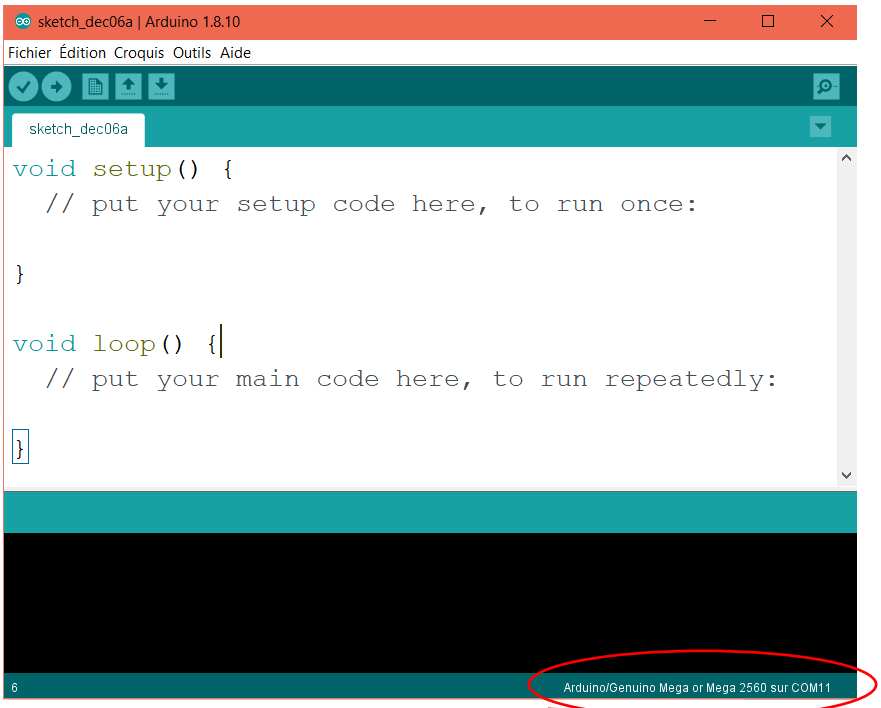
1. Programmation de l'allumage et de l'extinction d'une ampoule à l'aide du microcontrôleur Arduino et du langage Python

La carte Arduino est généralement programmée en langage C, avec l'Interface de programmation Arduino **(voir document pdf « PRESENTATION ARDUINO »)** :



La carte communique avec l'Interface via le PC par l'intermédiaire d'un port série USB. Pour l'exemple ci-dessus, la connexion entre la carte et l'interface de programmation est détecté et nommée comme un port de communication : COM11

Nous allons créer une interface de dialogue entre la carte Arduino et le logiciel EduPython :

Nous allons utiliser le protocole Firmata :

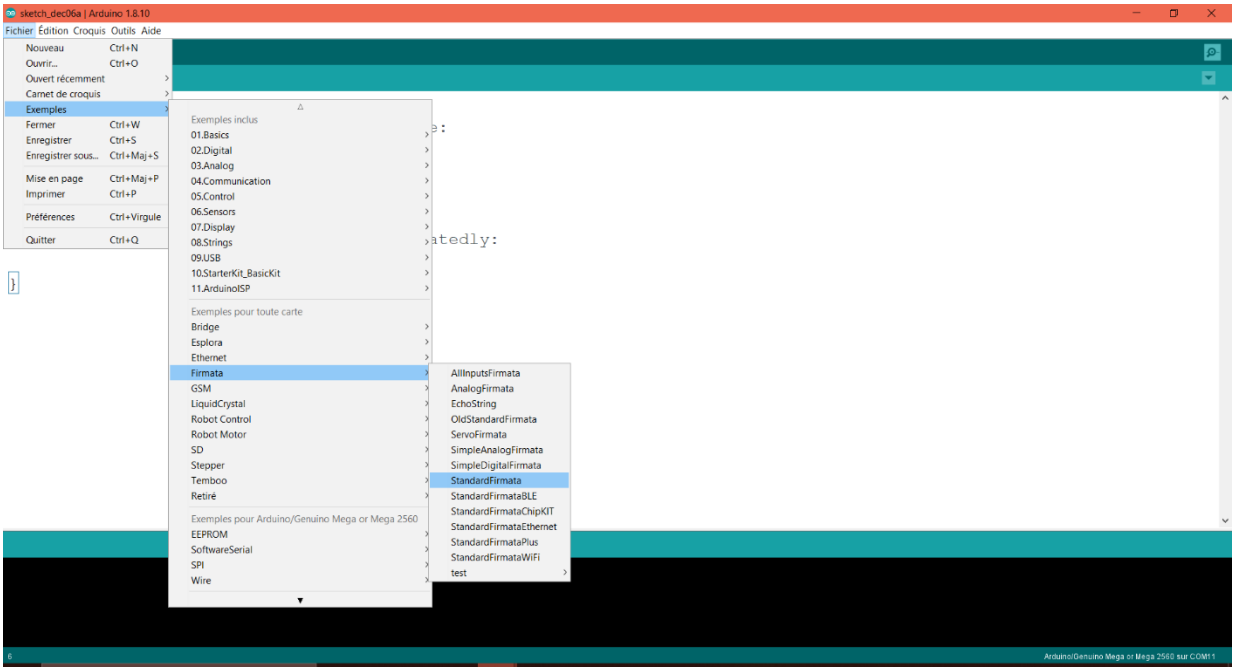
Étape 1: Installation du Firmware sur la carte :

