1 Carte micro:bit

1. présentation

Exo 1.1 Hello World

1. Connecter la carte micro:bit à l'ordinateur à l'aide du câble USB.
2. Ouvrir le logiciel Mu ou, sur un ordinateur personnel, se connecter à https://python.microbit.org/v/3
3. Saisir le code suivant dans l'éditeur :

```
1 from microbit import *
2
3 display.scroll("Bonjour")
```

4. Envoyer le code avec « Flasher » ou, sur un ordinateur personnel, Send to micro:bit
5. La première ligne de code permet d'augmenter le langage Python des instructions spécifiques à la carte. (Elle sera supposée toujours présente pour la suite.)
   Que provoque l'autre ligne de code ?
   (Pour réexécuter le programme, appuyer sur le bouton « RESET » au dos de la carte à côté de la connexion USB)

Exo 1.2 Avec une boucle « Tant que »

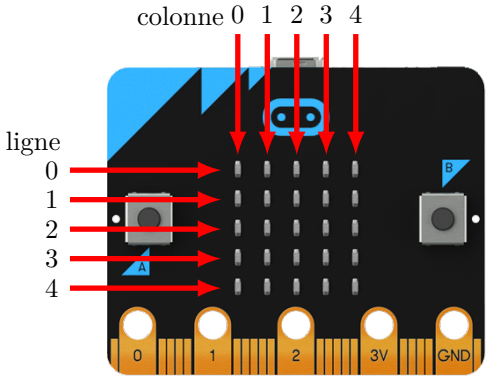
1. Pour que l'instruction se répète indéfiniment, on peut la positionner dans une boucle « tant que » avec une condition toujours « vraie ». Tester la modification :

```
3 while True :
4     display.scroll("Bonjour")
5     sleep(5000)
```

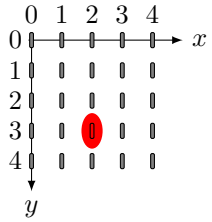
   À quoi sert l'instruction `sleep(5000)` ?
2. La condition peut avantageusement être dépendante de l'état d'un capteur.
   Par exemple, tester et expliquer ce que provoque les changements ci-dessous :

```
3 while not button_a.was_pressed() :
4     display.scroll("Bonjour")
5     sleep(5000)
6 display.show(Image.HAPPY)
```

2. Actionneurs : LED



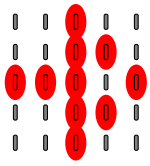
Formalisation :



La LED est allumée en $x = 2$ et $y = 3$ avec l'intensité maximale (9 sur 9) après l'instruction :
`display.set_pixel(2,3,9)`
(Pour l'éteindre : `display.set_pixel(2,3,0)`)

Exo 1.3 Affichage LED par LED

1. Allumer chaque LED représentant la flèche suivante :



2. Comme toutes les LED de la colonne 2 doivent être allumées, on se retrouve à écrire 5 fois « `display.set_pixel(2,...,9)` » avec les entiers de 0 à 4 à la place des ....

   Cela peut être dit, dans le langage Python, de la manière suivante :

```
1 for y in [0, 1, 2, 3, 4] :
2     display.set_pixel(2,y,9)
```

   a. Recopier et tester ces deux lignes.
   b. Les compléter pour afficher un « + ».
3. À partie du « + » obtenue à la question précédente, ajouter 3 instructions seulement pour dessinée la flèche proposée à la question 1.

Exo 1.4 Allumage
Compléter le programme suivant pour que chaque LED s'allume une par une, colonne par colonne.

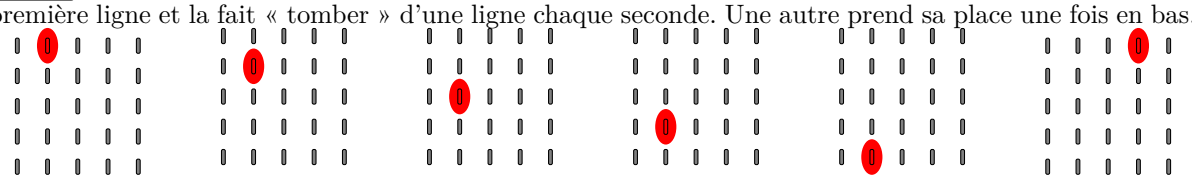
```
for x in ... :
    for y in ... :
        display.set_pixel(x,y,9)
        sleep(500)
```

Exo 1.5 Compléter le programme de l'exercice précédent, pour qu'ensuite, les LED s'éteigne, une à une, ligne par ligne, en commençant par le coin inférieur droit.

Exo 1.6 Utilisation des boutons
L'instruction `display.show('a')` affiche le caractère `'a'`.
On rappelle, d'autre part, que l'instruction `str(1)` transforme l'entier `1` en le caractère `'1'`.

1. En combinant ces deux instructions, afficher l'entier 1.
2. Réaliser un compteur, de 0 à 9, augmentant de 1 chaque seconde.
3. Réaliser un compte à rebours de 9 à 0.

Exo 1.7 Proposer le code d'un programme de goutte de pluie, qui allume une LED au hasard sur la première ligne et la fait « tomber » d'une ligne chaque seconde. Une autre prend sa place une fois en bas.



3. Capteurs : les boutons

La carte possède deux boutons A (à gauche) et B (à droite).
On peut tester, à tout moment, s'ils ont été actionnés à l'aide de `button_a.was_pressed()` ou `button_b.was_pressed()`.

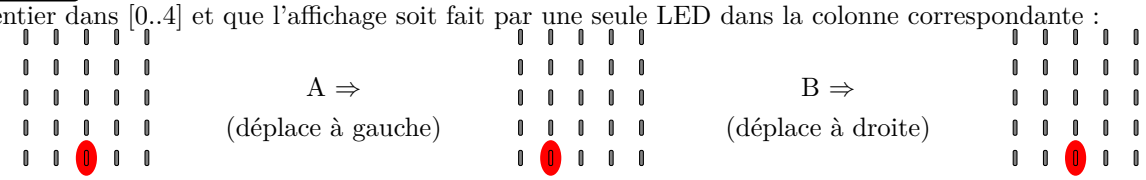
Exo 1.8 Compteur
On propose le programme suivant :

```
n = 0
while n<10 :
    display.show(str(n))
    if button_b.was_pressed() :
        n = n + 1
display.show('*')
```

1. Tester ce programme : que fait-il ?
2. Le modifier pour qu'il compte à rebours.

Exo 1.9 programmer la carte micro:bit pour qu'elle affiche un entier entre 0 et 9,
⋆ qui augmente à chaque appuis du bouton B ;
⋆ qui diminue à chaque appuis du bouton A ;
⋆ sans jamais dépasser [0..9]

Exo 1.10 Modifier le programme de l'exercice précédent pour limiter l'augmentation et la diminution de l'entier dans [0..4] et que l'affichage soit fait par une seule LED dans la colonne correspondante :



4. Un mini-jeu

Exo 1.11 Programme un mini jeu de course à obstacles :
⋆ Utiliser l'exercice 6.7 pour simuler l'arrivée d'obstacles
⋆ Utiliser l'exercice 6.10 pour simuler le coureur passant d'une piste à l'autre
⋆ Compléter en calculant le nombre d'obstacles évités et (plus difficile) en augmentant la vitesse des obstacles quand le score augmente.