

Python pour le microcontrôleur micro:bit

```

from microbit import *
en début de programme, pour pouvoir utiliser les fonctions du microcontrôleur

while True:
    ....instructions de la boucle
    ....instructions de la boucle
répète les instructions indéfiniment

note = 12
la variable note vaut maintenant 12

note = note + 1
la variable note vaut maintenant son ancienne valeur + 1

*                               /                               **
multiplication                division                puissance

display.scroll("Bonjour")
fait défiler la chaîne de caractères "Bonjour" sur l'afficheur

display.scroll(note)
fait défiler la valeur de la variable note sur l'afficheur

display.show("B")
affiche la chaîne de caractères "B" sur l'afficheur

display.clear()
efface l'afficheur

# on peut écrire ici ce que l'on veut
après "#" apparait un commentaire pour les personnes qui lisent le programme (non pris en compte par Python)

if button_a.is_pressed():
    ....instructions a effectuer si la condition est vérifiée
    ....instructions a effectuer si la condition est vérifiée
    instructions suivante (que la condition soit vérifiée ou non)
    instructions suivante (que la condition soit vérifiée ou non)
    si le bouton A est actuellement pressé alors ...

if not button_b.is_pressed():
    si le bouton B est actuellement non pressé alors

if note == 18:
    si la variable note vaut 18 alors

if not (note > 10):
    si la variable note n'est pas supérieure à 10 alors

!=                               or                               and
différent de                    ou                               et

sleep(500)
fait une pause de 500 ms

print("Bonjour")
écrit la chaîne de caractères "Bonjour" (en utilisant le REPL de Mu et un RESET de la carte)

print(note)
écrit la valeur de la variable note (en utilisant le REPL de Mu et un RESET de la carte)

print("La dernière note est", note, "sur 20.")

```

écrit la chaîne de caractères en remplaçant la variable note par sa valeur (en utilisant le REPL de Mu et un RESET de la carte)

```
if moyenne2 > moyenne1:
    print("En progrès")
else:
    print("Ce n'est pas mieux")
si moyenne2 > moyenne1 alors écrire "En progrès"
sinon écrire "Ce n'est pas mieux"

if moyenne >= 12:
    statut = "admis"
    mention = "avec mention"
elif moyenne >= 10:
    statut = "admis"
    mention = "sans mention"
else:
    statut = "non admis"
    mention = ""
print("Avec une moyenne de", moyenne, "vous êtes", statut, mention)
si moyenne supérieure ou égale à 12 alors ...
sinon, si moyenne supérieure ou égale à 10 alors ...
sinon ...
écrit la moyenne, le statut et la mention
```

```
etat_broche = pin0.read_digital()
la variable etat_broche vaut maintenant 1 si la broche n°0 reçoit un signal (tension d'environ 3,3V)
ou 0 si elle ne reçoit pas de signal (tension de 0V)
entrées "numériques" binaires en priorité sur les broches n°0, 1, 2, 8, 13, 14, 15 et 16
```

```
if pin0.read_digital():
la condition est vérifiée si la broche n°0 reçoit un signal (c'est-à-dire si elle est à environ 3,3V)
et elle est non vérifiée si la broche n°0 ne reçoit pas de signal (c'est-à-dire si elle est à 0V)
entrées "numériques" binaires en priorité sur les broches n°0, 1, 2, 8, 13, 14, 15 et 16
```

```
valeur_lue = pin0.read_analog()
la variable valeur_lue vaut maintenant un entier compris entre 0 et 1023 proportionnel à la tension de la broche n°0
(cette tension étant comprise entre 0V et environ 3,3V)
entrées "analogiques" possibles sur les broches n°0, 1 et 2 (et, si l'afficheur est désactivé, n°3, 4 et 10)
```

```
pin0.write_digital(d)
avec d = 1 : envoie un signal sur la broche n°0 (tension d'environ 3,3V)
avec d = 0 : n'envoie pas de signal sur la broche n°0 (tension de 0V)
sorties "numériques" binaires (tout ou rien) en priorité sur les broches n°0, 1, 2, 8, 13, 14, 15 et 16
```

```
pin0.write_analog(n)
met la sortie de la broche n°0 à une tension comprise entre 0V et environ 3,3V proportionnelle à n (où n est un entier
compris entre 0 et 1023)
sorties PWM (équivalentes "analogiques") en priorité sur les broches n°0, 1, 2, 8, 13, 14, 15 et 16 (maximum 3 en
même temps)
```

```
while compteur < 10:
    ...instructions de la boucle
    ...instructions de la boucle
instructions suivante
répète les instructions tant que la variable compteur est inférieure à 10
```

```
texte_lu = input()
attend que l'utilisateur écrive puis valide avec la touche ENTER (en utilisant le REPL de Mu et un RESET de la carte)
ce qui a été écrit est stocké sous forme de chaîne de caractères dans la variable texte_lu
```

```
valeur_lue = int(texte_lu)      ou      valeur_lue = float(texte_lu)
converti la chaîne de caractères texte_lu en un nombre entier (ou en un nombre décimal) et met ce nombre dans la
variable valeur_lue
```