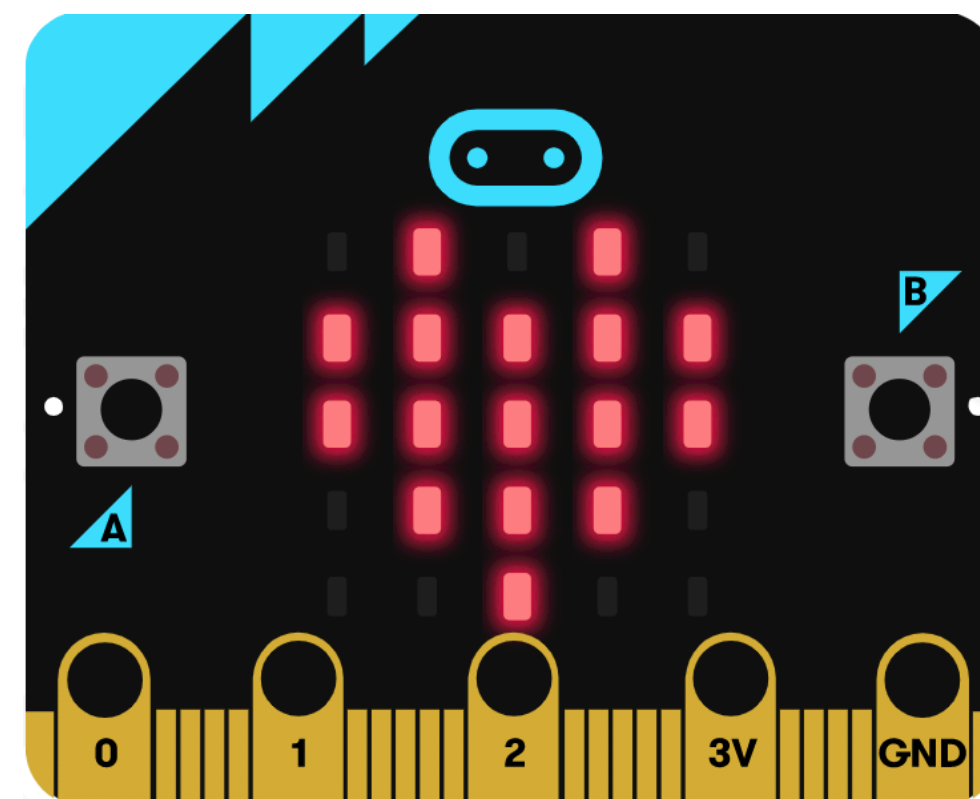


Programmer la carte micro:bit

avec python



Programmer la carte micro:bit

- Le dispositif Yes we code! de la fondation CGénial
- Programmer avec Mu / Thonny / python.microbit
- Les fonctionnalités de la carte
- Premiers exemples
- Le ruban de led Neopixel
- Le kit Grove
- Le robot Cutebot
- Idées de projet

[Lien vers le site académique SNT-NSI](#)

Nathalie Weibel (DRANE de Normandie)

avec python

Le dispositif Yes we code!

Richard Fuentes, Alexandra Costrachevici

- Informations : le site de la fondation CGénial
- Yes We Code! en Normandie : carte géographique
- Le padlet Yes We Code!



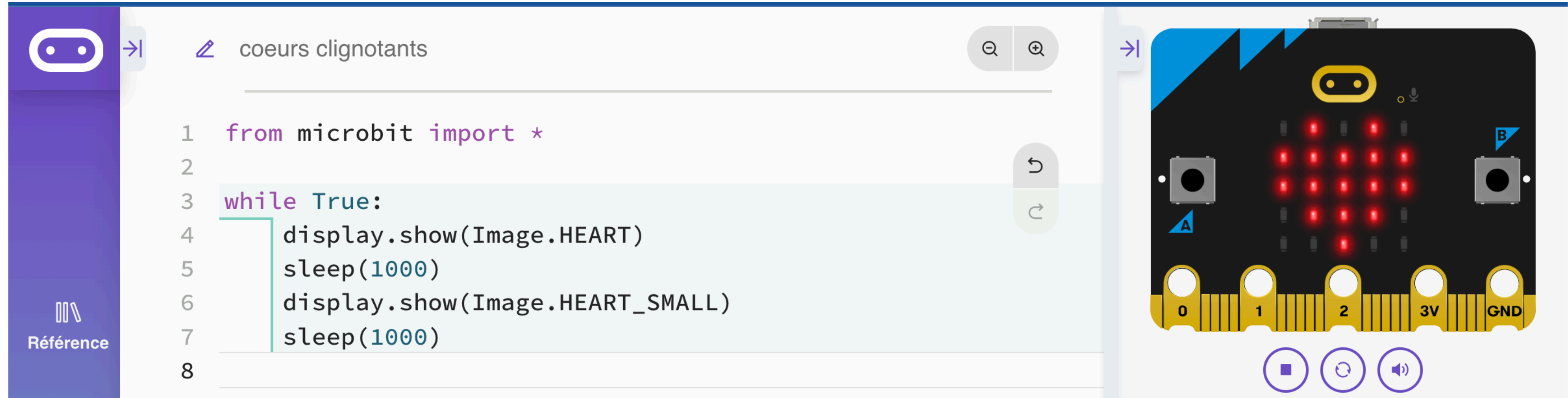
Programmer avec un logiciel : Mu ou Thonny



v. 1.1.0

- **Mu** et **Thonny** : éditeurs de code multiplateforme, avec un mode micro:bit
- flashage des programmes sur la carte
- auto-complétion (Mu), vérification du code, debuggage (Thonny)
- **affichage graphique** des données, console interactive REPL

