer les paquets).



Utilisez la zone de recherche pour trouver le paquet que vous recherchez.



Sélectionnez le paquet que vous souhaitez installer, puis cliquez sur le bouton **Install** (Installer).



Une fenêtre s'ouvre et affiche l'avancement de l'installation de votre paquet. Une fois l'installation terminée, le paquet sera disponible dans vos programmes et vous pourrez l'utiliser.



Dans le cadre de la Mission Space Lab, les paquets suivants seront nécessaire pour la bonne réalisation du projet (ceux à installer en priorité pour la suite sont en gras) :

- **thonny-astro-pi-replay** (cf note en bas)
- **astro-pi-replay**
- picamzero
- sense-hat
- **skyfield**
- gpiozero
- **numpy**
- scipy
- pandas
- logzero
- matplotlib
- **pillow**
- **opencv-python**
- **exif**
- scikit-learn
- scikit-image
- reverse-geocoder



Dans le cadre du challenge Mission Space Lab, seuls les codes utilisant la liste limitée de modules autorisés pourront être acceptés et envoyés à la Station Spatiale Internationale. [Une liste des modules interdits est listé ici.](#)

Seule exception, le paquet **thonny-astro-pi-replay** doit être installé en allant sur **Outils > Gérer les plugins** au lieu de **Outils > Gérer les paquets**.

Pour cette première phase, aucun module particulier sera nécessaire mais le sera pour la seconde phase.

ACTIVITE AVEC GOOGLE COLLAB

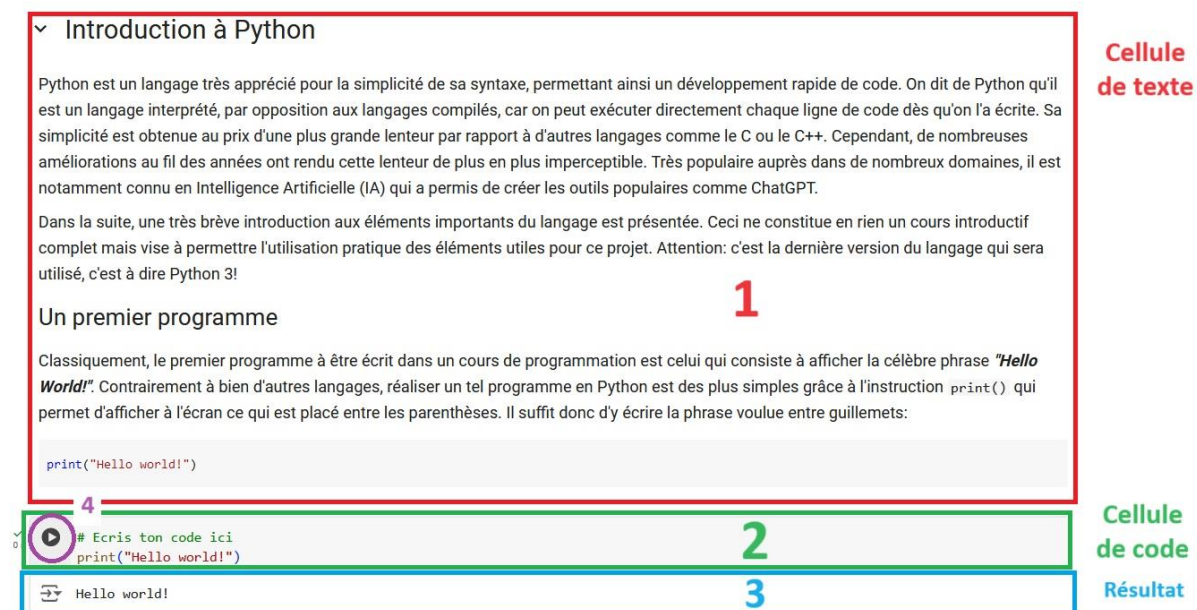
Présentation

Google Colab, ou Colaboratory, est une plateforme en ligne gratuite qui permet d'exécuter du code Python dans un environnement cloud. C'est comme un ordinateur puissant dans votre navigateur, accessible à tous et sans installation !

Parfait pour l'apprentissage automatique, la science des données et l'enseignement, Colab offre des notebooks Jupyter qui vous permettent d'écrire et d'exécuter du code, d'analyser des données, de créer des visualisations et bien plus encore. Le tout, avec l'accès à des GPU et TPU puissants pour le développement en Intelligence Artificielle.

Un notebook Jupyter est constitué de *cellules* de deux types différents:

- **Cellule de texte (voir 1)** : Elles contiennent du texte formaté, des images, des liens et des formules mathématiques écrite en Markdown. Vous pouvez les utiliser pour rédiger des explications, des titres ou des commentaires.
- **Cellule de code (voir 2)** : Elles contiennent du code exécutable dans un langage de programmation spécifique (Python, R, etc.). Vous pouvez les utiliser pour analyser des données, créer des visualisations, ou implémenter des algorithmes.



