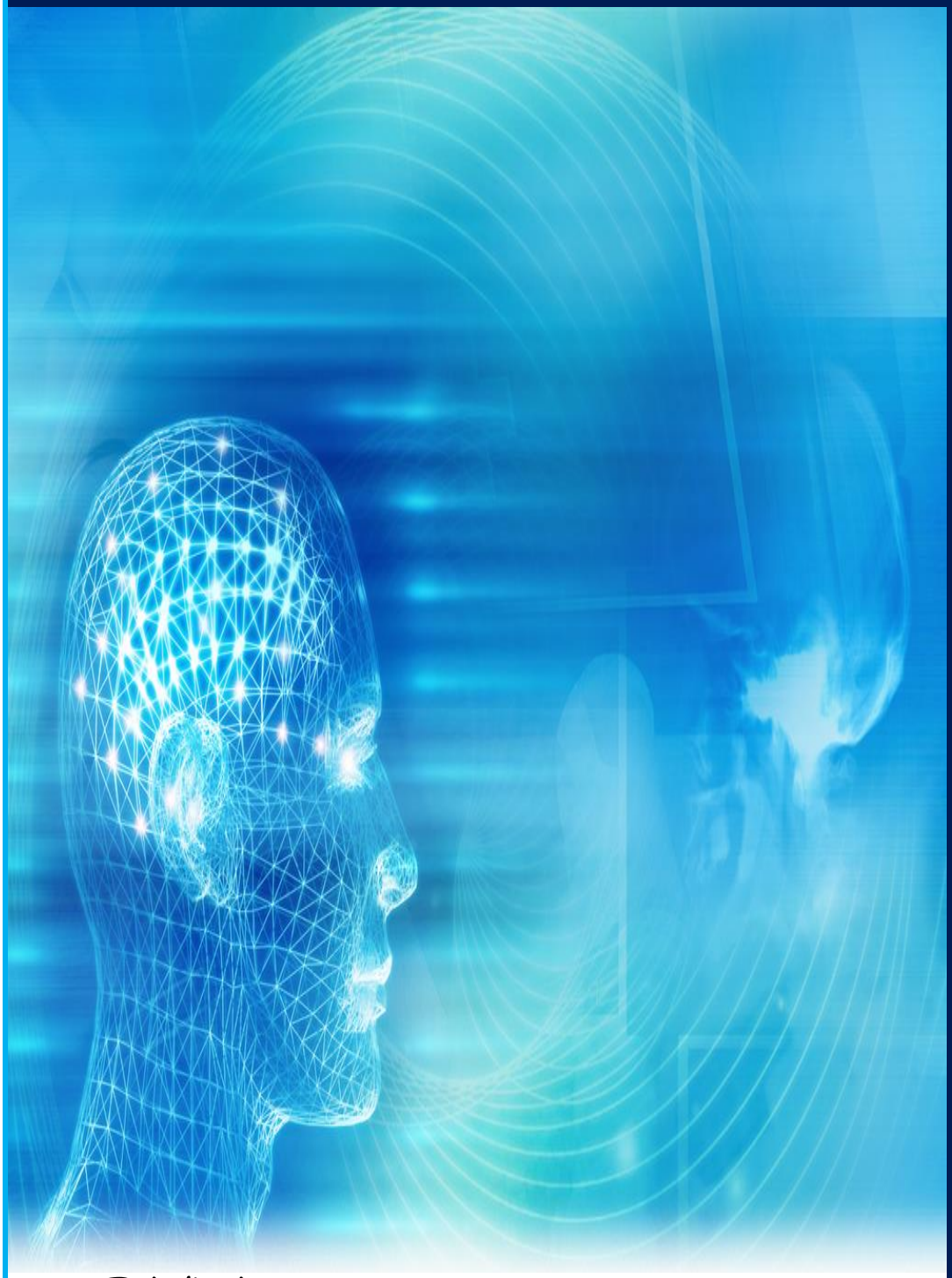
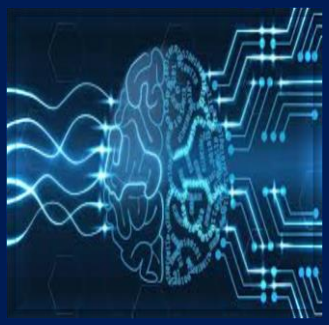
INTELLIGENCE ARTIFICIELLE