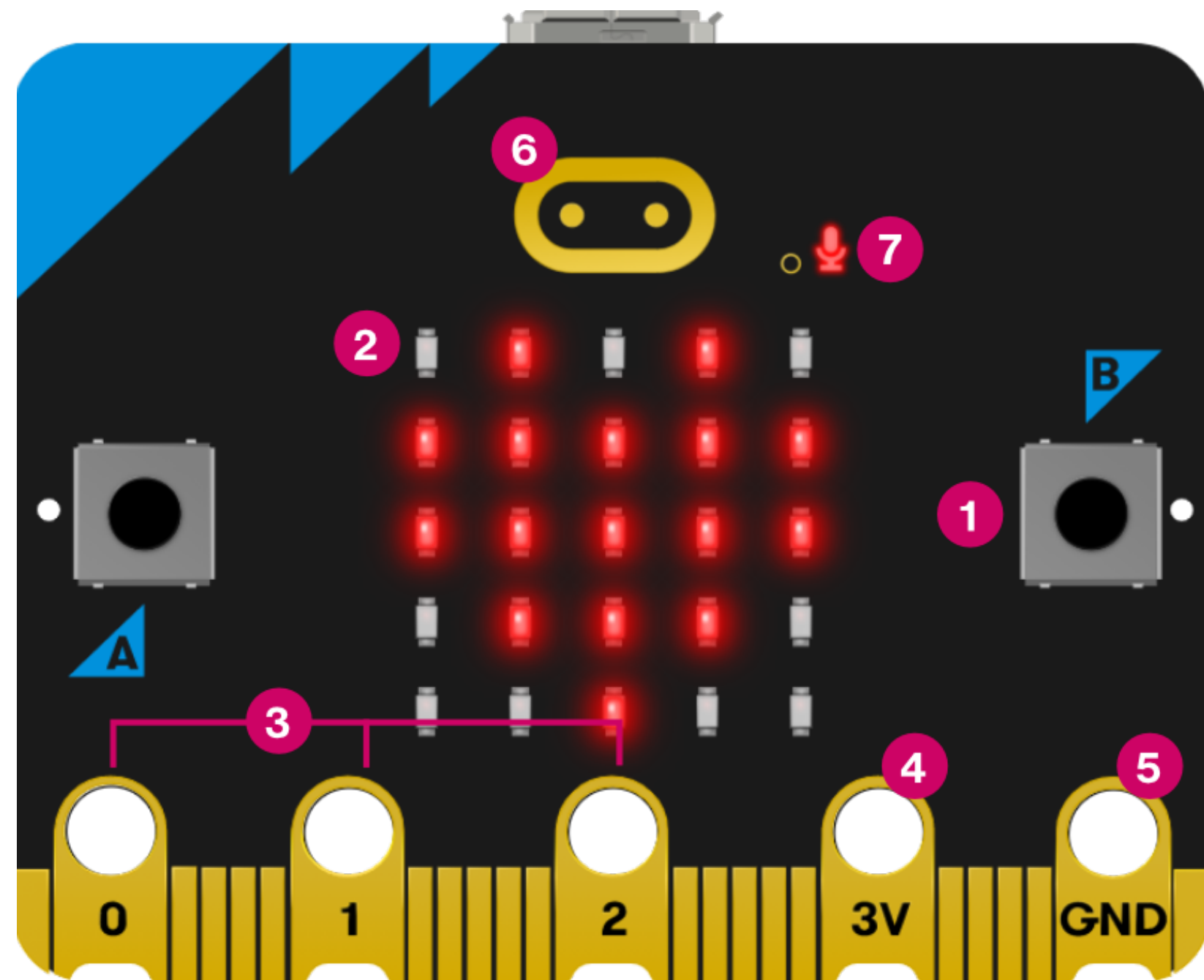
Programmer en ligne : python.microbit.org/



- python.microbit.org : éditeur python, avec simulateur
- flashage des programmes sur la carte
- auto-complétion, références
- affichage graphique des données

Pour certains usages on pourra préférer : Vittascience, Edublocks

Les fonctionnalités de la carte



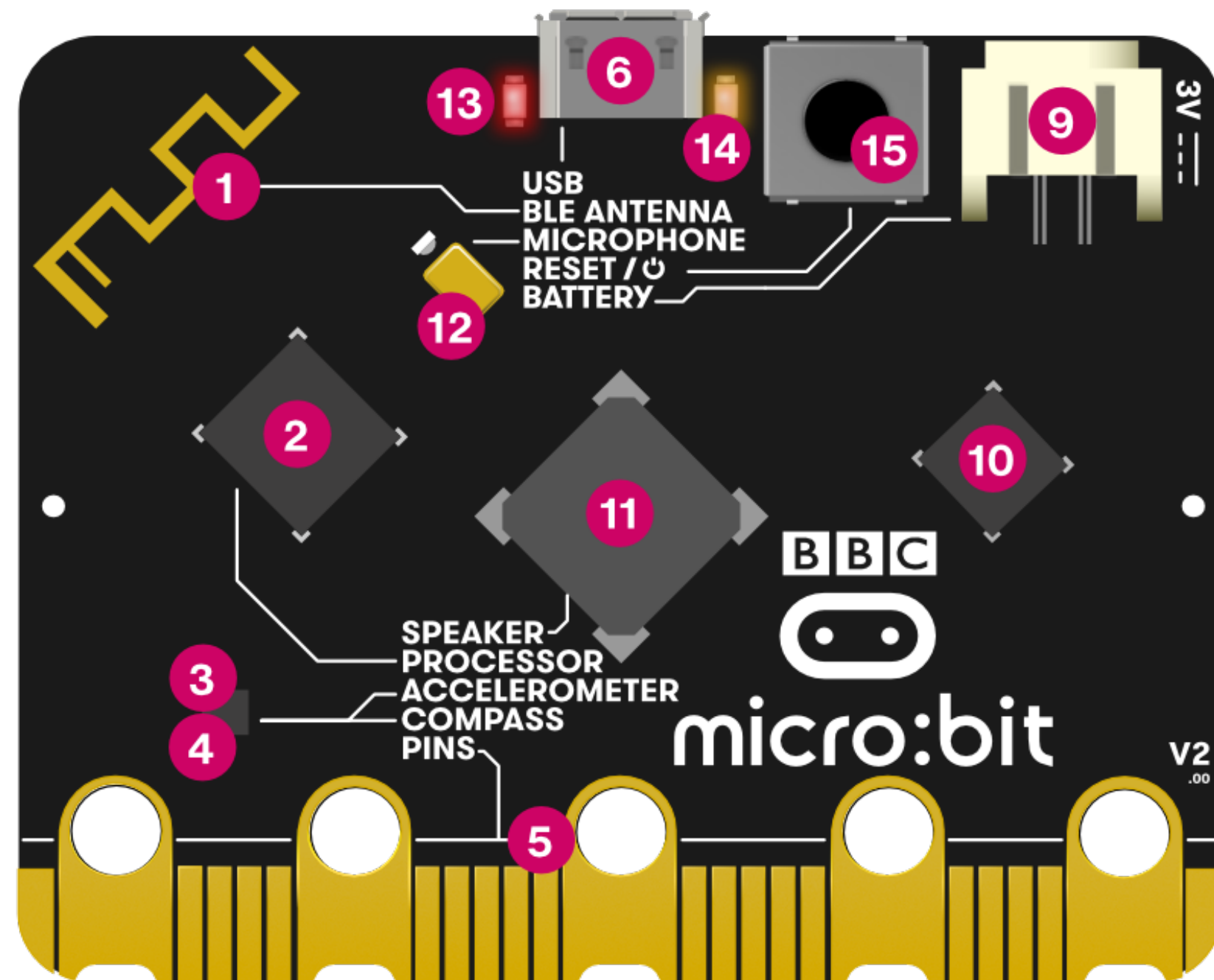
à l'avant :