Firmata est un protocole qui vise à faciliter la communication entre un ordinateur et un microcontrôleur. Les messages échangés par l'ordinateur et le microcontrôleur sont similaires à ceux du protocole MIDI, utilisé dans le domaine des instruments de musique électroniques.

- Installation du protocole Firmata sur la carte Arduino pour l'exécution d'un programme en langage Python ou autre. Ouvrir l'interface Arduino puis :

  fichier > Exemples > Firmata > StandardFirmata

  Voir exemple page suivante.



- Téléverser le programme dans la carte Arduino.
- La carte est maintenant prête à communiquer sous n'importe quel langage avec le PC. Le Firmware restera actif tant qu'un autre programme n'aura pas été téléversé, que la carte soit débranchée ou non.

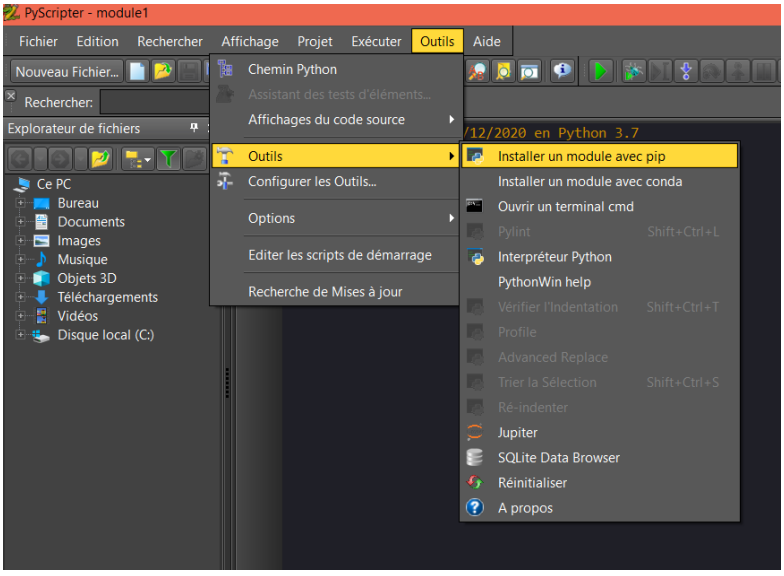
Étape 2: Installation et fonctionnement du module pyfirmata sous EduPython :

1. Installation :

Pour utiliser la carte avec EduPython, il faut installer la bibliothèque Pyfirmata, qui renferme les commandes Python compréhensibles par la carte Arduino.