Réalisé par :

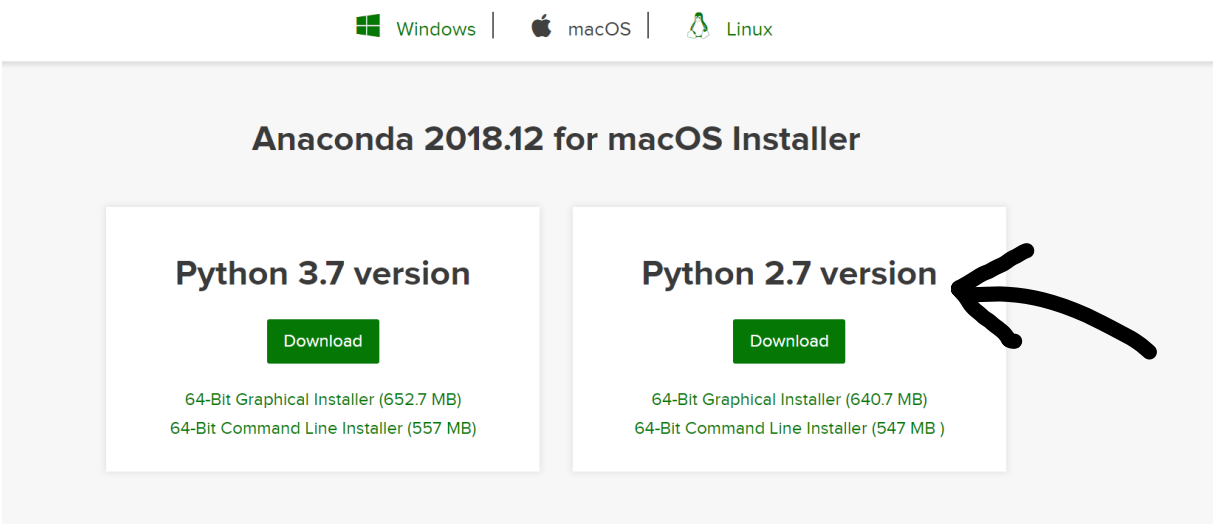
- AMINE LOUGUIT
- ZOUGAGH ABDELHAKIM

I. Installation Anaconda

Anaconda est une alternative puissante à virtualenv - un gestionnaire de paquets multi-plateforme, semblable à un pip , doté de fonctionnalités permettant de créer et de supprimer rapidement des environnements virtuels.

➢ Installation sous Windows

1. **Téléchargez le programme d'installation d'Anaconda :**
   https://www.anaconda.com/distribution/



La version récente de python est 3.x, mais nous nous contentons de python 2.7 pour des raisons de compatibilités.