- 1 : deux boutons poussoirs A et B programmables
- 2 : 25 LEDs rouges, programmables
- 3, 4, 5 : des broches de connexion, d'alimentation.

Et sur la V2 :

- 6 : un logo tactile
- 7 : une led témoin du microphone

Les fonctionnalités de la carte



à l'arrière :

1. une antenne **radio** et bluetooth,
2. un **micro-processeur** et capteur de température,
3. un **magnétomètre (boussole)**,
4. un **accéléromètre**,
5. des **broches de connexion**,
6. un **port micro-USB**,
9. un connecteur d'alimentation (2 piles AAA – 3V),
10. une puce d'interface USB

Et sur la V2 uniquement :

11. un haut-parleur
12. un microphone
13. une LED rouge, témoin d'alimentation
14. une LED jaune, témoin de communication USB
15. un bouton de réinitialisation et de marche/arrêt

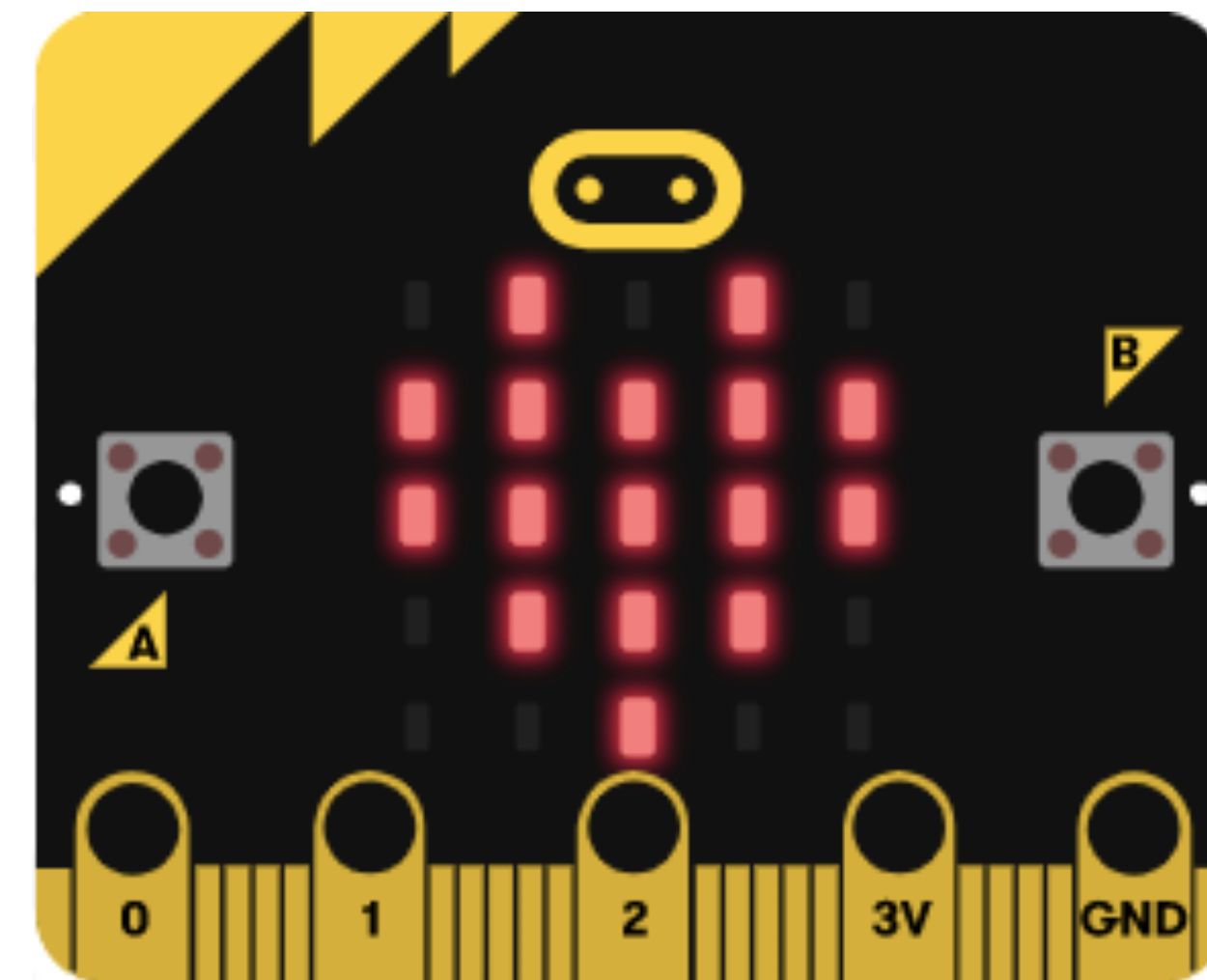
Parcours de découverte

Lien vers le [parcours de découverte](#)

Exercice 1

```
display.scroll("Hello, World!")
```

```
display.show(Image.HEART)
```

