

Programmation d'une entrée

définir le mode de fonctionnement de la broche

```
GPIO.setup(23,GPIO.IN) # on déclare la pin 23 comme une entrée
```

```
GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN) # on affecte une résistace de pull-down , résistance entre l'entrée du signal et la masse)
```

#Une méthode d'acquisition de l'état de l'entrée :

Créer une variable

```
etat_entre_23 == 0 # variable crée et mise a l'état 0
```

Acquisition de la valeur de l'entée

```
GPIO.input(23) == etat_entre_23 # lecture de l'état de l'entrée et enregistrement dans la variable
```

Le Raspberry Pi

Les opérateurs pour les conditions :

- == égal à
- != différent de
- < plus petit que
- > plus grand que
- <= inférieur ou égal à
- >= supérieur ou égal à

Conditions multiples :

- and (ET)
- or (OU)
- in / not in (DANS / PAS DANS)

Mots clé des conditions :

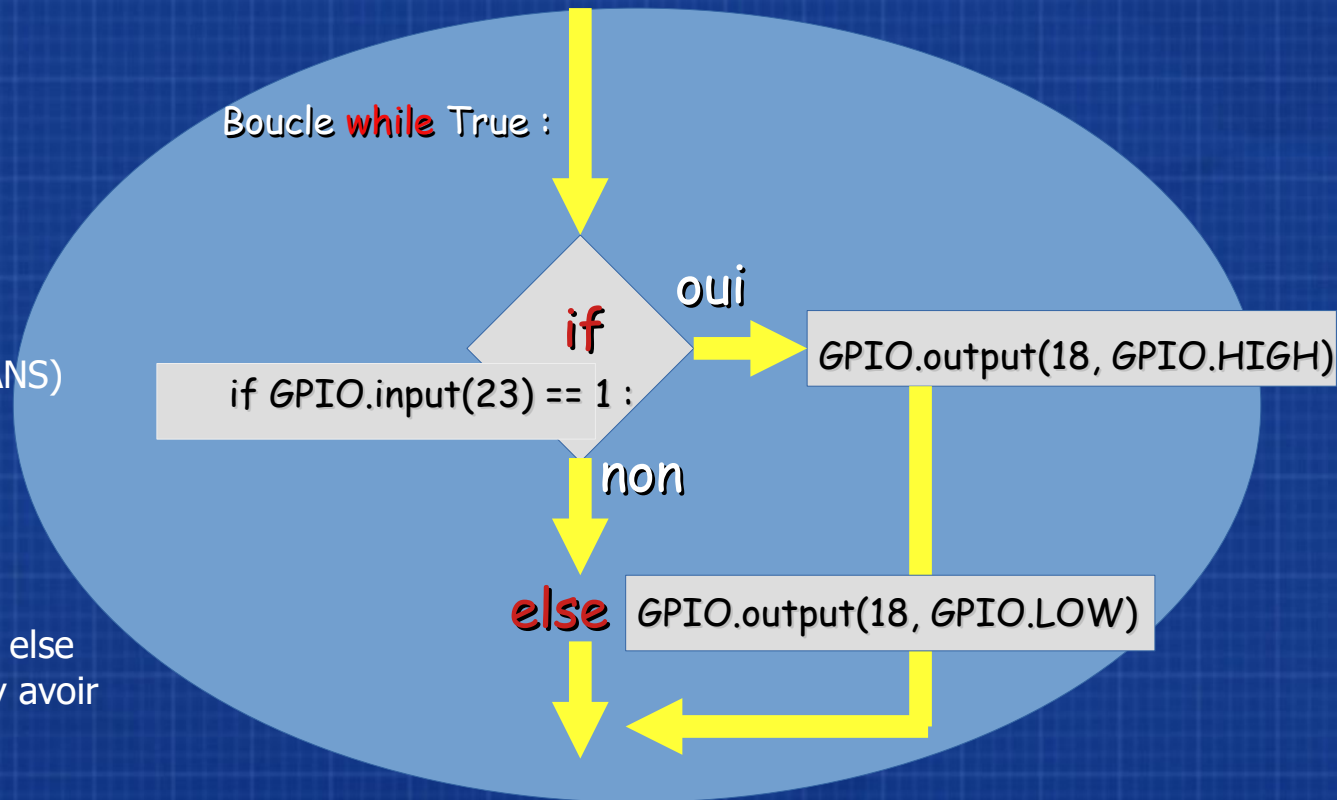
- if SI
- If not SI N'EST PAS
- else SINON un seul else
- elif SINON SI il peut y avoir plusieurs elif

