Séquence 3

Algorithme et programmation

SI

LYCÉE LÉONARD DE VINCI

Document Technique

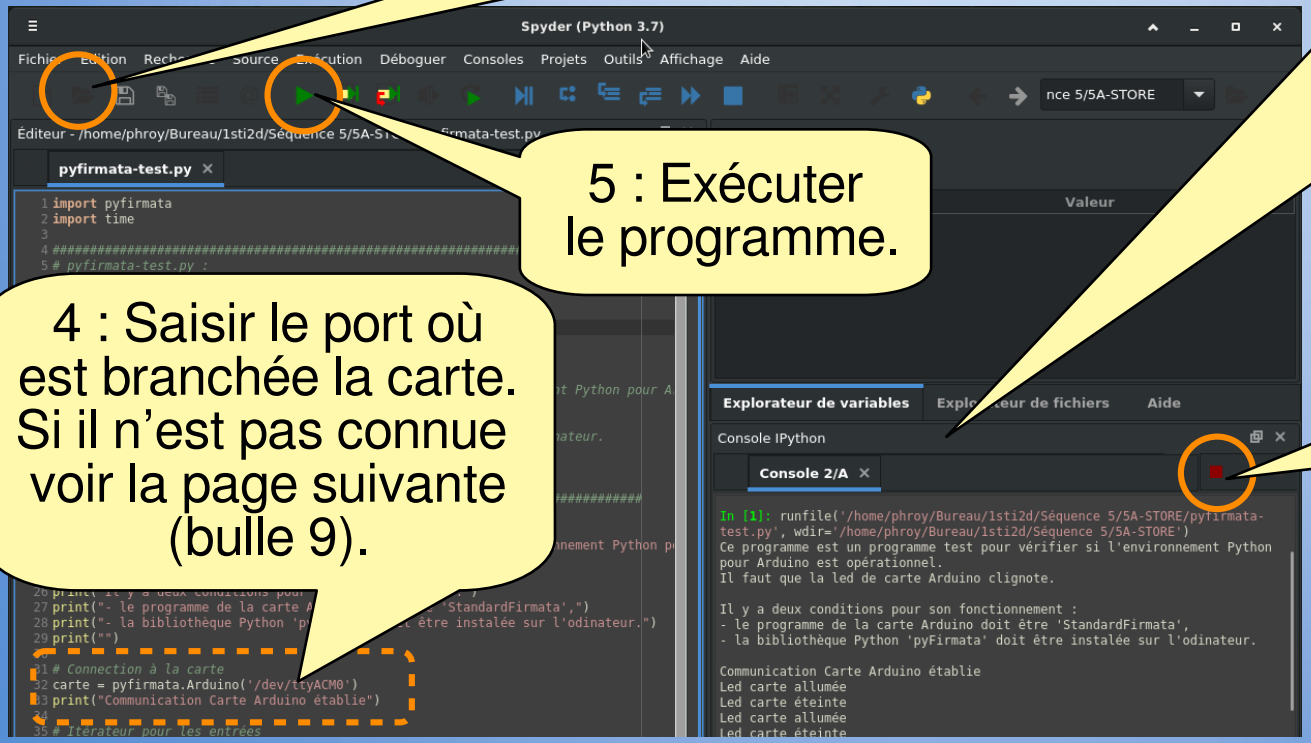
Python pour Arduino avec pyFirmata

1 – Tester si l'environnement pyFirmata + Arduino est fonctionnel

1: Brancher la carte Arduino sur l'ordinateur avec le cordon USB.

2 : Lancer le programme **Spyder**.

3 : Ouvrir le programme **pyfirmata-test.py** présent dans les ressources.



5 : Exécuter le programme.