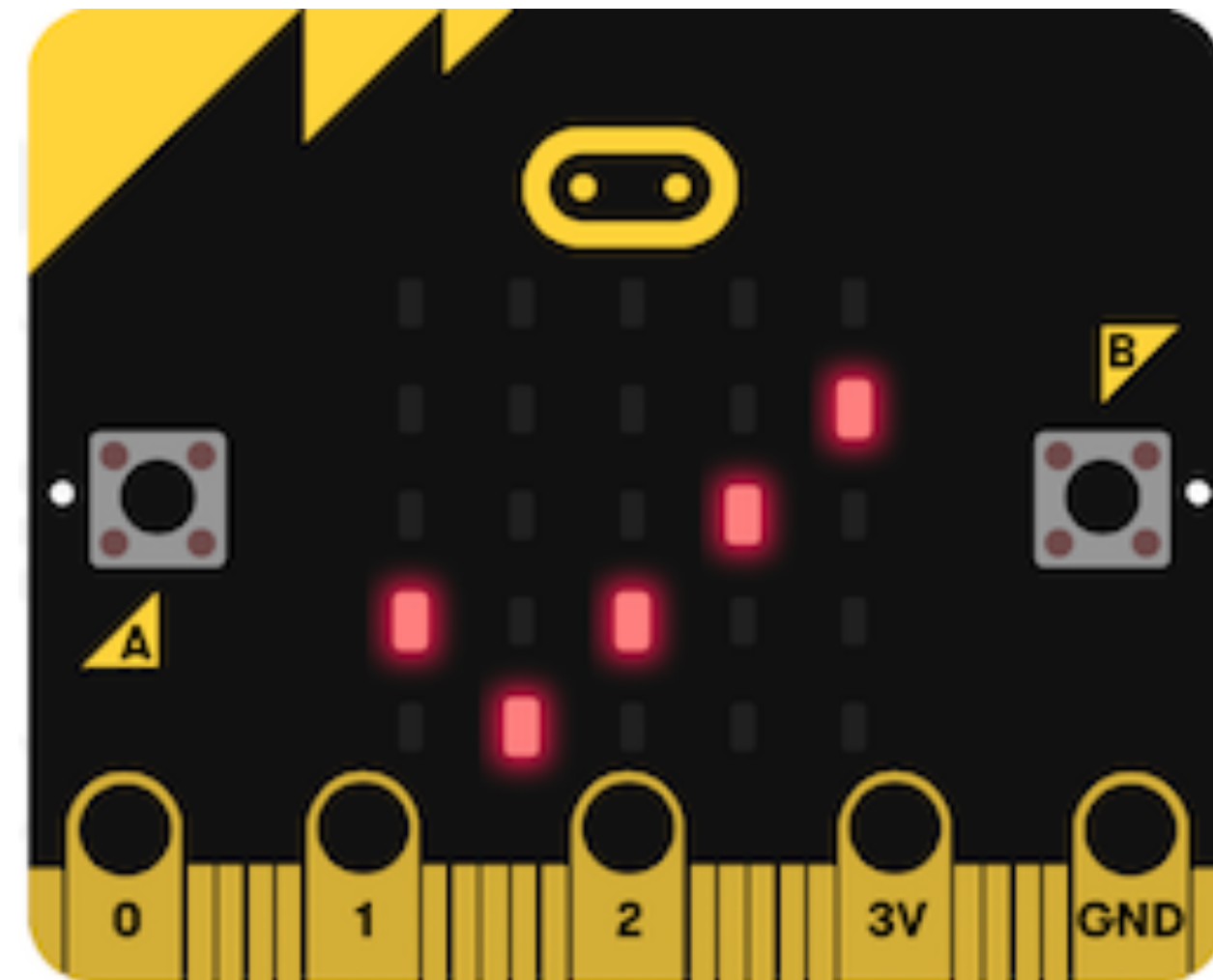


Afficher un texte, une image

Parcours de découverte

Exercice 2

```
from microbit import *  
display.show("?")  
while True:  
    if button_a.is_pressed():  
        display.show(Image.YES)  
    elif button_b.is_pressed():  
        display.show(Image.NO)
```

