Thème Informatique embarquée – TP n°2
Capteurs

Capteur de température

✪ Instructions utiles

La carte Micro:bit n'a pas de capteur de température dédié.
Elle utilise la température fournie par la puce de silicium du processeur principal. Comme le processeur chauffe peu en fonctionnement, sa température est une bonne approximation de la température ambiante.

`temperature()`	Renvoie la température de la carte Micro:bit, et donc de la température ambiante, en degrés Celsius.

❖ *Exercice 1 : Thermomètre de secours*

- Dans le logiciel *MU Editor*, écrire un programme qui affiche le défilement de la température en degré Celsius (°C).
- Enregistrer votre fichier en **exo1_temp_sec.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

❖ *Exercice 2 : Celsius versus Fahrenheit*

Le degré Fahrenheit (°F), du nom du physicien allemand, est une unité de température utilisée principalement aux États-Unis.
Les degrés Fahrenheit (°F) sont liés aux degrés Celsius (°C) par la relation :

$$Temp_{(°F)} = 1{,}8 \times Temp_{(°C)} + 32$$

- Dans le logiciel *MU Editor*, écrire un programme qui affiche :
  - le défilement de la température en degré Celsius (°C) si on appuie sur le bouton A.
  - le défilement de la température en degré Fahrenheit (°F) si on appuie sur le bouton B.
- Enregistrer votre fichier en **exo2_temp_cvf.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

Capteur de luminosité

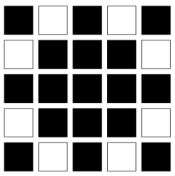
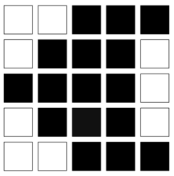
✪ Instructions utiles

La matrice à LEDs de la carte Micro:bit peut servir de capteur rudimentaire pour mesurer la luminosité ambiante.

`display.read_light_level()`	Renvoie un nombre entier compris entre 0 et 255 représentant le niveau de luminosité. Plus le nombre est proche de 0 et plus la luminosité est faible.

❖ *Exercice 3 : Le soleil a rendez-vous avec la lune*

On souhaite afficher un soleil ou une lune en fonction de la luminosité ambiante.
Le soleil et la lune seront représentés par les motifs ci-après.

Soleil	Lune
	

🖱 Compléter et écrire le programme suivant qui devra afficher :
- une image de lune si on baisse la luminosité.
- une image de soleil dans l'autre cas.

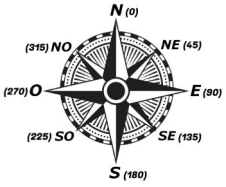
```
from microbit import *

lune = ..........
soleil = ..........
while True:
    if ..... > .....:
        display.show(.....)
        sleep(10)
    else:
        display.show(.....)
        sleep(10)
```

🖱 Dans le logiciel *MU Editor*, flasher le programme sur la carte et observer son bon fonctionnement en recouvrant la carte avec sa main, par exemple.
🖱 Enregistrer votre fichier en **exo3_solune.py** sur votre espace réseau.
👁 Faire vérifier auprès du professeur.

Boussole et capteur d'intensité magnétique

✪ Instructions utiles

Le magnétomètre est capable de fournir une valeur comprise entre 0 et 360 degrés. Il s'agit de l'angle entre le nord magnétique et la position actuelle de la carte Micro:bit.

`compass.calibrate()`	Permet d'étalonner la boussole. Cette commande exécute un petit jeu : au début la carte fait défiler « TILT TO FILL SCREEN » puis il faut incliner la carte pour déplacer le point au centre de la matrice à leds jusqu'à remplir la totalité de la matrice.	
`compass.heading()`	Donne le cap de la boussole sous la forme d'un nombre entier compris entre 0 et 360, représentant l'angle en degrés, dans le sens des aiguille d'une montre, le 0 correspondant au Nord.	

❖ Exercice 4 : Ne pas perdre le nord !

Compléter et écrire le programme ci-après afin qu'il indique le Nord.

```
from microbit import *

compass.calibrate()

while True:
    if (compass.heading()< ..... or compass.heading()> .....) :
        display.show(Image.ARROW_N)
    else:
        display.show(Image.DIAMOND_SMALL)
```

Dans le logiciel *MU Editor*, flasher le programme sur la carte et observer son fonctionnement.
Enregistrer votre fichier en **exo4_perdre_nord.py** sur votre espace réseau.
Faire vérifier auprès du professeur.

✪ Instructions utiles

Le magnétomètre permet de détecter le champ magnétique selon les directions x, y et z lorsqu'un objet magnétique est approché de la carte Micro:bit. Les valeurs s'affichent en nanoTesla (nT).

`compass.calibrate()`	Permet d'étalonner le capteur magnétique. Cette commande exécute un petit jeu : au début la carte fait défiler « TILT TO FILL SCREEN » puis il faut incliner la carte pour déplacer le point au centre de la matrice à leds jusqu'à remplir la totalité de la matrice.
`compass.get_x()`	Détecte le champ magnétique en nT selon l'axe x.
`compass.get_y()`	Détecte le champ magnétique en nT selon l'axe y.
`compass.get_z()`	Détecte le champ magnétique en nT selon l'axe z.