6 : Vérifier la bonne exécution du programme :

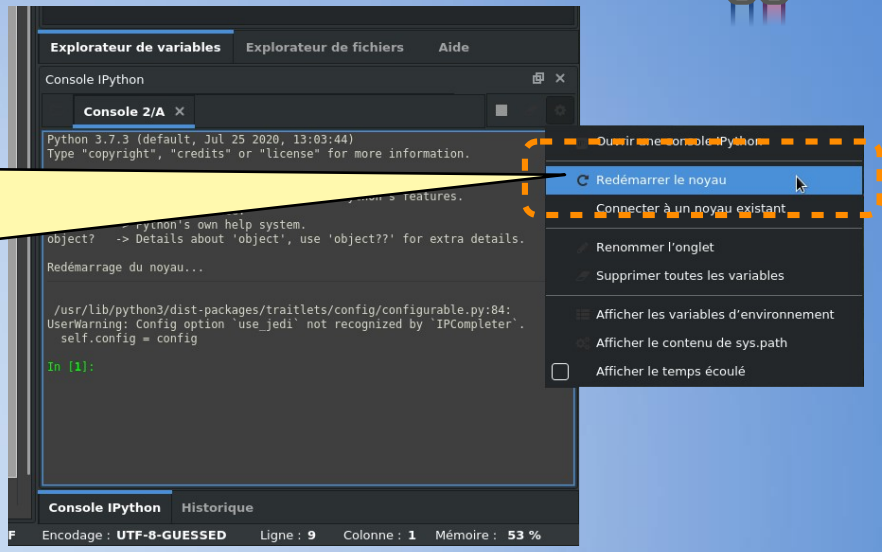
- la led de la carte clignote,
- les messages de la console sont affichés.

4 : Saisir le port où est branchée la carte. Si il n'est pas connue voir la page suivante (bulle 9).

7 : Stopper le programme.

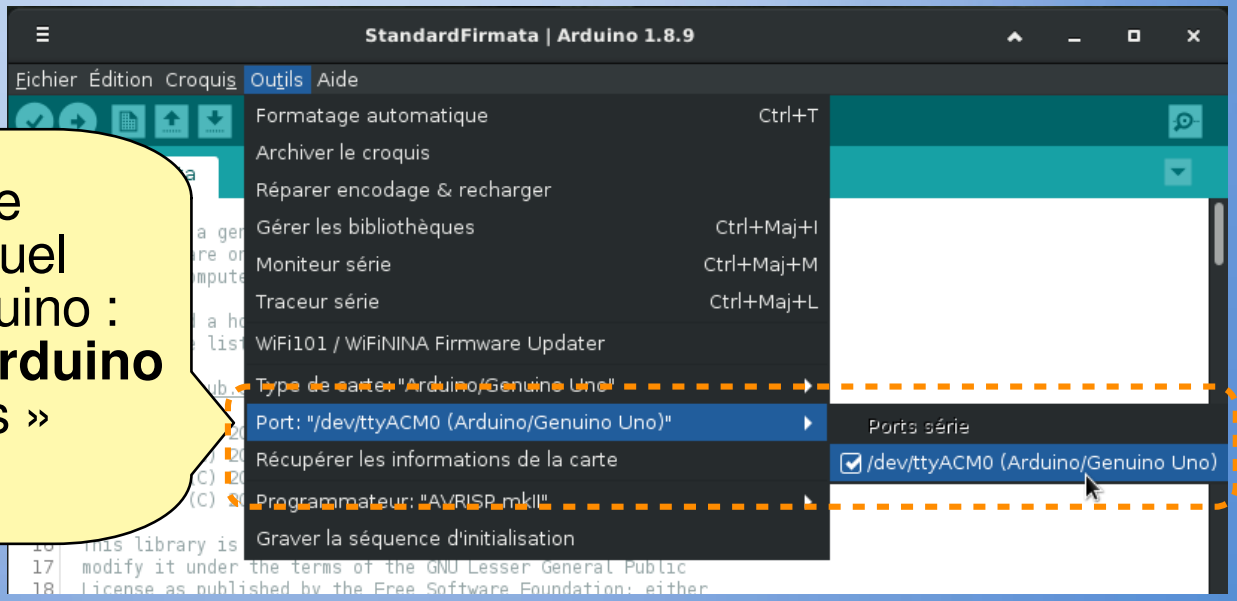
1 – Tester si l'environnement pyFirmata + Arduino est fonctionnel

8 : Dans **Spyder**, après l'arrêt du programme, il peut arriver qu'on ne puisse pas relancer le programme. Il faut alors **redémarrer le noyau**.