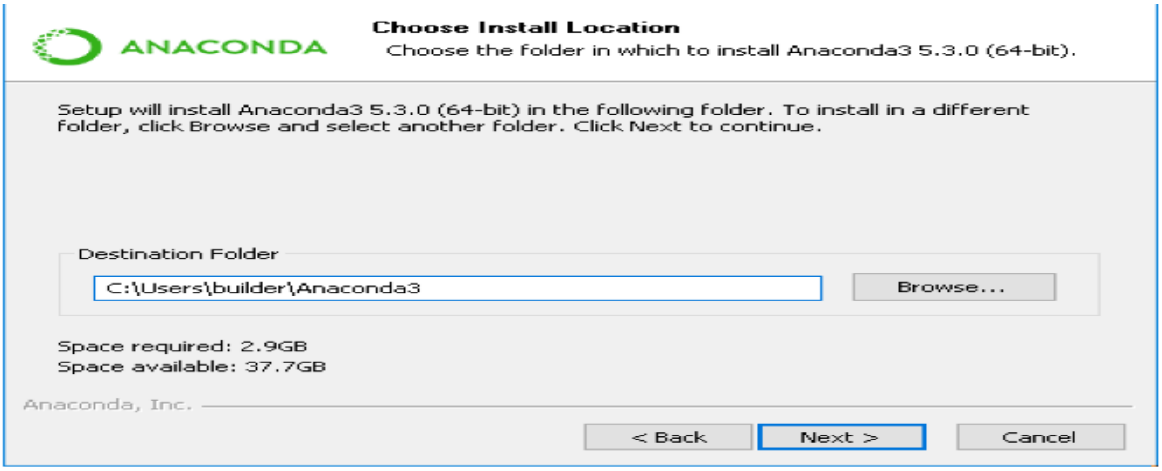
2. **Double-cliquez sur le programme d'installation pour lancer.**

REMARQUE 1: pour éviter les erreurs d'autorisation, ne lancez pas le programme d'installation à partir du dossier Favoris.
REMARQUE 2: Si vous rencontrez des problèmes lors de l'installation, désactivez temporairement votre logiciel antivirus lors de l'installation, puis réactivez-le à la fin de l'installation. Si vous avez installé pour tous les utilisateurs, désinstallez Anaconda, réinstallez-le pour votre utilisateur uniquement et réessayez.

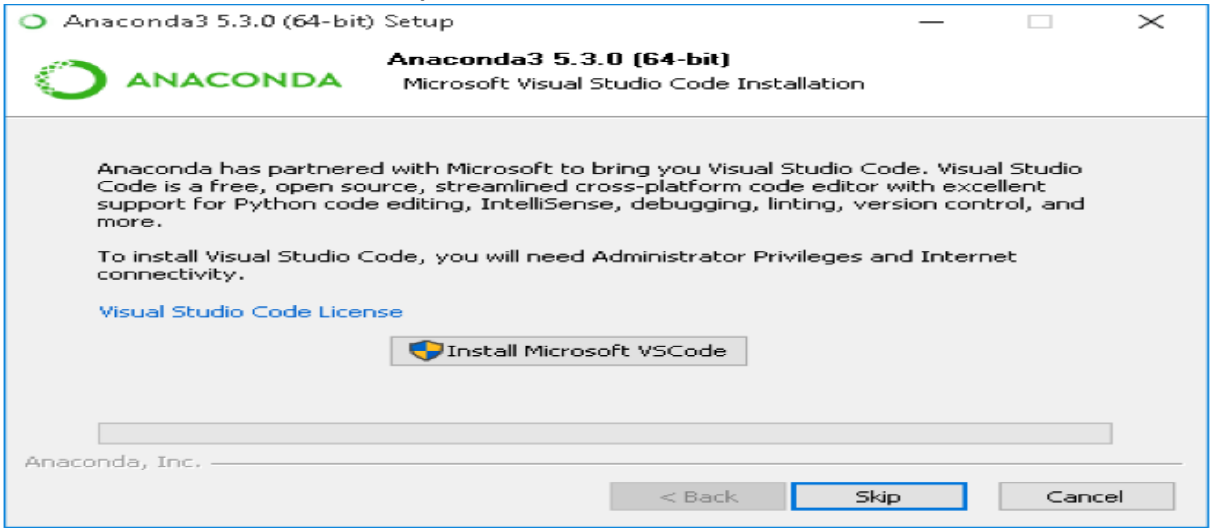
3. **Cliquez sur Suivant.**
4. **Lisez les conditions de licence et cliquez sur «J'accepte».**
5. **Sélectionnez une installation pour «Just Me» sauf si vous effectuez une installation pour tous les utilisateurs (ce qui nécessite des privilèges d'administrateur Windows), puis cliquez sur Suivant**.
6. **Sélectionnez un dossier de destination pour installer Anaconda et cliquez sur le bouton Suivant.**

REMARQUE 3: Installez Anaconda dans un chemin de répertoire ne contenant pas d'espaces ni de caractères unicode.