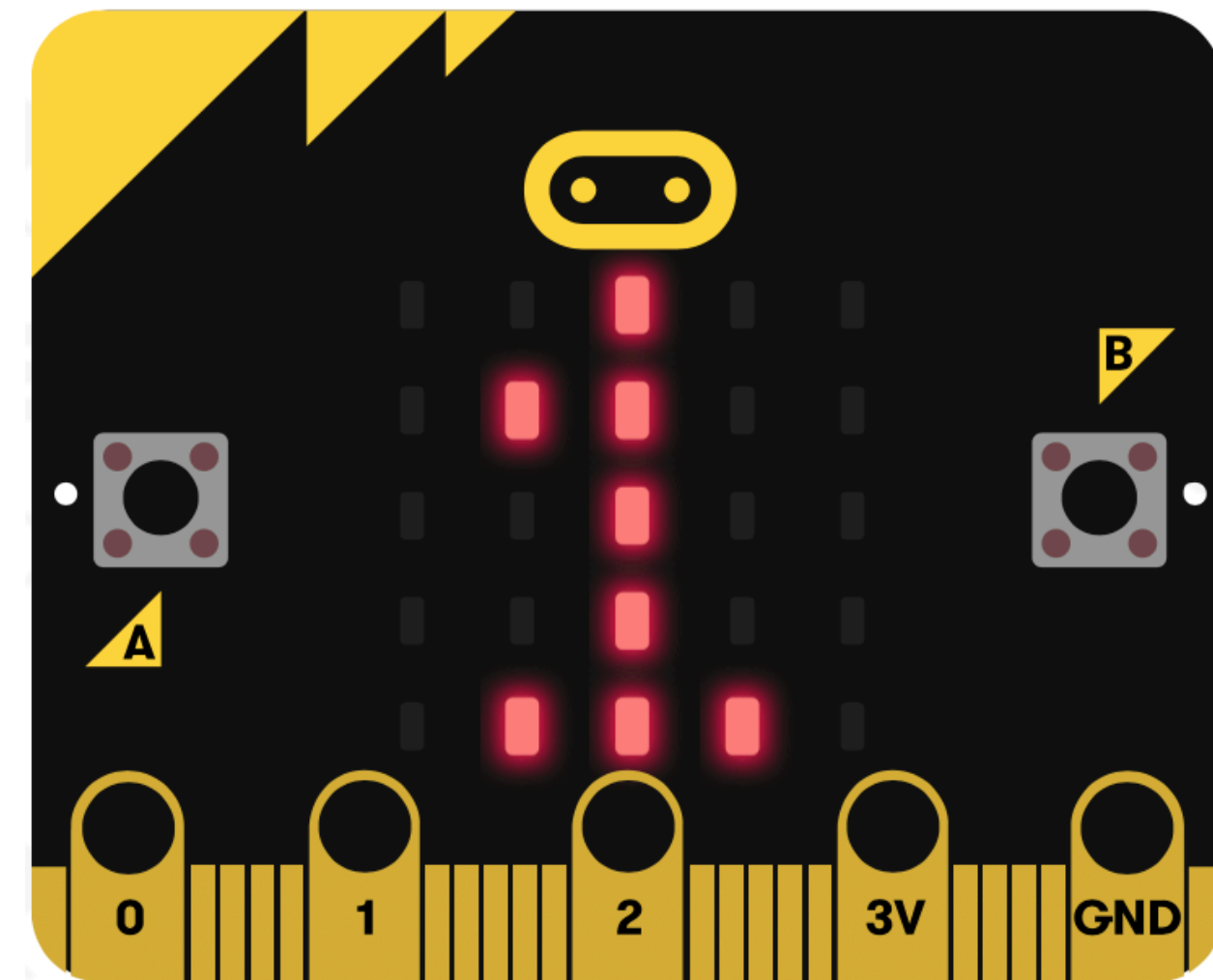


Utiliser les boutons, les instructions conditionnelles

Parcours de découverte

Exercice 3

```
from microbit import *  
while True :  
    display.show(1)  
    sleep(500)  
    display.clear()  
    sleep(500)  
  
for k in range(10):  
    display.show(k)  
    sleep(500)
```

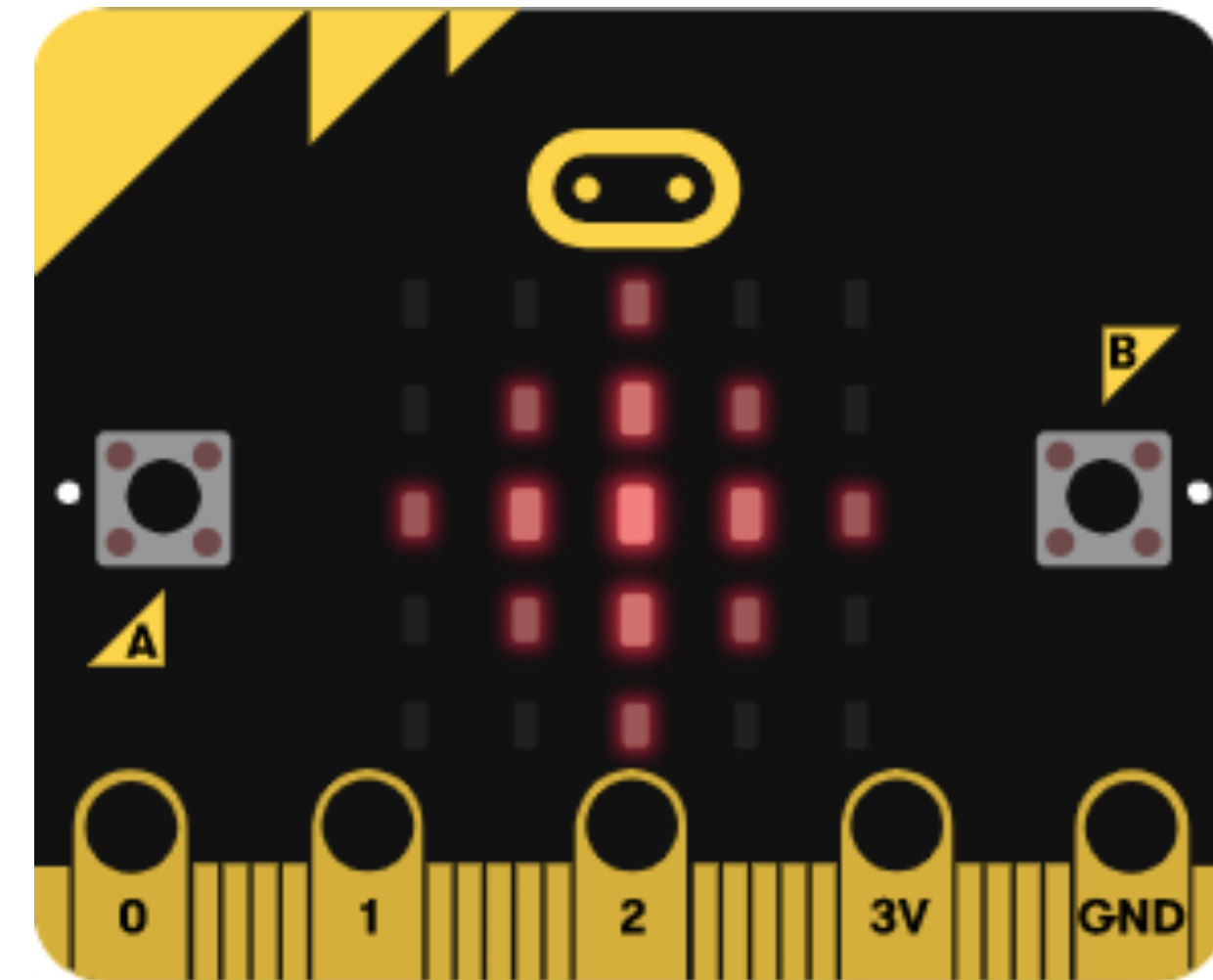


Faire clignoter un affichage. Utiliser des boucles

Parcours de découverte

Exercice 4

```
from microbit import *  
eclat1 = Image("00300:03630:36963:03630:00300")  
eclat2 = Image("00300:03330:33333:03330:00300")  
while True:  
    if button_a.is_pressed():  
        display.show(eclat1)  
        sleep(1000)  
        display.show(eclat2)
```