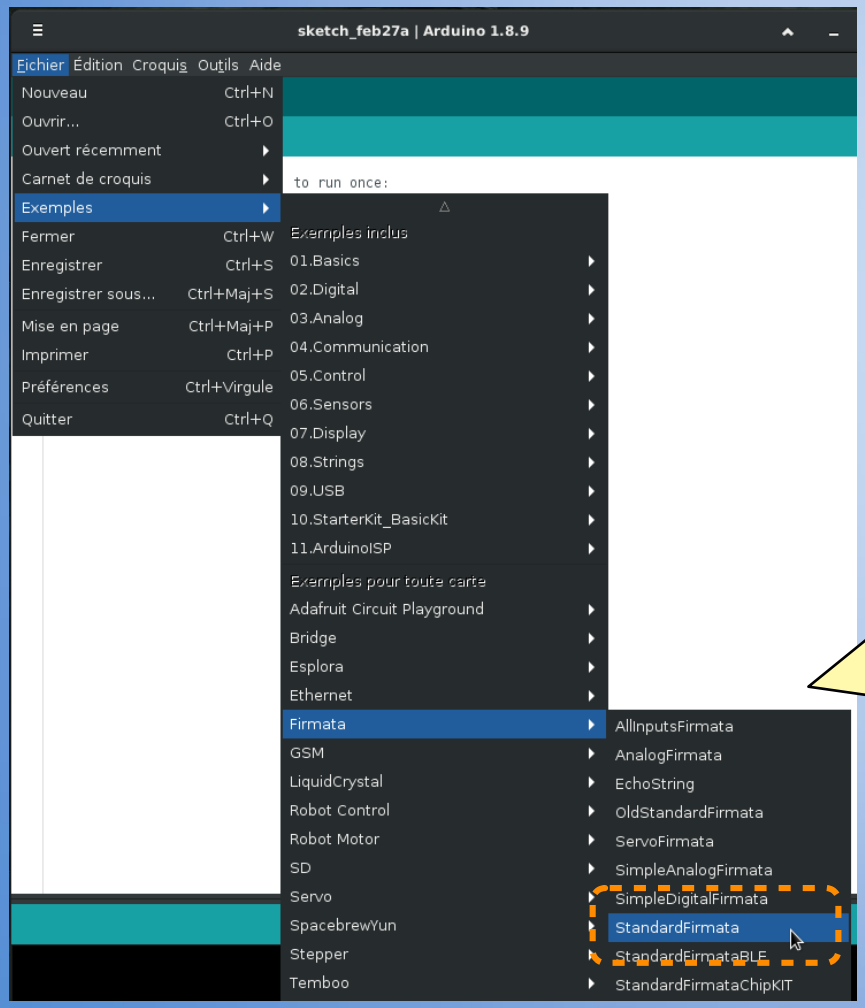
9 : Identifier le port de communication sur lequel est branché la carte Arduino :

- lancer le **programme Arduino**
- aller sur le menu « Outils »
- lire le **port** détecté



2 – Préparer la carte Arduino (charger le programme StandardFirmata)

Firmata est un **protocole de communication** utilisé pour une liaison entre un microcontrôleur et un ordinateur via le port série.



Le principe est de mettre le microcontrôleur (ici la carte **Arduino**) dans mode d'écoute. Il exécutera alors les instructions envoyées par l'ordinateur.

1 : Lancer le programme **Arduino**

2 : Charger le programme **StandardFirmata**

Le programme **StandardFirmata** fait partie des exemples de l'IDE d'**Arduino**.