REMARQUE 4: n'installez pas en tant qu'administrateur sauf si des privilèges d'administrateur sont requis.



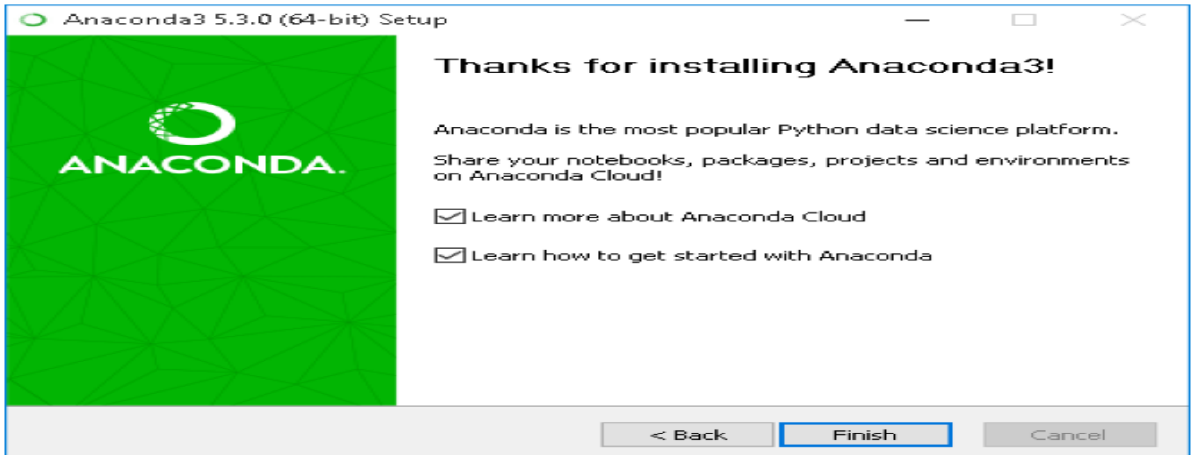
7. **Choisissez si vous souhaitez ajouter Anaconda à votre variable d'environnement PATH. Nous vous recommandons de ne pas ajouter Anaconda à la variable d'environnement PATH, car cela pourrait interférer avec d'autres logiciels. Utilisez plutôt le logiciel Anaconda en ouvrant Anaconda Navigator ou l'invite Anaconda à partir du menu Démarrer.**
8. **Choisissez si vous souhaitez enregistrer Anaconda en tant que votre Python par défaut. Sauf si vous envisagez d'installer et d'exécuter plusieurs versions d'Anaconda, ou plusieurs versions de Python, acceptez la valeur par défaut et laissez cette case cochée.**
9. **Cliquez sur le bouton Installer. Si vous souhaitez voir les packages installés par Anaconda, cliquez sur Afficher les détails.**
10. **Cliquez sur le bouton Suivant.**
11. **Facultatif: pour installer VS Code, cliquez sur le bouton Installer Microsoft VS Code. Une fois l'installation terminée, cliquez sur le bouton Suivant.**



Ou pour installer Anaconda sans VSCode, cliquez sur le bouton Ignorer.

REMARQUE 5: L'installation de VS Code avec le programme d'installation d'Anaconda nécessite une connexion Internet. Les utilisateurs hors ligne peuvent éventuellement trouver un programme d'installation de VS Code hors ligne de Microsoft.