Création d'images – Choix aléatoire – Gestes

Parcours de découverte

Exercice 5

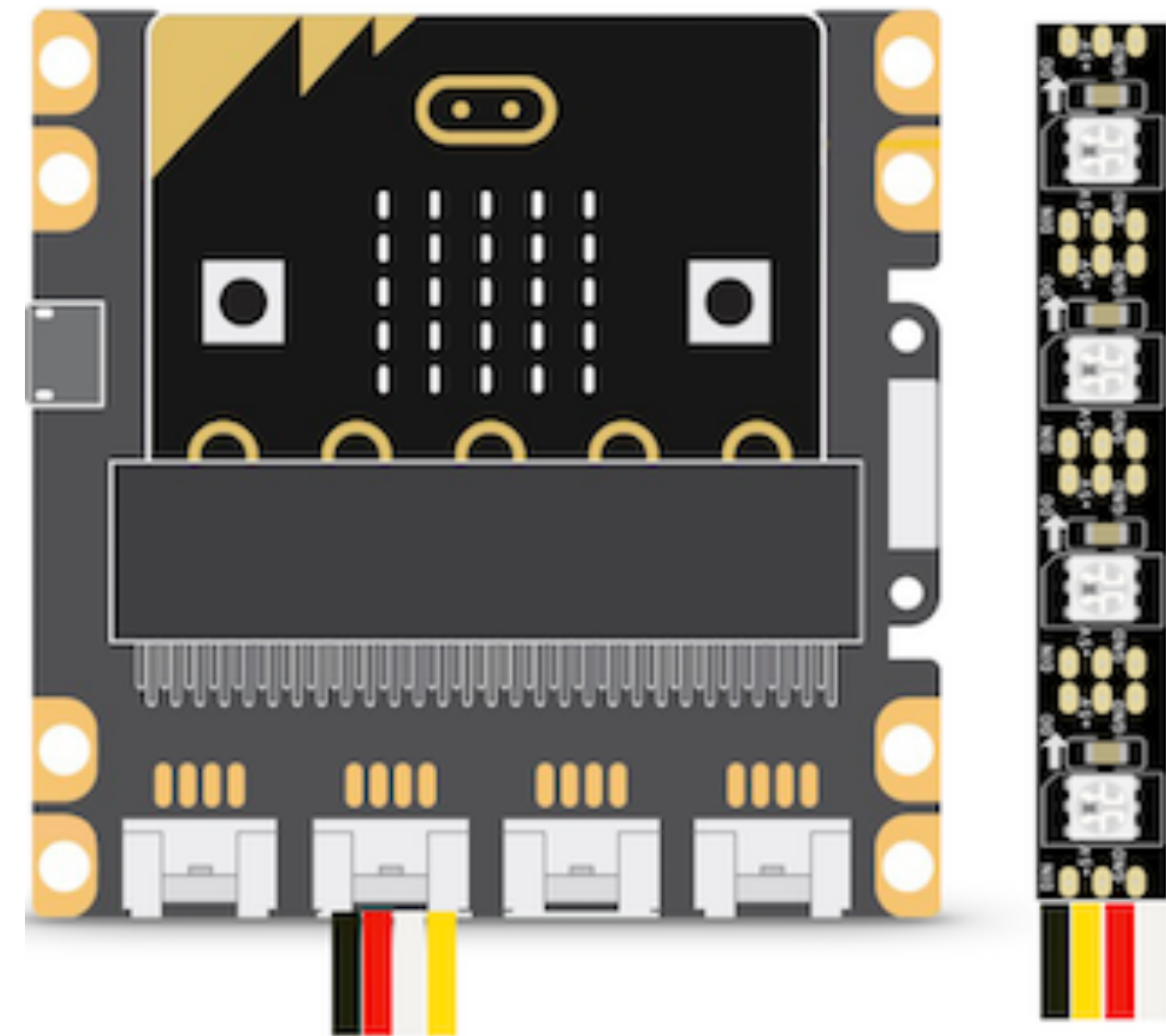
```
from microbit import *
import radio
radio.on()
vote_ok = True
while vote_ok:
    display.show(Image.SQUARE_SMALL)
    sleep(100)
    if button_a.is_pressed():
        display.show(Image.YES)
```

```
elif button_b.is_pressed():
    display.show(Image.NO)
    if button_a.was_pressed():
        radio.send("A")
        vote_ok = False
    elif button_b.was_pressed():
        radio.send("B")
        vote_ok = False
```

Parcours de découverte

Exercice 6

```
from microbit import *  
import neopixel  
np = neopixel.NeoPixel(pin0, 30)  
while True:  
    for i in range(30):  
        np[i] = (255, 0, 0)  
        sleep(200)  
        np.show()  
    np.clear()
```



Ruban de leds Neopixel

Défi : Jeu de Nim sur un ruban NeoPixel

Matériel : 3 cartes micro:bit, 1 ruban neopixel

Fonctionnement attendu :

- Deux joueurs s'affrontent au jeu de Nim. Chacun joue à l'aide d'une carte micro:bit. L'un des joueurs a la couleur rouge, l'autre la couleur bleu.
- Le ruban NeoPixel est piloté par une 3e carte : il est totalement allumé et toutes ses leds sont éclairées en blanc.
- Les joueurs doivent s'approprier tour à tour entre une et trois leds qui s'afficheront en rouge ou bleu selon le joueur. Celui qui s'empare de la dernière led du ruban gagne la partie.

Fonctionnement

Défi : Jeu de Nim sur un ruban NeoPixel

- Chacun leur tour, les joueurs appuient 1, 2 ou 3 fois sur le bouton A, et valident leur choix avec le bouton B. La carte affiche les chiffres 1, 2 ou 3 lors des appuis sur le bouton A et l'image `Image.YES` lors de la validation avec le bouton B.
- La valeur choisie est envoyée par radio à la carte qui pilote le ruban de led. Le nombre correspondant de led(s) s'allume en couleur sur le ruban, en rouge ou bleu selon le joueur.
- Lorsque la dernière led est atteinte, le ruban fait clignoter ses leds de la couleur du joueur gagnant.