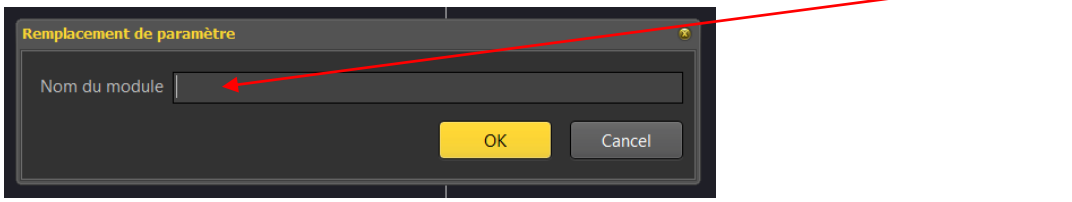
- Démarrer EduPython et faire :

  `Outils > installer un module avec pip`



Exemple sous EduPython 3.0

Une fenêtre apparait à l'écran, écrire le nom de la bibliothèque à installer : **pyfirmata**



2. Utilisation :

Dans EduPython, Nous pouvons taper un code qui sera reconnu par la carte, à condition de le faire commencer par les lignes suivantes :

Les commentaires précisent les fonctions de chaque ligne :

```
#Importation du module pyfirmata
from pyfirmata import Arduino, util
#importation du module time
import time

#Choisir un nom pour la carte («carte» par exemple) et son adresse («COM11»)
carte = Arduino('COM11')

# Ces deux lignes lancent la fonction mesure en temps réel
acquisition = util.Iterator(carte)
acquisition.start()
```

Les entrées et sorties sur la carte se définissent grâce à la fonction get_pin(). Avec pour un choix de paramètres dans la parenthèse :

Par exemple :

```
entree1 = carte.get_pin('d:2:i')
```

a	analogique
d	digitale
p	PWM

Numéro de pin choisi par l'utilisateur

i	input
o	output

Ici, le pin digitale numéro 2 est utilisé en sortie.

Remarque :

- Prévoir en fin d'initialisation (définition des entrées et lancement de la fonction mesure du temps réel) une pause de 1 seconde nécessaire à l'initialisation de la carte :

```
time.sleep(1.0)
```

- Prévoir en fin de programme la fonction carte.exit() à la fin du code, pour terminer l'acquisition des mesures proprement :

```
carte.exit()
```

3. Fonction read() et write() :

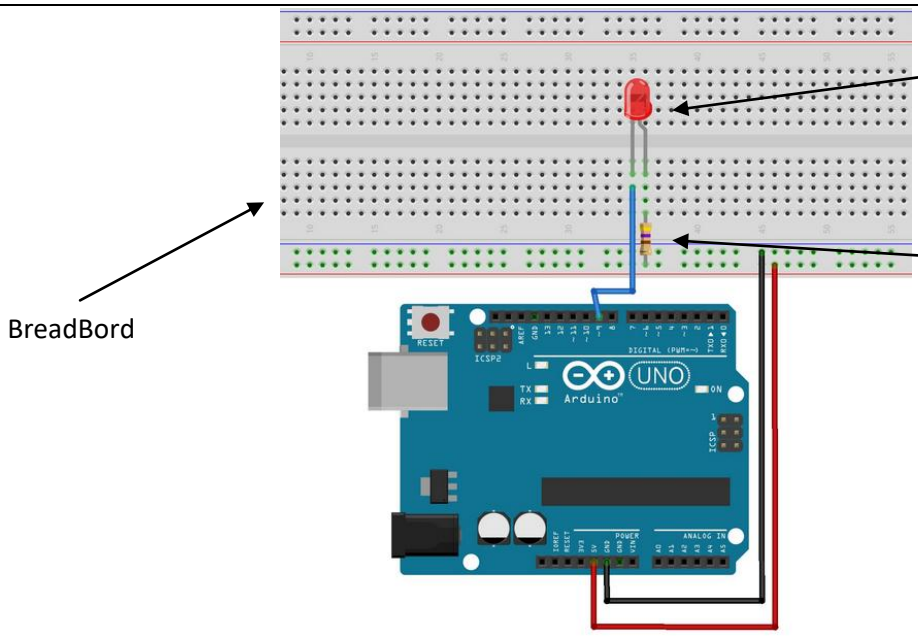
Les cas possibles :

Entrée		Sortie
Analogique	Digitale	Digitale
read()	read()	write()
Lis la valeur de tension sur l'entrée analogique. Cette fonction renvoie un nombre type Float compris entre 0 et 1 qui correspond à une tension comprise entre 0V et 5V	Lis l'état de la tension sur l'entrée analogique. (type Boolean) 0 pour 0V (False) 1 pour 5V (True)	Impose une tension sur la sortie: 0V pour write(0) 5V pour write(1). Il est possible d'utiliser write(False) ou write(True)

