Unité 7 : micro:bit avec Python — Compétence 2 : Boutons et Accélérations

Dans cette leçon, vous allez apprendre à utiliser les boutons et l'accéléromètre de la carte BBC micro:bit, puis écrire un programme pour créer un dé et collecter les valeurs dans une liste à transférer, afin de créer un diagramme de données statistiques.

Cette leçon comporte deux parties :

Partie 1 : Etude des instructions sur les boutons et utilisation de l'accéléromètre.

Partie 2 : Utilisation d'un bouton pour générer des données.

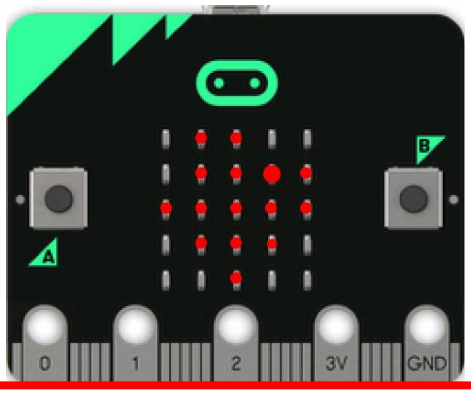
Objectifs :

- Lire et traiter les boutons A et B sur la carte micro:bit.
- Observer la différence entre **.was** et **.is**.
- Transférer des données du script Python vers la calculatrice.
- Étudier les données collectées à partir du micro:bit.
- Utiliser l'accéléromètre pour contrôler l'affichage.

Conseil de l'enseignant : Comme dans la Compétence 1, cette leçon est conçue de l'intérieur vers l'extérieur. Le code n'est pas introduit séquentiellement, mais développé pour se concentrer d'abord sur les fonctionnalités du bouton ou du déplacement spatial de la carte micro:bit, puis terminer par établir une connexion entre les listes Python et les listes TI-83 Premium CE.

1. La carte micro:bit possède deux boutons, étiquetés A et B, de chaque côté de la carte. Le module Python micro:bit a deux méthodes similaires pour utiliser les boutons. Vous allez d'abord tester les instructions, puis écrire un programme qui vous permet de collecter des données et de les analyser ailleurs dans la calculatrice TI-83 Premium CE.

   Il y a aussi une puce accéléromètre/boussole à 3 axes à l'arrière du micro:bit. Une convention (orientations des axes) est nécessaire afin d'interpréter le mouvement et l'orientation de la carte micro:bit.



Conseil de l'enseignant : Sur la version micro:bit 2, il y a aussi un bouton « tactile » au-dessus de l'écran (l'ovale doré). Dans le module, ce bouton est appelé « Logo Touch » et utilise l'instruction **'is_touched( )'**. Ce bouton tactile n'est pas abordé dans ces leçons, mais il est facile à intégrer dans votre projet en fonction de ce que vous apprenez dans cette leçon. Notez la différence entre le comportement de **.is_** et **.was** ci-dessous...

2. **Partie 1 : Utilisation des boutons.**

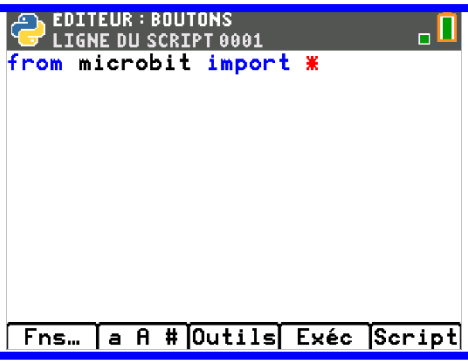
   Démarrer un nouveau programme Python dans un nouveau document.

   Appuyer sur la touche **[prgm]** et sélectionner **2 : Python App** puis **[F3] Nouv**.

   Nous appelons le programme **BOUTONS.**

   Dans **2nde catalog**, sélectionner l'instruction d'importation **from SCRIPT import** et complétez manuellement l'instruction.

   from microbit import *



3. Importer également la bibliothèque **TI_System**.
   Ajouter la boucle **while** :

   while not escape( ):

   depuis

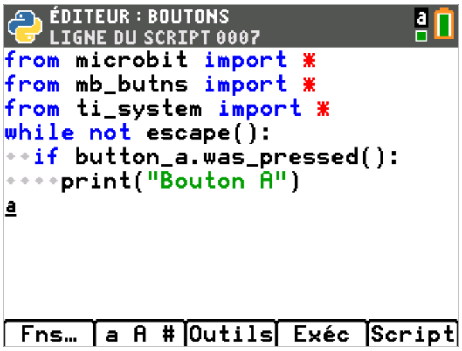
   [Fns] > modul > 4 : ti_system > 5 : while not escape( )