Fonctionnement

Défi : Jeu de Nim sur un ruban NeoPixel

Programme sur la carte du joueur A

```
from microbit import *
import radio
radio.config(group=42)
radio.on()

nombre = 0
while True:
    if button_a.was_pressed() and nombre < 3:
        nombre = nombre + 1
        display.show(nombre)
        sleep(100)
    if button_b.was_pressed():
        radio.send('R' + str(nombre))
        display.show(Image.YES)
        nombre = 0
        sleep(500)
```

Programme sur la carte du joueur B

Programme identique en
remplaçant R par B

cartes des joueurs

Défi : Jeu de Nim sur un ruban NeoPixel

Programme sur la carte pilotant le ruban NeoPixel

```
from microbit import *
import neopixel
import radio
radio.config(group=42)
radio.on()
np = neopixel.NeoPixel(pin0, 30)

rouge = (240, 0, 0)
bleu = (0, 0, 240)
blanc = (200, 200, 200)

for x in range(30):
    np[x] = blanc
np.show()
sleep(500)

position = 0
code = ""
nombre = 0
```

```
def clignoter(couleur):
    for x in range(30):
        np[x] = couleur
    np.show()
    sleep(500)
    np.clear()
    sleep(100)

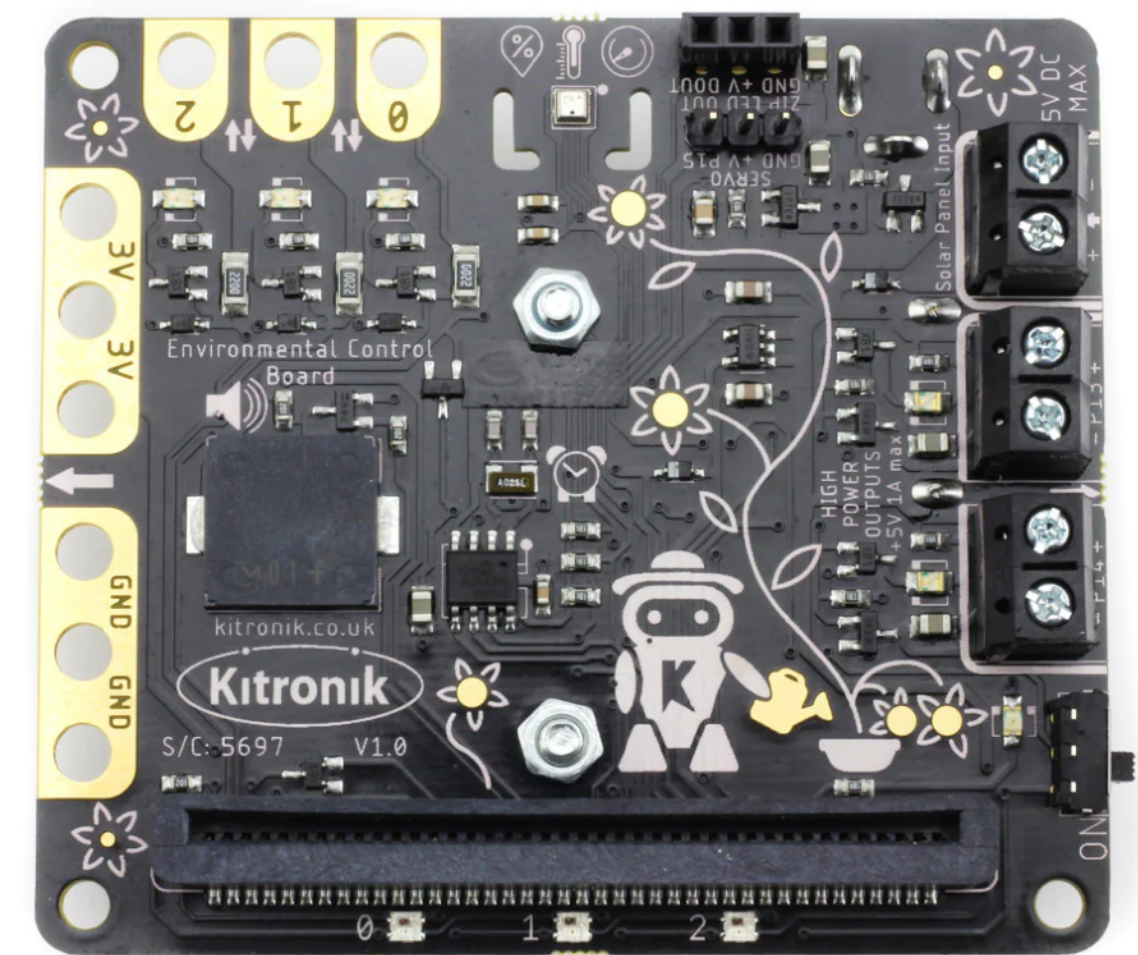
while True:
    info = radio.receive()
    if info:
        code = info[0]
        nombre = info[1]
        if code == "R":
            couleur = rouge
        elif code == "B":
            couleur = bleu
        coup = 0
        while position < 30 and coup < int(nombre):
            np[position] = couleur
            np.show()
            sleep(100)
            position = position + 1
            coup = coup + 1
        if position == 30:
            clignoter(couleur)
            sleep(500)
```

carte pilotant le ruban

La carte kitronik environmental

- La carte kitronik permet (entre autres fonctionnalités) la mesure de plusieurs données environnementales : température, humidité, pression.

- Pour une utilisation avec un éditeur textuel, il faut télécharger les librairies ici : [kitronik-climate](#)



La carte kitronik environnemental

La bibliothèque bme280 peut être téléchargée de façon indépendante :

```
import bme280  
b = bme280.bme280()
```

- `b.temperature()`
- `b.pressure()`
- `b.humidity()`
- `b.altitude()`

La carte kitronik environnemental

Exercice 7

Matériel : 1 carte micro:bit, 1 module kitronik environnemental

Écrire un programme qui permet d'afficher la valeur de la pression lorsqu'on appuie sur le bouton A et de l'humidité lorsqu'on appuie sur le bouton B

On peut également envoyer les données par radio à une autre carte qui fera l'affichage sur écran LCD.