2 – Préparer la carte Arduino (charger le programme StandardFirmata)

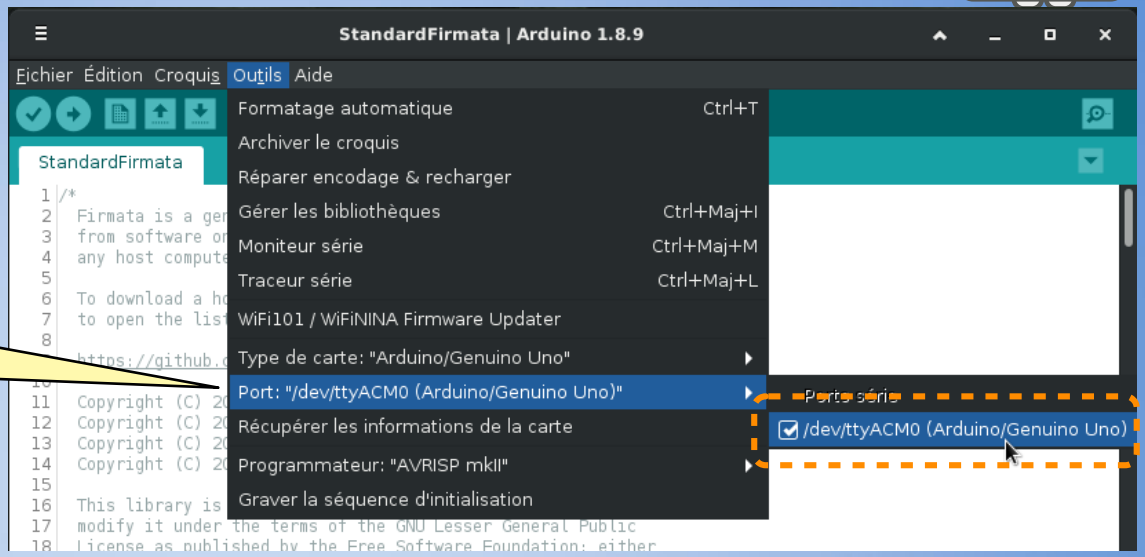
3 : Brancher la carte Arduino sur l'ordinateur avec le cordon USB