   La plupart de vos programmes micro:bit *seront conçus de cette façon.*

4. Pour tester le bouton A, charger la librairie relative aux boutons puis ajouter la condition :

   ♦♦**if button_a.was_pressed( ):**
   ♦♦♦♦**print("Bouton A")**

   if *est mis en retrait pour faire partie de la* boucle **while** *et* **print( )** est mis en retrait encore plus pour faire partie *du* bloc **if.** *N'oubliez pas qu'une indentation correcte est très importante en Python. Une mise en retrait incorrecte peut provoquer des erreurs de syntaxe ou une exécution incorrecte de votre code. Notez les symboles losange gris clair (♦♦)* qui *indiquent l'espacement de mise en retrait (indentation).*



   if se trouve dans **[Fns] > Ctl** .
   La *condition* **button_a.was_pressed( )** se trouve à

   [Fns] < modul> 9 : boutons > 2 : .was_pressed( )

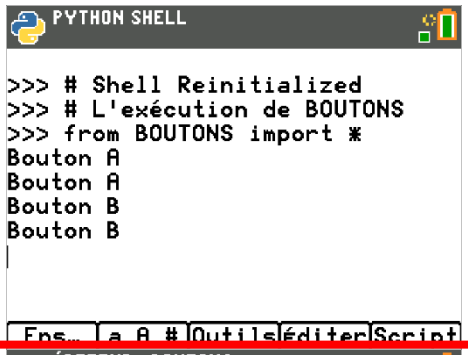
   print( ) se trouve dans **[Fns] < E/S**.

   Taper le texte « Bouton A » sur le côté de la fonction **print( )**.

   Note : **is_pressed( )** *sera également utilisé plus tard.*

5. Vous êtes prêt à tester le programme. Appuyer sur **[F4] Exec** pour exécuter le programme. On dirait qu'il ne se passe rien. Appuyez et relâchez le bouton A sur la carte micro:bit. Vous verrez « Bouton A » apparaître sur l'écran de la calculatrice. Chaque fois que vous appuyez sur le bouton, le texte apparaîtra comme dans cette image.

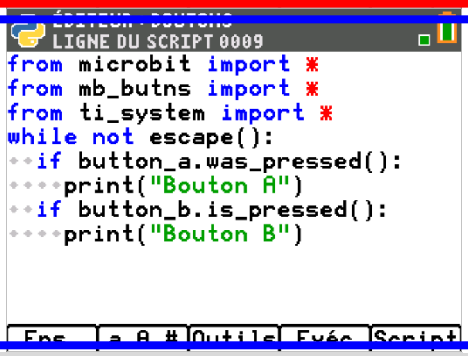
   Appuyez sur **[annul]** et revenez à l'éditeur Python.



6. Ajoutez une autre instruction **if** pour vérifier le fonctionnement du bouton B à l'aide de la condition **button_b.is pressed( )**. Notez que « IS » est différent de « WAS ».
   Vous verrez bientôt en quoi ils diffèrent.

   ♦♦**if button_b.is_pressed() :**
   ♦♦♦♦**print("Bouton B")**

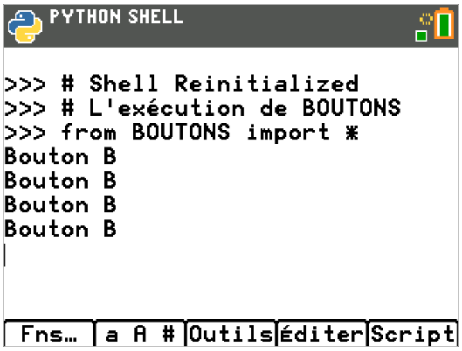
   Astuce : encore une fois, faites attention aux indentations !



Conseil de l'enseignant : Les deux fonctions se comportent différemment. Il existe des circonstances de programmation où le choix entre les deux comportements sera important.

7. Exécutez à nouveau le programme (appuyez sur **[F4] Exec**). Essayez les deux boutons A et B.
   Appuyez sur chaque bouton *et* **appuyez sans relâcher** chaque bouton. Vous verrez « Bouton B » affiché à plusieurs reprises tant que le bouton B est maintenu enfoncé, mais pas « Bouton A ». Il y a une différence entre **.was_pressed( )** (qui nécessite une libération du bouton pour être réinitialisé) et **.is_pressed( )** qui vérifie simplement si le bouton est enfoncé au moment même où l'instruction est traitée.

   Remarque : si vous appuyez sur le bouton B rapidement, le programme peut ne pas afficher le bouton B car le bouton n'est pas enfoncé au moment même où l'instruction **if** *est en cours de traitement.*



Conseil à l'enseignant :
.was_pressed( ) nécessite une action complète de « appuyer-relâcher » pour détecter les pressions individuelles (clics).