12. **Après une installation réussie, la boîte de dialogue <<Merci >> d'avoir installé Anaconda»:**



13. **Une fois votre installation terminée, vérifiez-la en ouvrant Anaconda Navigator, un programme fourni avec Anaconda: dans le menu Démarrer de Windows, sélectionnez le raccourci Anaconda Navigator. Si Navigator s'ouvre, vous avez correctement installé Anaconda.**

➢ Installation sous Linux :

1. **Dans votre navigateur, téléchargez le programme d'installation Anaconda pour Linux.**
2. **Entrez ce qui suit pour installer Anaconda for Python 2.7:**

```
bash ~/Downloads/Anaconda2-5.3.0-Linux-x86_64.sh
```

REMARQUE: Incluez la commande bash que vous utilisiez ou non le shell Bash.
REMARQUE: Si vous n'avez pas téléchargé dans votre répertoire de téléchargements, remplacez ~ / Downloads / par le chemin du fichier que vous avez téléchargé.
REMARQUE: Choisissez «Installer Anaconda en tant qu'utilisateur» sauf si des privilèges root sont requis.

3. **Le programme d'installation vous invite à «Pour poursuivre le processus d'installation, veuillez consulter le contrat de licence.» Cliquez sur Entrée pour afficher les termes de la licence.**

4. **Faites défiler jusqu'au bas des termes de la licence et entrez «Oui» pour accepter.**

5. **Le programme d'installation vous invite à cliquer sur Entrée pour accepter l'emplacement d'installation par défaut, sur CTRL-C pour annuler l'installation ou spécifier un autre répertoire d'installation. Si vous acceptez l'emplacement d'installation par défaut, le programme d'installation affiche «PREFIX = / home / <utilisateur> / anaconda <2 ou 3>» et poursuit l'installation. Cela peut prendre quelques minutes.**

6. **Le programme d'installation vous invite à «Voulez-vous que le programme d'installation ajoute l'emplacement d'installation d'Anaconda <2 ou 3> à PATH dans votre /home/<user>/.bashrc?» Entrez Oui.**

REMARQUE: Si vous entrez «Non», vous devez ajouter manuellement le chemin d'accès à Anaconda, sans quoi conda ne fonctionnera pas.

7. **Le programme d'installation décrit Microsoft VS Code et vous demande si vous souhaitez installer VS Code. Entrez oui ou non. Si vous avez sélectionné oui, suivez les instructions à l'écran pour terminer l'installation de VS Code.**

8. **L'installateur termine et affiche < Merci > d'avoir installé Anaconda <2 ou 3>!»**

9. **Fermez et ouvrez la fenêtre de votre terminal pour que l'installation prenne effet, ou vous pouvez entrer la commande :**

```
source ~/.bashrc.
```

```
louguit@louguit-PORTEGE-Z30-B:~$ conda -V
conda 4.5.12
louguit@louguit-PORTEGE-Z30-B:~$ conda update conda
Solving environment: |
```

II. Créer un environnement

Après avoir installé Anaconda, voici quelques commandes pour commencer:

Créer un environnement :

```
conda create --name <envname> python=<version>
```

où `<envname>` dans un nom arbitraire pour votre environnement virtuel, et `<version>` est une version spécifique de Python que vous souhaitez configurer.

Activer et désactiver votre environnement :

```
# Linux, Mac
source activate <envname>
source deactivate
```

ou,

```
# Windows
activate <envname>
deactivate
```

Afficher une liste des environnements créés :

```
conda env list
```

Supprimer un environnement :

```
conda env remove -n <envname>
```

Install additional Python packages to a virtual environment :

```
conda install -n yourenvname [package]
```

Trouvez plus de commandes et de fonctionnalités dans la documentation officielle du conda .