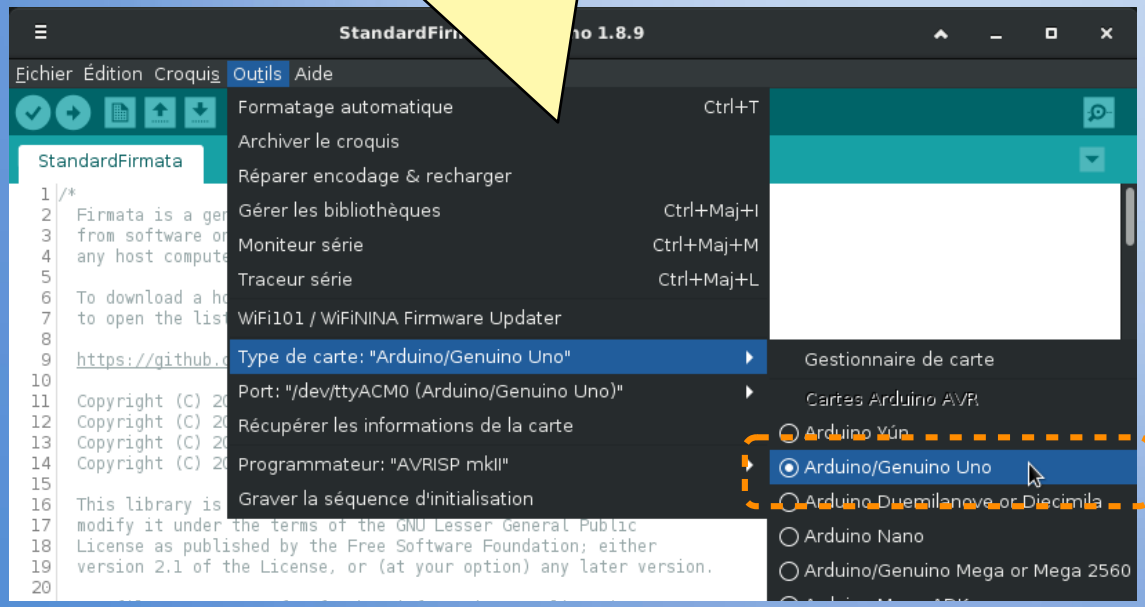
4 : Définir le port de communication

5 : Définir le type de carte **Arduino Uno**

6 : Téléverser le programme vers la carte

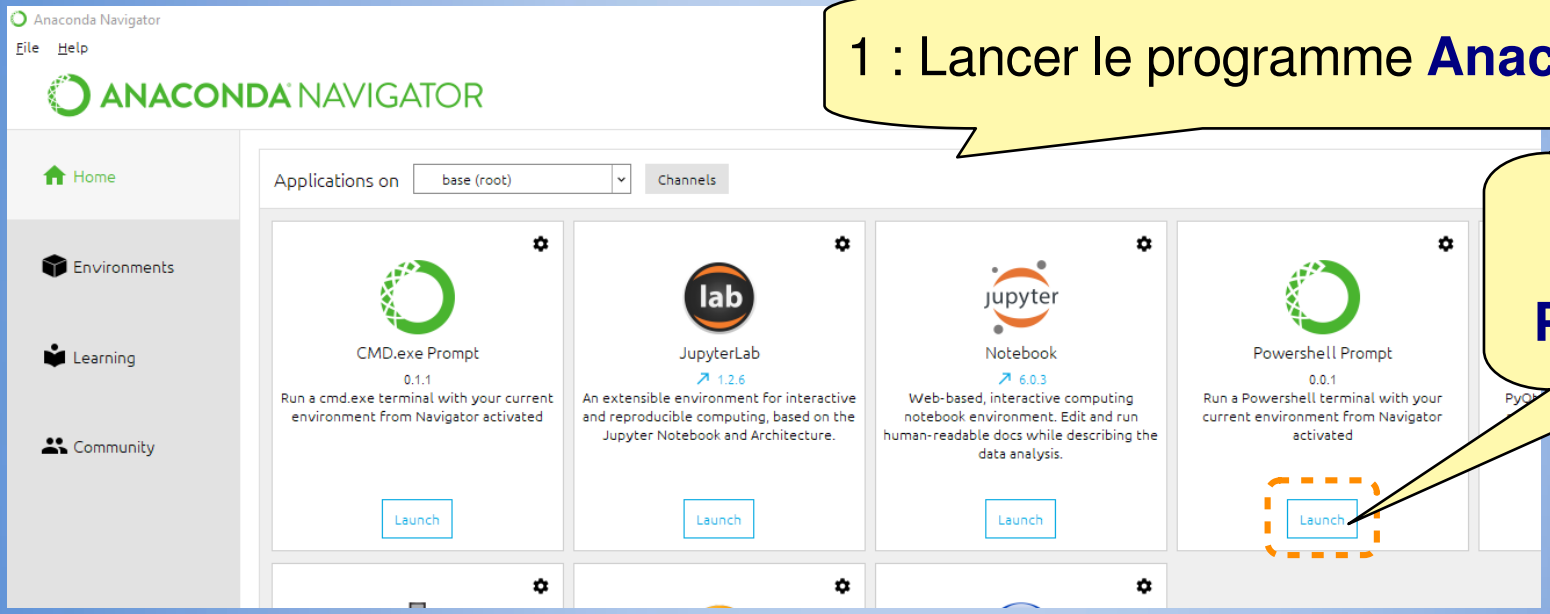
7 : Carte prête





3 – Installer la bibliothèque Python pyFirmata

La bibliothèque **pyFirmata** permet d'utiliser le protocole **Firmata** dans un programme Python. Il faut donc installer la bibliothèque **pyFirmata**.



1 : Lancer le programme **Anaconda**.