❖ Exercice 5 : Mesurer le champ magnétique

Compléter le programme ci-dessous permettant de mesurer et d'afficher le champ magnétique.

```
from microbit import *
import math


.....

while True:
    val=compass.get_x()**2+compass.get_y()**2+compass.get_z()**2
    magnetisme=int(math.sqrt(val)/.....)
    .....
```

Information : *la sixième ligne du programme permet d'obtenir une valeur du champ magnétique en µT (microTesla).*

- Dans le logiciel *MU Editor*, flasher le programme sur la carte et observer son bon fonctionnement en approchant, au dessus du bouton B de la carte, différents objets comme un petit et un gros aimant, une gomme ou tube de colle, un morceau de papier ou de carton , du papier aluminium ou la pointe d'une paire de ciseaux.
- Enregistrer votre fichier en **exo5_mesure_champ.py** sur votre espace réseau.
- Relever les valeurs obtenues en fonction des différents objets et conclure.
- Faire vérifier auprès du professeur.

❖ Exercice 6 : Détecteur magnétique de proximité

- Dans le logiciel *MU Editor*, écrire un programme qui affiche un visage souriant lorsqu'on approche un aimant de la carte Micro:bit (au dessus du bouton B) et un visage triste lorsque l'on s'en éloigne.
- Enregistrer votre fichier en **exo6_detect_prox.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

Capteur de mouvement (accéléromètre)

✪ Instructions utiles

L'accéléromètre mesure l'accélération de la carter Micro:bit et permet donc de détecter quand la carte est en mouvement.
Ce capteur peut aussi détecter d'autres actions, par exemple, quand la carte est secouée, inclinée ou encore lorsqu'elle tombe.
Des accéléromètres sont présents dans les smartphones ou les manettes de jeux afin de pouvoir afficher une image dans le sens ou bien de changer de direction dans un jeu.
L'accéléromètre mesure le mouvement selon trois axes :

- X : l'inclinaison de gauche à droite
- Y : l'inclinaison d'avant en arrière
- Z : le mouvement haut et bas

`accelerometer.get_x()`	Permet de détecter un mouvement de gauche à droite en renvoyant un nombre compris entre -1023 et 1023, le 0 correspondant à la position d'« équilibre ».
`accelerometer.get_y()`	Permet de détecter un mouvement en avant et en arrière en renvoyant un nombre compris entre -1023 et 1023, le 0 correspondant à la position d'« équilibre ».
`accelerometer.get_z()`	Permet de détecter un mouvement de haut en bas en renvoyant un nombre compris entre -1023 et 1023, le 0 correspondant à la position d'« équilibre ».
`accelerometer.is_gesture("shake")`	Renvoie **True** ou **False** si la carte est actuellement secouée ou non.
`accelerometer.was_gesture("shake")`	Renvoie True ou False si la carte a été secouée ou non.

❖ Exercice 7 : En avant, en arrière, à gauche, à droite !

```
from microbit import *

while True:
     abscisse=accelerometer.get_x()
     if abscisse>500:
          display.show(Image.ARROW_E)
          sleep(200)
     if abscisse<-500:
          display.show(Image.ARROW_W)
          sleep(200)
     if (abscisse>-500 and abscisse<500):
          display.show(Image.DIAMOND_SMALL)
```

- Dans le logiciel *MU Editor*, écrire le programme ci-dessus, flasher le programme sur la carte et observer son fonctionnement.
- Améliorer ce programme pour qu'il affiche une flèche vers le haut si on incline la carte en avant et une flèche vers le bas si on incline la carte vers l'arrière.
- Enregistrer votre fichier en **exo7_hbgd.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

❖ Exercice 8 : Podomètre

```
from microbit import *

choc=0
display.show(choc)

while True:
     if accelerometer.was_gesture("shake"):
          choc=choc+1
          display.show(choc)
```

- Dans le logiciel *MU Editor*, écrire le programme ci-dessus, flasher le programme sur la carte et observer son fonctionnement.
- Modifiez le programme ci-dessus afin qu'il réalise un podomètre pour lequel :
  - la matrice leds doit afficher en permanence le nombre de pas effectués.
  - le bouton A devra remettre à zéro le nombre de pas.
  - le bouton B affichera le défilement de la distance parcourue en mètre sachant qu'un pas est équivalent à une distance de 60 cm.
- Enregistrer votre fichier en **exo8_podo.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

Pour aller plus loin…

❖ *Défi 1 : Dé numérique*

- Écrire un programme qui indique qui simule un dé à 6 faces en affichant un face au hasard sur la matrice leds de la carte lorsque celle-ci est secouée.
- Enregistrer votre fichier en **defi1_denum.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.

❖ *Défi 2 : Intrusion*

- Écrire un programme qui permet de compter combien de fois une porte a été ouverte en utilisant le capteur de champ magnétique.
- Enregistrer votre fichier en **defi2_intrusion.py** sur votre espace réseau.
- Faire vérifier auprès du professeur.