Quelques Screenshot

Sous Windows :

```
(base) C:\Users\Amine>conda create -n AI python=2.7 anaconda
Solving environment: |
```

```
(base) C:\Users\Amine>conda activate AI

(AI) C:\Users\Amine>
```

```
(AI) C:\Users\Amine>python
Python 2.7.15 |Anaconda, Inc.| (default, Dec 10 2018, 21:57:18) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Sous Linux :

```
(base) louguit@louguit-PORTEGE-Z30-B:/$ conda create -n AI python=2.7 anaconda
Collecting package metadata: done
Solving environment: done

## Package Plan ##

  environment location: /home/louguit/anaconda2/envs/AI

  added / updated specs:
    - anaconda
    - python=2.7
```

```
  xlwt               pkgs/main/linux-64::xlwt-1.3.0-py27h3d85d97_0
  xz                 pkgs/main/linux-64::xz-5.2.4-h14c3975_4
  yaml               pkgs/main/linux-64::yaml-0.1.7-had09818_2
  zeromq             pkgs/main/linux-64::zeromq-4.2.5-hf484d3e_1
  zict               pkgs/main/linux-64::zict-0.1.3-py27_0
  zlib               pkgs/main/linux-64::zlib-1.2.11-h7b6447c_3
  zstd               pkgs/main/linux-64::zstd-1.3.7-h0b5b093_0


Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate AI
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) louguit@louguit-PORTEGE-Z30-B:/$ conda activate AI
(AI) louguit@louguit-PORTEGE-Z30-B:/$ python
Python 2.7.15 |Anaconda, Inc.| (default, Dec 14 2018, 19:04:19)
[GCC 7.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

III. Initialisation avec PYTHON :

Les commandes basics sous LINUX :

- On a attaqué tout d'abord les commandes basics qu'on peut utiliser sous LINUX.
  A savoir : - **mkdir** : créer un répertoire.
  - **cd** : se déplacer.
  - **ls** : lister le contenu.
  - **touch** : créer un fichier.
  - **cp** : copier des fichiers.
  - **rm** : supprimer des fichiers.
  - **pwd** : afficher le nom du dossier courant.

  On a cité juste les commandes plus utilisés vu qu'on a d'autres commandes sur LINUX.

Utilisation de l'interpréteur PYTHON :

- On a commencé par l'utilisation de l'interpréteur simple sous PYTHON.
- On a vu **les opérateurs**. (+, -, /, *, %)
- On a traité **les booléennes**. (True or False)
- On a manipulé la fonction **dir()**, son rôle est de lister l'ensemble des méthodes offertes par une classe spécifique.

  Exemple :

Figure 1:Exemple de l'utilisation de la fonction dir()

Les structures de données :

- On a manipulé **les listes**, parmi les caractères des listes on trouve la notion mutable et ses éléments peuvent être hétérogènes.

Exemple :

A= [1,'GSEII', 23,'WORLD']

- On peut créer une liste de liste.

Exemple :

A= [[‘a’ ,’b’, ’c’],[[1, 2, 3]],[[‘d’,3,’e’]]

Exercise :

- Créer une liste A qui contient les lettres de l’alphabet. Créer ensuite une liste B qui contient les caractères numériques. Calculer une liste Fusion qui combine chaque caractère de A et un chiffre de B.

Correction :