The screenshot shows a Google Colab notebook interface. At the top, there is a red-bordered cell labeled '1' and 'Cellule de texte'. It contains an introduction to Python and a 'Hello world!' program. Below it is a green-bordered cell labeled '2' and 'Cellule de code', containing the code `print("Hello world!")`. To the left of this cell is a play button icon labeled '4'. Below the code cell is a blue-bordered cell labeled '3' and 'Résultat', which displays the output 'Hello world!'.

L'exécution des cellules de code se fait séquentiellement, ce qui permet de suivre le déroulement d'un programme ou d'une analyse. Pour exécuter le code d'une cellule de code, il suffit d'appuyer sur le bouton *Play* à gauche (entouré en 4). Les résultats de l'exécution du code s'affichent directement sous la cellule correspondante (voir 3).

Pour pouvoir travailler avec Google Collab, **il faut cependant disposer d'un compte Google**. Toute adresse email permettant de se connecter à Youtube, Gmail ou un autre service Google comme les adresses @gmail.com fonctionnent.

Sauvegarder son travail

La sauvegarde de toute modification est régulièrement effectuée automatiquement dans Google Collab. Il n'est pas donc pas nécessaire d'effectuer une action. Il est néanmoins possible de manuellement enregistrer le travail en allant dans **Fichier > Enregistrer** ou en utilisant les touches **CTRL+S**.

Exporter le code et sauvegarde du notebook

Il pourrait être utile d'exporter le code pour l'utiliser dans Thonny par exemple. Il y a deux formats d'exportation disponibles :

- *Notebook Jupyter* : c'est le format notebook identique à Google Collab qui peut être utilisé dans tout client Jupyter Notebook sur votre ordinateur sans internet. L'extension du fichier est *.ipynb et peut être téléchargé sous **Fichier > Télécharger > Télécharger le fichier .ipynb**
- *Code Python* : c'est un simple document texte contenant uniquement le contenu des cellules de code tandis que les cellules de texte sont mis en commentaires. L'extension du fichier est *.py et peut être téléchargé sous **Fichier > Télécharger > Télécharger le code .py**

Pour garder votre propre copie du notebook Google Collab dans votre propre espace Google Drive, il convient d'aller dans **Fichier > Enregistrer une copie dans le Drive**. Vous pourrez alors ensuite le modifier et le partager à vos élèves à votre guise comme vous le feriez pour un document Google Docs ou Sheet.

Mise en place en classe

Il convient de prévoir un ordinateur par étudiant et disposant d'une connexion internet fiable. Ensuite, les élèves sont invités à ouvrir le navigateur web pour accéder au notebook de la première phase à partir du lien suivant :

https://colab.research.google.com/drive/1F_sij13imZNYCzcnPuMY5_sSVe3IYNRH?usp=sharing

En haut à droite, ils doivent se connecter sur un compte Google en appuyant sur *Connexion* :



Si vous ne voulez pas perdre de temps sur la connexion sur le compte Google, il est préférable de créer des comptes Google à l'avance et de s'y connecter sur les ordinateurs avant l'activité.

Tout est prêt ! Vous pouvez commencer l'activité !

COMMENTAIRES & REPONSE AUX EXERCICES

La fiche élève est en principe suffisamment explicite pour nécessiter le moins d'éclaircissements pour tout débutant en programmation. Il arrive souvent que des erreurs se produisent sans que l'on comprenne son origine. Le plus souvent, ce sont soit

- *Des erreurs d'écriture* : une variable, un nom de fonction ou autre a été écrit avec une lettre manquante ou en trop ou intervertie.
- *Des erreurs de syntaxe Python* : par exemple, les variables ne peuvent pas commencer avec des chiffres en premier ; l'indentation du code n'est pas respectée ; il y a des arguments en trop ou manquants dans l'appel d'une fonction, etc.

Dans la plupart des cas, le message d'erreur contient beaucoup d'indices pour pouvoir retrouver l'origine de l'erreur. Il est donc conseillé de prendre le temps de déchiffrer les messages d'erreurs avant de tenter toute correction du code.

Section 2 : Les variables

La partie la plus importante pour la suite concerne la notion d'objet avec leurs propriétés et méthode. En effet, les listes et tuples sont des objets qui sont très fréquemment utilisés pour la suite du projet. De plus, d'autres objets seront manipulés par la suite et la compréhension de ces concepts facilite grandement leur usage.

Exercices de la section 2.1.