4. Programme Test :

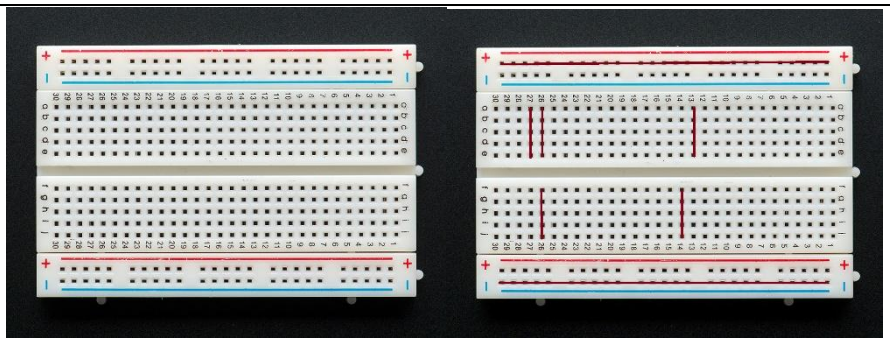
Si vois avez correctement réalisé les actions précédentes, nous allons pouvoir réaliser un test de connexion

En premier lieu, réaliser le montage suivant : (la Led est connectée à la pin digitale 9 de la carte Arduino par l'intermédiaire de la **BreadBord**)



Avant de commencer : La plaque "breadboard" :

Ensuite il faut prévoir une **plaque d'essai** ou **breadboard** qui est indispensable pour tester et interfacer ses projets. C'est une plaque en plastique avec des rangées de trous (par cinq ou plus) dans lesquels nous planterons nos composants.



Plaque d'essai ou Breadboard (à gauche en rouge la façon dont sont reliés les broches à l'intérieur)

Nous allons tester la programmation de la carte Arduino en langage Python grâce au le module **firmata** à l'aide du programme suivant :

```
#Importation du module pyfirmata
from pyfirmata import Arduino, util
#imoratation du module time
import time

#Choisir un nom pour la carte(«carte pa r exemple») et son adresse («COMX»), le X
correspond au numéro de port série sur lequel est branché votre carte.
carte = Arduino('COMX')

#es deux lignes lancent la fonction mesure en temps réel
acquisition = util.Iterator(carte)
acquisition.start()

#1
led_test= carte.get_pin('d:9:o')

time.sleep(1.0)
print("Début du test")

#2
for i in range(0,10):
    led_test.write(1)
    time.sleep(1)
    led_test.write(0)
    time.sleep(1)

print("Fin du test")
carte.exit()
```

Télécharger le programme : **SNT_pyfirmata_programme_test.py**

Compléter les commentaires #1 et #2.

Appel n°1 : Appeler le professeur pour vérification ☐

Effectuer le test sous EduPython.

Compléter le tableau suivant en expliquant l'exécution du code correspondant :

code	explication
`led_test = carte.get_pin('d:9:o')`	
`time.sleep(1.0)`	
`led_test.write(1)`	
`for i in range(0,10):`	

Étape 3 : Application à la domotique :

Application : programmation domotique d'un allumage automatique en fonction de l'éclairage :

Partie A : allumages différents en fonction d'une seule mesure

Cahier des charges :

Vous disposez de deux Led (une led rouge (nommée dans le prgramme Pyhton led_rouge) et une led verte (led_verte)), ainsi que d'un capteur de luminosité grove V1.2101020132.

Lorsque que le capteur mesure la luminosité :

si celle-ci est forte, la Led verte est allumée pendant 10 secondes tandis que la Led rouge est éteinte,

Si la luminosité est basse (proche de l'obscurité) la Led rouge est allumée et clignote (allumée 1 seconde, éteinte 1 seconde, et ainsi de suite pendant 10 secondes) et la Led verte est éteinte.

La Led rouge est reliée à la **pin 3 digitale** configurée en **SORTIE**.

La Led verte est reliée à la **pin 5 digitale** configurée en **SORTIE**.

Le capteur de luminosité est relié à la **pin A0 analogique de la carte GROVE** configurée en **ENTRÉE**.