2 : Lancer un terminal **Powershell**.

3 : Dans le terminal exécuter la commande : **`pip install pyfirmata`**.

```
C:\windows\System32\WindowsPowerShell\v1.0\powershell.exe
ase) PS C:\Users\philippe.roy> pip install pyfirmata
ollecting pyfirmata
  Downloading pyFirmata-1.1.0-py2.py3-none-any.whl (14 kB)
Collecting pyserial
  Downloading pyserial-3.5-py2.py3-none-any.whl (90 kB)
     |                                | 90 kB 1.9 MB/s
Installing collected packages: pyserial, pyfirmata
Successfully installed pyfirmata-1.1.0 pyserial-3.5
(base) PS C:\Users\philippe.roy>
```

4 – Carte de référence pyFirmata

Entête du programme

Tout programme commence par l'import des bibliothèques, la création de l'objet carte et le démarrage de l'itérateur.

Import des Bibliothèques
La bibliothèque `time` n'est obligatoire mais recommandée car elle est très souvent utilisée.

Création de l'objet carte
C'est à cet endroit qu'il faut spécifier le port de communication (par exemple : 'COM3' '/dev/ttyACM0', ...).

Démarrage de l'itérateur
C'est nécessaire pour lire les entrées de la carte.

```
import pyfirmata
import time

# Connexion à la carte Arduino
carte = pyfirmata.Arduino('/dev/ttyACM0')
print("Communication Carte Arduino établie")

# Itérateur pour les entrées
it = pyfirmata.util.Iterator(carte)
it.start()
```

Fin du programme

Tout programme termine par la fermeture du port de communication.

Destruction de l'objet carte

```
# Fermer la connexion à la carte Arduino
carte.exit()
```

4 – Carte de référence pyFirmata

Définition des entrées-sorties

La définition des entrées-sorties de l'objet **carte** se fait avec la fonction :

```
carte.get_pin('type:broche:mode').
```

- **a** pour analogique
- **d** pour binaire (digital)

numéro de la broche (0 à 13)

- **i** pour input (entrée)
- **o** pour output (sortie)
- **p** pour pwm (sortie variable)

Le mode pwm ne s'applique uniquement aux sorties binaires 3, 5, 6, 9,10 et 11 (carte UNO).

```
bouton = carte.get_pin('d:10:i')          # bouton relié sur la broche 10

led = carte.get_pin('d:13:o')             # led reliée sur la broche 13

cpt_son = carte.get_pin('a:0:i')          # capteur sonore relié sur la
                                          # broche analogique A0
```

4 – Carte de référence pyFirmata

Lecture des entrées et écriture des sorties

La lecture des entrées se fait avec la fonction : **`entree.read()`**

En fonction de la tension au borne de la broche, la valeur retournée est
- pour une entrée binaire : un valeur binaire : 0 (0 V) ou 1 (5 V)
- pour une entrée analogique : une valeur décimale comprise entre 0 et 1

```
bouton = carte.get_pin('d:10:i')   # bouton sur la broche 10
cpt_son = carte.get_pin('a:0:i')   # capteur sonore sur la broche A0
if bouton.read()== True :          # test si le bouton est appuyé
  print(cpt_son.read())            # affiche la valeur du capteur sonore
```

L'écriture des sorties fait avec la fonction : **`sortie.write(valeur)`**

En fonction de la tension qu'on souhaite appliquer à la broche : la **`valeur`** est
- pour une sortie binaire : une valeur binaire : 0 (0 V) ou 1 (5 V)
- pour une sortie pwm : une valeur décimale comprise entre 0 et 1

```
led_r = carte.get_pin('d:13:o')    # led rouge sur la broche 13
led_v = carte.get_pin('d:11:p')    # led verte sur la broche 11 en PWM
led_r.write(1)                     # allumer la led rouge (5 V)
led_r.write(0)                     # éteindre la led rouge (0 V)
led_v.write(0.20)                  # allumer la led verte faiblement (1 V)
```