1) **Change les indices de l'affichage de la liste et vois ce qu'il se produit. Que se passe-t-il lorsque l'indice est grand?**

Le principal constat est que l'on ne peut faire référence qu'à des éléments existants dans une liste. C'est pourquoi, aucun problème ne survient lorsque l'indice est inférieur au nombre total d'éléments dans la liste tandis qu'un message d'erreur s'affichera lorsqu'on tente d'accéder à un élément inexistant.

```
# Provoque une erreur car l'indice dépasse le nombre d'éléments de la liste
maListe[0]
monAutreListe[4]
```

2) **Crée toutes les variables nécessaires et affiche le début du poème suivant de Victor Hugo :**

Il y a plusieurs manières de mener à bien cet objectif, le plus simple étant d'afficher directement les chaînes de caractères. Ici, on propose une approche où des variables sont créées pour chaque ligne mais cette approche produit un code plus long. Dans un autre contexte, cette approche est plus intéressante car la variable peut être modifiée en amont du programme plusieurs fois avant son affichage alors que la première approche aurait nécessité la modification manuelle du code pour réaliser cela.

```
ligne1 = "Demain, dès l'aube, à l'heure où blanchit la campagne,"
ligne2 = "Je partirai. Vois-tu, je sais que tu m'attends."
ligne3 = "J'irai par la forêt, j'irai par la montagne."
```

```
ligne4 = "Je ne puis demeurer loin de toi plus longtemps."  
print(ligne1)  
print(ligne2)  
print(ligne3)  
print(ligne4)
```

Section 4 : les fonctions

Les fonctions permettent la création des blocs élémentaires de tout programme complexe. Dans la philosophie de la programmation fonctionnelle, une fonction devrait être aussi courte que possible et ne réaliser qu'une seule tâche simple mais bien faite. Cela permet un code plus lisible et facile à réparer en cas de bug.

La notion de fonction est introduite avec un exemple mathématique par analogie aux fonctions mathématiques. Cela permet la mise en lien avec ce concept plus familier pour les élèves. Si cette analogie mathématique ne convient pas, il est possible de le remplacer par une fonction qui salue un individu dont le nom est indiqué en argument comme suit

```
def saluer(nom):  
    print("Bonjour ", nom, " !")
```

Erreur fréquentes

En recopiant, certains élèves retirent les commentaires de la fonction comme ceci :

```
def PerimetreCercle(Rayon):  
  
    Perimetre = 2 * 3.14 * Rayon  
  
    return Perimetre
```

A ce stade, certains oublient aussi de respecter l'indentation et oublient que ces instructions font partie de la fonction :

```
def PerimetreCercle(Rayon):  
  
    Perimetre = 2 * 3.14 * Rayon  
  
return Perimetre
```

Idéalement, il vaudrait mieux éviter de laisser des lignes vides entre la définition de la fonction et ses intructions. Donc ceci est plus correct et lisible :

```
def PerimetreCercle(Rayon):  
    Perimetre = 2 * 3.14 * Rayon  
  
    return Perimetre
```

Exercices de la section 4.1.

1) **Crée une fonction qui affiche un triangle faite d'astérisques comme ceci:**

```
def monSapin():
    print("  *")
    print(" ***")
    print(" *****")
    print(" *******")
    print("*****")
```

2) Crée une fonction qui calcule le volume d'une sphère et le renvoie.

```
def volumeSphere(R):
    pi = 3.14159265359
    return 4*pi*R*R*R/3
```

Exercices de la section 5.4.

1) **Crée une fonction qui affiche un triangle faite d'astérisques et correspondant à la hauteur indiqué en argument de la fonction:**

En utilisant des possibilités inabordées dans cette introduction, il est possible de produire un code plus court mais celui-ci a le mérite de n'utiliser que les concepts abordés à une exception. Pour mener à bien cet exercice, un argument de `print()` doit être abordé : de base, la fonction réalise un retour à la ligne suivante automatiquement mais en indiquant `end=''` en second argument, le retour à la ligne est remplacé par la chaîne vide `''` et permet ainsi l'affichage des caractères les uns à la suite des autres.

```
def monSapin(hauteur):
    i = 0
    while i < hauteur:
        # On affiche d'abord les espaces
        j = 1
        while j < hauteur - i:
            # L'argument end='' permet de ne pas faire de retour à la ligne suivante
            print(" ", end='')
            j = j + 1

        # Ensuite les étoiles
        j = 2 * i + 1
        while j > 0:
```

```
print("*", end='')
j = j - 1

# On réalise un retour à la ligne suivante
print(" ")
i = i + 1

monSapin(9)
```

2) Crée une fonction qui calcule le solde de ton compte épargne pour un nombre de mois et pour un taux d'intérêt indiqué en argument de la fonction.

```
def epargne(solde, mois, taux):
    # Initialisation du compteur
    i = 0
    total = solde

    # Calcul
    while i < mois:
        total = total * (1 + taux/100)
        i = i + 1

    # Affichage du solde final
    print(total)

epargne(100, 12, 1)
```


LIENS

Projet Mission Space Lab

La plupart de ces liens sont uniquement disponible en anglais.

Page de présentation du projet

<https://astro-pi.org/mission-space-lab/>

Règlementation relative au projet

<https://astro-pi.org/mission-space-lab/rulebook>

Guide du créateur

<https://projects.raspberrypi.org/en/projects/mission-space-lab-creator-guide/0>

Guide de l'estimation de la vitesse de l'ISS

<https://projects.raspberrypi.org/en/projects/astropi-iss-speed>

Guide du mentor

<https://astro-pi.org/mission-space-lab/>

Ressources Python

Voir liens proposés à la fin de la fiche élève.

IDE Thonny

<https://thonny.org/>