On donne la documentation relative au capteur de luminosité :

Ce capteur se branche sur l'entrée/sortie analogique de la carte.

Le pin analogique convertie la tension **lue** (read en anglais) en un flottant dont la valeur est comprise entre 0 (Noir extrême) et 1 (luminosité maximale).

1. Ecrire l'algorithme correspondant au fonctionnement du système.

Créer un fichier **SNT_IHM_Application1.docx** sous Word et l'ajouter à votre dossier SNTIHM.

Ecrire votre l'algorithme en langage courant à l'aide des fonctionnalités de Word ou l'écrire sur papier, prendre une photo et l'insérer dans le fichier .docx

Appel n°2 : Appeler le professeur pour vérification ☐

2. Ecrire le programme python correspondant en modifiant les parties #1 et #2 du programme **SNT_pyfirmata_programme_test.py**. Enregistrer votre programme sous le nom **SNT_allumage_automatique.py**

Aide 1 : Vous devez initialiser dans la partie #1 trois variables :

1) la valeur mesurée pour la luminosité
2) l'état de la Led rouge led_rouge
3) l'état de la Led verte led_verte

Aide 2 : Vous devez utiliser les méthodes read() et write() définie en haut de la page 4.
Aide 3 : Vous allez devoir vous servir des conditionnelle « SI, ELSE » partie 5 (cf. https://monlyceenumerique.fr/snt_seconde/python/python_en_seconde.php#5) et de la boucle iterative « FOR » partie 7 sur la page du site monlyceenumerique.fr (cf. https://monlyceenumerique.fr/snt_seconde/python/python_en_seconde.php#7)

Appel n°3 : Appeler le professeur pour vérification ☐

3. Réaliser le montage et le faire fonctionner.

Appel n°4 : Appeler le professeur pour vérification ☐

Partie B : variation des allumages en fonction de mesures prises sur une durée fixée

Cahier des charges :

Vous disposez du même matériel ; lle montage ne change pas.

Désormais, votre programme fonctionnera sur une durée d'environ 30 secondes :

chaque seconde une mesure de luminosité est prise

si celle-ci est forte, la Led verte est allumée la Led rouge est éteinte

Si la luminosité est basse (proche de l'obscurité) la Led rouge est allumée et la Led verte est éteinte.

4. Ecrire l'algorithme correspondant au fonctionnement du système.

Dans le fichier **SNT_IHM_Application1.docx** :

Ecrire votre l'algorithme en langage courant à l'aide des fonctionnalités de Word ou l'écrire sur papier, prendre une photo et l'insérer dans le fichier .docx

Appel n°5 : Appeler le professeur pour vérification ☐

5. Ecrire le programme python correspondant en modifiant la partie #2 du programme **SNT_allumage_automatique.py**.
   Enregistrer votre programme sous le nom **SNT_allumage_automatique_evolutif.py**

   Aide 1 : Utiliser une variable temps_ecoule
   Aide 2 : Vous allez devoir vous servir d'une instruction répétitive conditionnelle « WHILE » partie 8 sur la page du site monlyceenumerique.fr (cf. https://monlyceenumerique.fr/snt_seconde/python/python_en_seconde.php#8).

Appel n°6 : Appeler le professeur pour vérification ☐

6. Réaliser le montage et le faire fonctionner en vérifiant l'évolution de l'allumage en fonction de la luminosité.

Appel n°7 : Appeler le professeur pour vérification ☐

7. Comment modifier le programme **SNT_allumage_automatique_evolutif.py** afin qu'il fonctionne désormais indéfiniment ?

Appel n°8 : Appeler le professeur pour vérification ☐

Annexe :

Il y a plusieurs façon pour définir et reconnaitre le sens d'une LED :

Première manière :

Sur le composant :

- l'**anode** est le côté où **la patte** du composant est **la plus longue**.
- la cathode est donc la patte la plus courte.

Deuxième manière :

Vu du dessus, la LED n'est pas totalement circulaire, il y a toujours un côté plat. Ce côté plat représente la cathode. Cette astuce est indispensable pour reconnaitre la cathode lorsque les pattes sont coupées.

Sur un schéma électrique, la cathode est représentée par le côté où il y a le bord plat à droite du triangle.