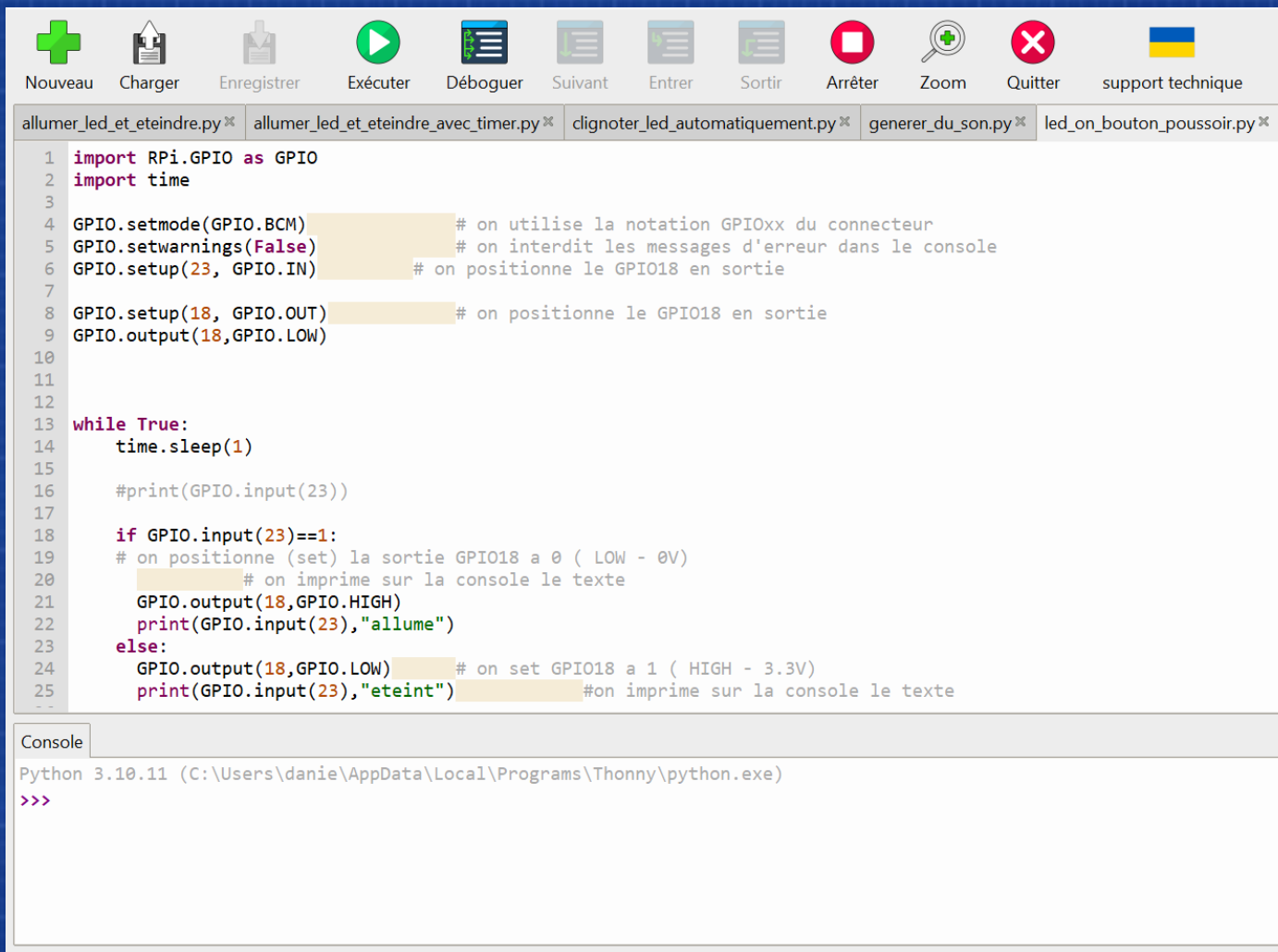
```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(23, GPIO.IN)
GPIO.output(18, GPIO.LOW)
```