Pression et humidité

La carte kitronik environnemental

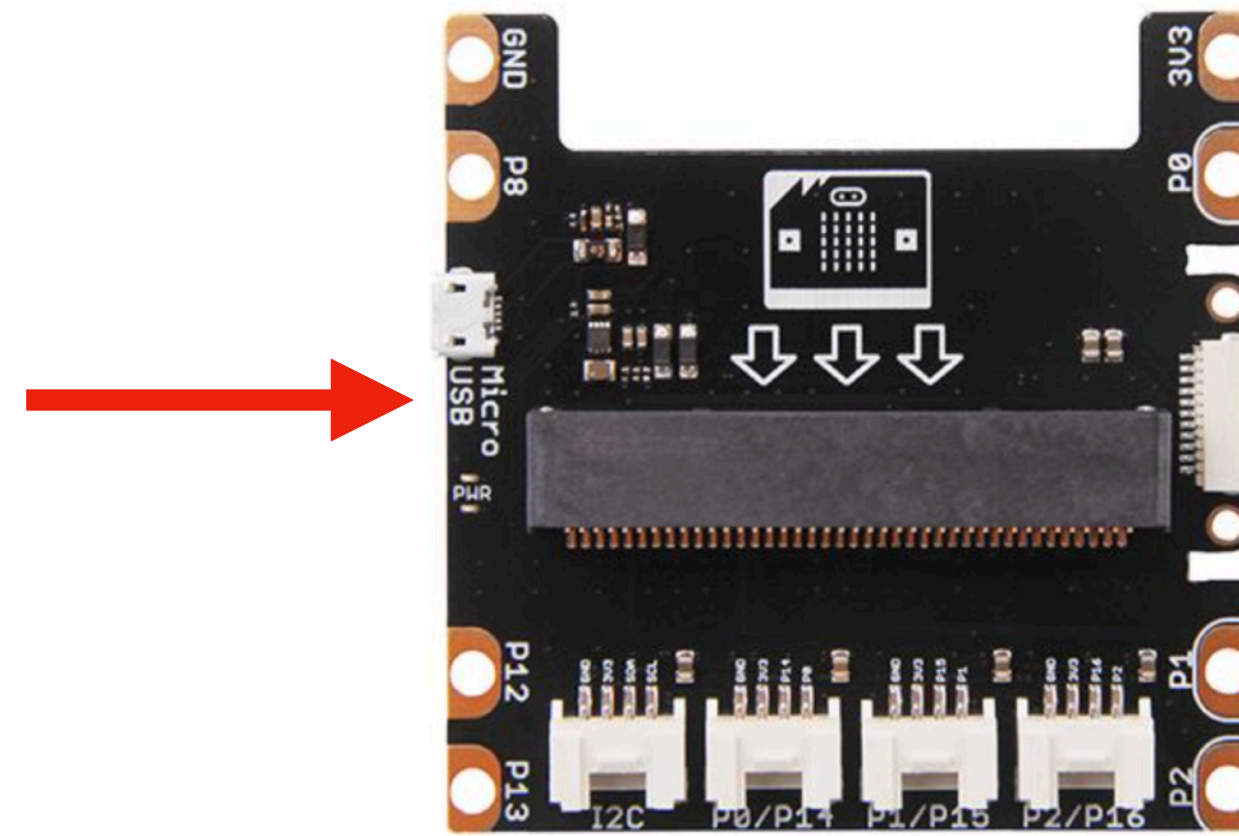
Exercice 7

```
from microbit import *
import bme280
b = bme280.bme280()

while True:
    if button_a.is_pressed():
        pression = b.pressure()
        display.scroll(int(pression))
        sleep(500)
    elif button_b.is_pressed():
        temp = b.temperature()
        display.scroll(int(temp))
        sleep(500)
```

Le kit Grove

Alimentation externe 5V pour l'utilisation de composants "gourmands" en énergie



Le kit Grove

- Ressources pour exploiter le kit
 - Articles et livret sur le site disciplinaire SNT/NSI de Normandie
- Exercice 7bis : reprendre l'exercice 7 en affichant les données sur les 2 lignes de l'écran LCD

Le robot Smart Cutebot



Ressources pour programmer Cutebot

1. Écrire un programme qui permet au robot de suivre une ligne noire
2. Écrire les programmes qui permettent au robot d'être téléguidés à l'aide d'une autre carte micro:bit.

Le concours vidéo

[Cahier des charges 2023/24](#)

[Lauréats 2023](#)

[Lauréats 2022](#)

zoom sur le [labyrinthe](#)

Ressources

Thématiques des projets

Prix thématiques 2023/24

- ✓ Environnement – développement durable
- ✓ Santé
- ✓ Social/ Aide à la personne
- ✓ Jeux/Loisirs
- ✓ Robotique
- ✓ Sécurité
- ✓ Sport
- ✓ Art/design
- ✓ Education tutoriels
- ✓ Expérimentation scientifique/technique

Idées de projet

Quelques suggestions

- ✓ Jeu de chasse au trésor connectée
- ✓ Réalisation de tutoriels vidéos pour utiliser la carte :le robot : https://www.youtube.com/watch?v=79Dmald6o_I
- ✓ Programmation d'un véhicule autonome
- ✓ Jardin et plantes connectés
- ✓ Dispositifs numériques connectés pour un escape game
- ✓ Parcours d'épreuves pour robot Cutebot/Maqueen
- ✓ Mini-golf : <https://sites.google.com/view/imake-it/aktivitäten/minigolf-challenge> (en allemand)
- ✓ Crazy basketball game : https://twitter.com/PinkyPepper_/status/1306606607829266434?s=20
- ✓ Yes We Play Soccer : <https://www.youtube.com/watch?v=NSP1ke0COb8>
- ✓ El encuentro : <https://www.youtube.com/watch?v=hL464MVQdPQ>
- ✓ E-videur : <https://www.youtube.com/watch?v=n5MlW0zg97I>

Quelques conseils

- Trouver une place pour tous
- Permettre des approches différentes
- Laisser une place à la créativité