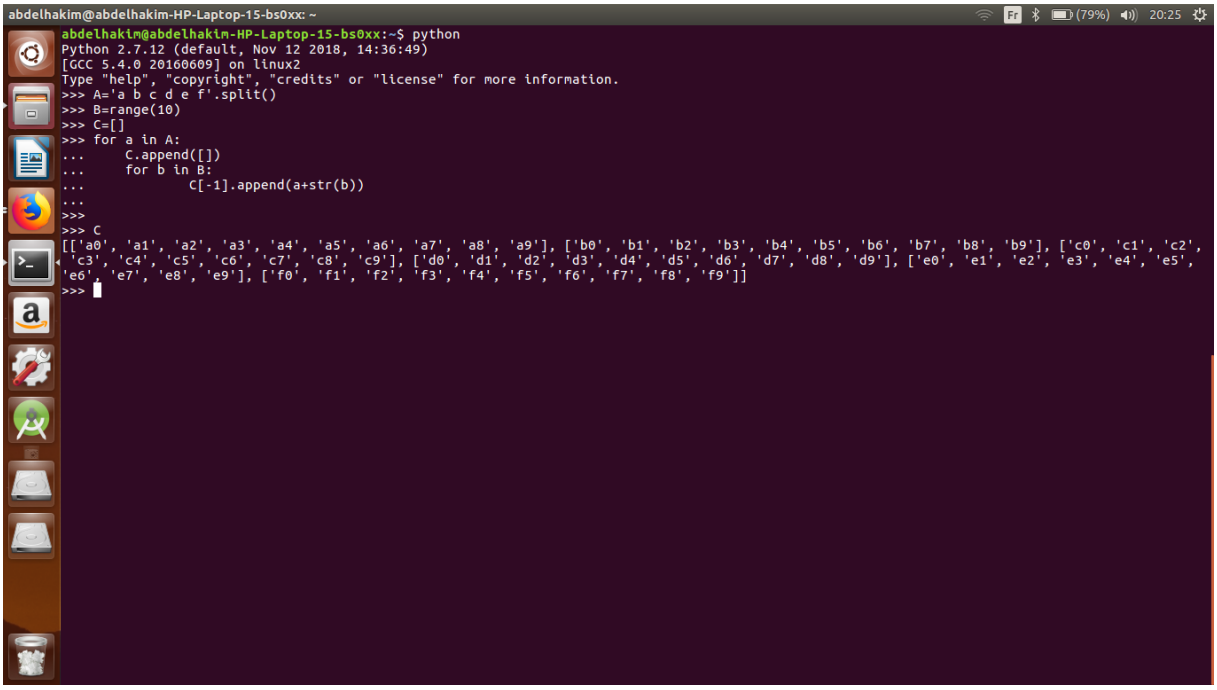


Figure 2: Correction d'exercice

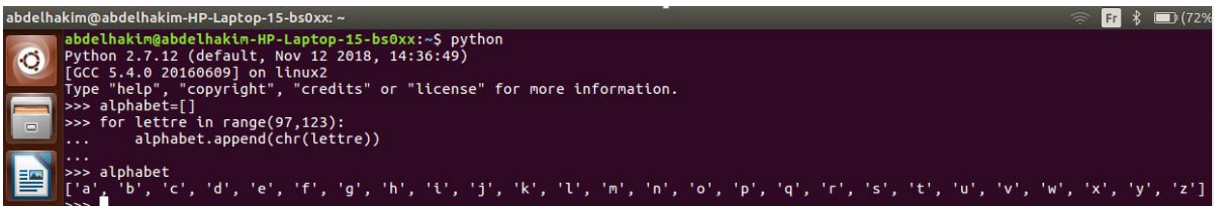
- On a travaillé cet exercice seulement avec l’interpréteur PYTHON.

Recherche :

- Chercher la méthode pour obtenir la liste des alphabets.

Réponses :

- La méthode qui nous permet de lister les alphabets est :



- Les numéros 97 et 123 signifient l'équivalent de la lettre A et Z en code ASCII.
- Après on a vu **les tuples**, parmi les caractères des listes on trouve la notion immutable et aussi on ne peut pas modifier ses éléments.
  Exemple :

Figure 3: Lister les alphabets

A= (4, 3, 7, 8, 9, 10)

- On a passé après directement à manipuler **les sets**, se sont une structure de données pour représenter une liste non ordonnée sans des répétitions.

- On a vu aussi **les dictionnaires**, se sont une structure de données qui permet un accès rapide à une collection indexé par un ensemble de **clés**. La clé doit être un objet *immutable* pour permettre une identification unique des valeurs.

- On a fini cette partie avec la notion de programmation orienté objet sous LINUX. On a cité cet exemple dans le cours qui permet de clarifier l'utilisation des fonctions.

```
class Rectangle:
  def __init__(self,width):
    """
    Constructeur par une longueur
    """
    self.width = width

  def get_width(self):
    """
    Renvoir la longueur du rectangle
    """
    return self.width

  def  get_surface(self):
    return self.width**2

  def __str__(self):
    return "Carre [width=%d]" %self.width
```

Figure 4: Classe rectangle