Programmation

Combinaisons des conditions :

if 0 < age <= 100 :



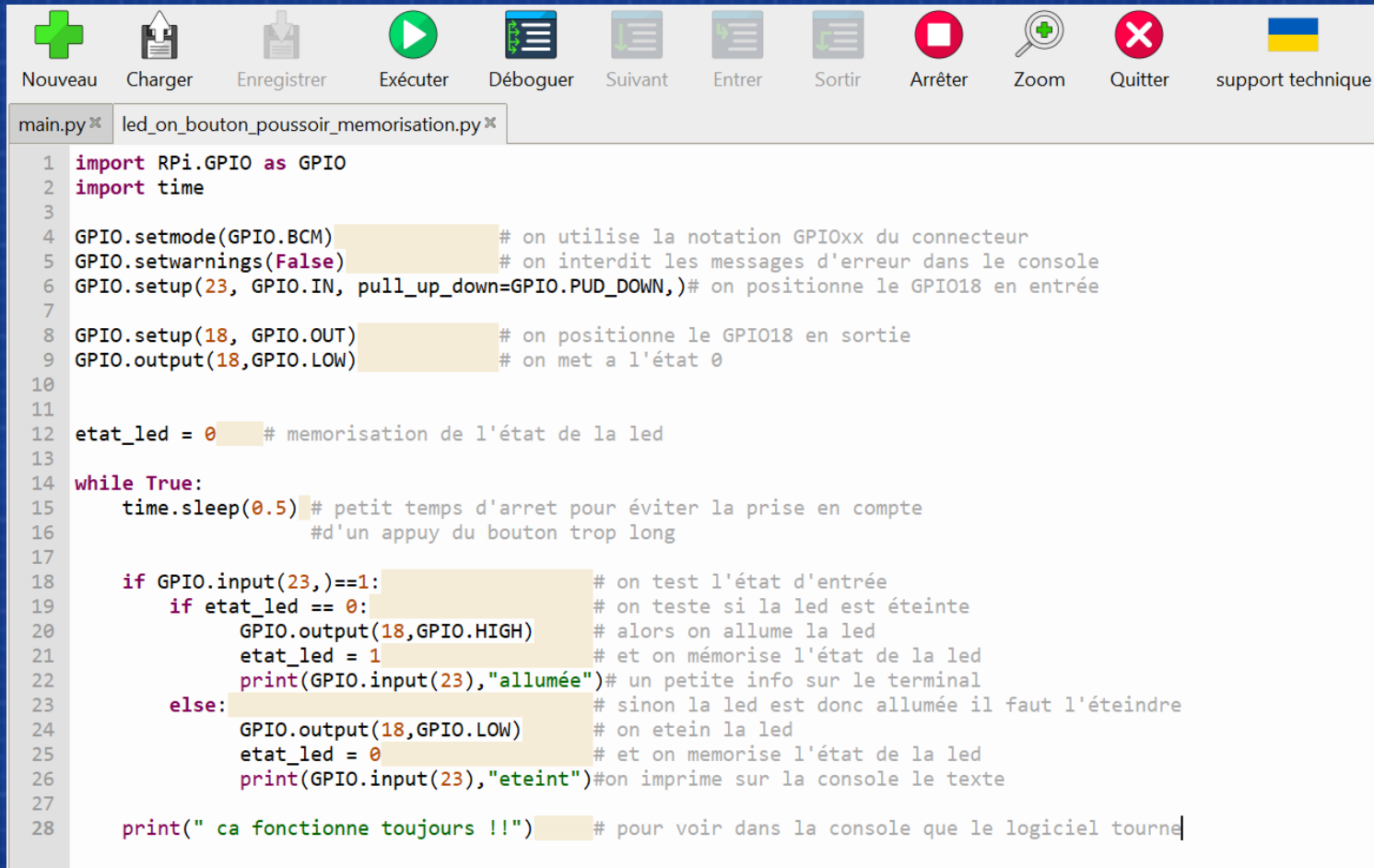


The screenshot shows a Python IDE window with a toolbar at the top containing icons for 'Nouveau', 'Charger', 'Enregistrer', 'Exécuter', 'Débuguer', 'Suivant', 'Entrer', 'Sortir', 'Arrêter', 'Zoom', 'Quitter', and 'support technique'. Below the toolbar, several Python files are open in tabs: 'allumer_led_et_eteindre.py', 'allumer_led_et_eteindre_avec_timer.py', 'clignoter_led_automatiquement.py', 'generer_du_son.py', and 'led_on_bouton_poussoir.py'. The active file is 'led_on_bouton_poussoir.py', which contains the following Python code:

```
1 import RPi.GPIO as GPIO
2 import time
3
4 GPIO.setmode(GPIO.BCM) # on utilise la notation GPIOxx du connecteur
5 GPIO.setwarnings(False) # on interdit les messages d'erreur dans le console
6 GPIO.setup(23, GPIO.IN) # on positionne le GPIO18 en sortie
7
8 GPIO.setup(18, GPIO.OUT) # on positionne le GPIO18 en sortie
9 GPIO.output(18,GPIO.LOW)
10
11
12
13 while True:
14     time.sleep(1)
15
16     #print(GPIO.input(23))
17
18     if GPIO.input(23)==1:
19         # on positionne (set) la sortie GPIO18 a 0 ( LOW - 0V)
20         # on imprime sur la console le texte
21         GPIO.output(18,GPIO.HIGH)
22         print(GPIO.input(23),"allume")
23     else:
24         GPIO.output(18,GPIO.LOW) # on set GPIO18 a 1 ( HIGH - 3.3V)
25         print(GPIO.input(23),"eteint") #on imprime sur la console le texte
--
```

Below the code editor is a 'Console' window showing the Python version and the path to the Python executable:

```
Python 3.10.11 (C:\Users\danie\AppData\Local\Programs\Thonny\python.exe)
