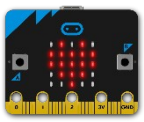
Challenge – Robot Maqueen

Thématique : Architectures matérielles et systèmes d'exploitation **Niveau :** début de 1ere **Type d'activité :** challenge **Point(s) du programme traité(s) :** Périphériques d'entrée et de sortie **Résumé :** L'objectif est de réaliser deux challenges robotiques (le tour de piste le plus rapide et le suivi de ligne le plus rapide). L'activité permet de revoir les bases de la programmation abordée en SNT, de travailler sur la modularisation du code (fonctions) et la programmation embarquée (capteur, SOC et actionneurs). **Prérequis :** Programmation python de base (variables, structures conditionnelles et itératives)	**Durée : 3**h **Matériel / logiciels par binôme :** • Poste informatique équipé de l'IDE Thonny • Un robot Maqueen (lien) + 3 piles AAA • Une carte microbit (lien) + câble USB A - micro-USB B M/M • piste robotique niveau 1 (lien) • piste suivi de ligne niveau 1 (lien) **Documents ressources :** • annexe 1 : exemples de scripts microbit • codes sources de l'annexe 1 • annexe 2 : analyse structurelle microbit • annexe 3 : robot maqueen • correction des exercices préliminaires

1) Introduction

MicroPython est une implémentation du langage de programmation Python 3 optimisé pour fonctionner sur des microcontrôleurs.
MicroPython vise à être aussi compatible que possible avec le Python normal afin de vous permettre de transférer facilement du code du bureau vers un microcontrôleur ou un système intégré.

La carte **micro:bit** est une carte électronique créé pour la promotion de la programmation dans l'éducation. Elle est très simple d'utilisation, mais peut également s'intégrer dans des applications complexes.



Elle est équipée d'un processeur ARM, de deux boutons, d'une matrice 5x5 leds, de 22 broches dont 3 analogiques, d'un accéléromètre, d'un magnétomètre et d'une antenne bluetooth, ce qui en fait une carte parfaite pour les objets connectés.
Le connecteur latéral permet de connecter la carte à des capteurs, des actionneurs ou à d'autres cartes (Arduino, Raspberry, ...).

La programmation peut se faire depuis un navigateur web en Block Editor ou en JavaScript, mais elle peut aussi se faire en langage **MicroPython** depuis un navigateur ou depuis le logiciel **Thonny**.

Exemple de programme :

```
from microbit import *
endless=True
while endless :
    if button_a.is_pressed():
        endless=False
    display.scroll("hello!")
    sleep(2000)
display.show(Image.HAPPY)
```

La première ligne de ce programme importe la bibliothèque de fonctions micro:bit
Les boucles **while** se répètent tant que la condition spécifiée (variable **endless**) est vraie. Le code qui doit être répété est en retrait (indentation).
L'instruction de **sleep()** provoque la pause du micro:bit pendant un nombre défini de millisecondes.
L'instruction de **display.scroll()** fait défiler un message sur la matrice de leds.
Nous sortons de la boucle (variable **endless** passe faux) lorsque l'appui sur le bouton A est détecté (test conditionnel à l'aide la structure **if**).
Une fois sortie de la boucle, un visage joyeux est affiché sur la matrice de leds et le programme est terminé.

2) Rappels de SNT en 2nde

2.1 Exercice 1

Vous pouvez activer et désactiver les leds de la matrice en utilisant leurs coordonnées. Le code suivant définit chacune des leds (pixels) de la rangée supérieure (y = 0) avec une luminosité maximum (9).

```
from microbit import *

for x in range(5):
    display.set_pixel(x,0,9)
    sleep(500)
```

La méthode set_pixel est définie comme suit : ***display.set_pixel (x, y, b)***
Il y a 3 arguments. Les deux premiers arguments spécifient la colonne (x) et la ligne (y). La dernière valeur est la luminosité du pixel et doit être un nombre compris entre 0 et 9.
Adaptez le programme de sorte que la rangée médiane de pixels soit activée au lieu de la rangée supérieure.

2.2 Exercice 2

Modifiez le code de l'exercice précédent pour qu'il éclaire les pixels de la première colonne au lieu de la première ligne.

2.3 Exercice 3

Utilisez 2 boucles, une pour la coordonnée x et une pour la coordonnée y. Allumez chacun des pixels, un à la fois, ligne par ligne.

2.4 Exercice 4

Etudiez le code suivant :

```
from microbit import *
num = 0
while True:
    if button_a.was_pressed():
        display.show(str(num))
        sleep(250)
        display.clear()
    sleep(50)
```

Adaptez le programme de manière à ce que le chiffre affiché s'incrémente de 1 chaque fois que vous appuyez sur le bouton A. S'il est égal (==) à 10, réinitialiser à 0.

2.5 Exercice 5

Ecrivez un programme qui utilise les deux boutons. Faites en sorte que l'utilisateur puisse augmenter le nombre en appuyant sur le bouton B et le diminuer en appuyant sur le bouton A. Assurez-vous que le nombre ne soit jamais inférieur à 0 et jamais supérieur à 9.

2.6 Exercice 6

Ecrivez un programme qui utilise les données de l'accéléromètre. Faites en sorte que l'utilisateur puisse déplacer le pixel allumé vers la gauche en inclinant vers la gauche la carte et vers la droite en inclinant vers la droite.
Utiliser l'instruction **dx = accelerometer.get_x ()** .

- Si dx est supérieur à 100, on avance d'un pixel vers la droite.
- Si dx est inférieur à -100, on avance d'un pixel vers la gauche.

3) Prise en main du robot maqueen

Le robot maqueen est équipé d'une carte micro:bit.
Lire l'annexe et tester les 3 programmes.

3.1 Exercice 1

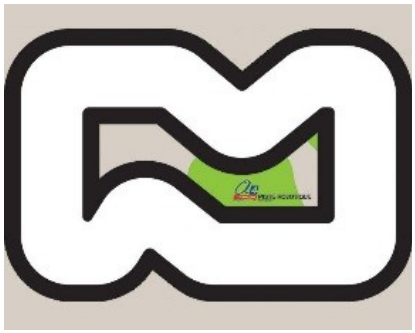
Réaliser une nouvelle fonction nommée **clignote()** qui fait clignoter les 2 LEDs rouges en même temps.

3.2 Exercice 2

Réaliser 4 nouvelles fonctions, nommées **avance(v)**, **recule(v)**, **gauche(v)** et **droite(v)** qui comme leur nom l'indique permettent respectivement d'avancer, reculer, tourner à gauche et tourner à droite. Le paramètre v correspond à la vitesse de rotation des moteurs (0 à 255).

4) Challenge robot maqueen

Challenge n°1 : Réaliser un programme qui permet au robot d'effectuer le parcours 1 sans faire de sortie de route !!



Challenge n°2 : Réaliser un programme qui permet au robot d'effectuer le parcours 2 sans quitter la ligne !!