>>>
```



The image shows a terminal window on a Raspberry Pi. The window title is "main.py x led_on_bouton_poussoir_memorisation.py x". The code is written in Python and controls an LED connected to GPIO pin 18 and a button connected to GPIO pin 23. The code includes comments in French explaining the setup and the logic of the program. The code is as follows:

```
1 import RPi.GPIO as GPIO
2 import time
3
4 GPIO.setmode(GPIO.BCM) # on utilise la notation GPIOxx du connecteur
5 GPIO.setwarnings(False) # on interdit les messages d'erreur dans le console
6 GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN,)# on positionne le GPIO18 en entrée
7
8 GPIO.setup(18, GPIO.OUT) # on positionne le GPIO18 en sortie
9 GPIO.output(18,GPIO.LOW) # on met a l'état 0
10
11
12 etat_led = 0 # memorisation de l'état de la led
13
14 while True:
15     time.sleep(0.5) # petit temps d'arrêt pour éviter la prise en compte
16                     #d'un appuy du bouton trop long
17
18     if GPIO.input(23)==1: # on test l'état d'entrée
19         if etat_led == 0: # on teste si la led est éteinte
20             GPIO.output(18,GPIO.HIGH) # alors on allume la led
21             etat_led = 1 # et on mémorise l'état de la led
22             print(GPIO.input(23),"allumée")# un petite info sur le terminal
23         else: # sinon la led est donc allumée il faut l'éteindre
24             GPIO.output(18,GPIO.LOW) # on etein la led
25             etat_led = 0 # et on memorise l'état de la led
26             print(GPIO.input(23),"eteint")#on imprime sur la console le texte
27
28     print(" ca fonctionne toujours !!") # pour voir dans la console que le logiciel tourne
```