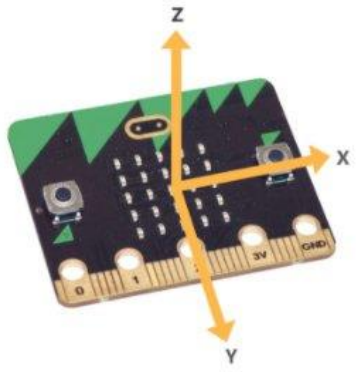
.was_pressed( ) détecte un événement appuyer-relâcher qui peut se produire même lorsque le bouton a été cliqué à un autre moment dans l'exécution de la boucle. Le bouton doit être relâché pour qu'un autre clic se produise. Le micro:bit « se souvient » que le bouton a été enfoncé.

.is_pressed( ) est comme un événement « est enfoncé ? ». Certains langages de programmation pilotés par les événements ont une fonction similaire telle que « mouse_down ». **.is_pressed( )** ne produit **True** que si le bouton **est enfoncé** au moment exact **où l'instruction est traitée.**

*Dans le reste de cette leçon, le code du bouton B est ignoré***.**

8. **Utilisation de l'accéléromètre.** Le micro:bit dispose d'un accéléromètre électronique qui peut mesurer l'intensité des accélérations dans trois directions différentes (un accéléromètre 3 axes ou 3D).
Cette leçon explore les possibilités de l'accéléromètre et montre comment l'utiliser dans la programmation de votre TI-83 Premium CE avec la carte micro:bit.

La carte



Charger la librairie relative aux capteurs. Cette librairie se trouve à :

[Fns] < modul > 8 : micro :bit > 3 : capteurs

9. Obtener les valeurs de l'accélération du mouvement à partir du micro:bit et les stocker dans les variables **ax, ay, az:**

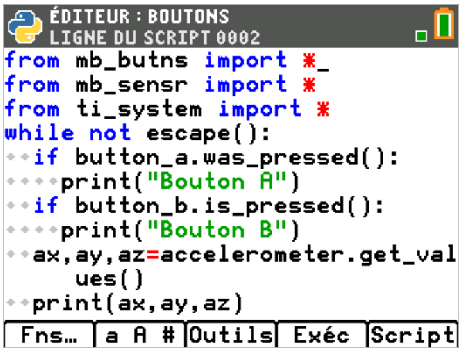
♦♦**ax,ay,az = accelerometer.get_values( )**

Astuce : encore une fois, faites attention aux indentations !

Taper ♦♦**ax,ay,az** puis

Trouver **accelerometer.get_values( )** à

[Fns] < modul > 0 : capteurs > 4 : var,var,var=.get values( )



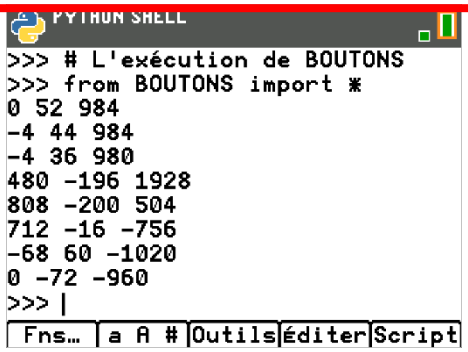
Puis afficher les valeurs des accélérations selon les axes *Ox, Oy et Oz* :

♦♦**print(ax,ay,az)**

Rappel : **print( )** est dans **[Fns] < E/S**

Notez que ces deux instructions sont indentées pour faire partie de la boucle **while**, *mais pas* de *l'instruction* **if** *au-dessus. N'oubliez pas que l'indentation de chaque ligne détermine la signification de ce que réalise le script. Alors SOYEZ PRUDENT !*

10. Exécuter le programme et regarder l'affichage de la calculatrice pour les différentes valeurs lorsque que vous déplacez la carte micro:bit dans les airs. Tournez la carte micro:bit, tenez-la sur chaque bord, secouez-la. Une partie de l'exécution du programme est affichée dans cette image.



L'accéléromètre du BBC micro :bit 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.

Pour chaque axe, la convention de repérage permet de comprendre le signe obtenu lors d'une mesure. Un nombre positif ou négatif qui indique une mesure en milli-g. Lorsque la lecture est de 0, on est « aligné » selon cet axe.

Nous allons réaliser un « niveau à bulle » très simple qui utilise l'instruction **accelerometer.get_x( )** pour mesurer l'alignement de la carte selon l'axe X. L'algorithme est le suivant

```
Lire l'accélération selon l'axe X
    Si lecture >20 :
        Afficher "Droite"
    Ou Si lecture < -20 :
        Afficher "Gauche"
    Sinon :
        Afficher "—"
```

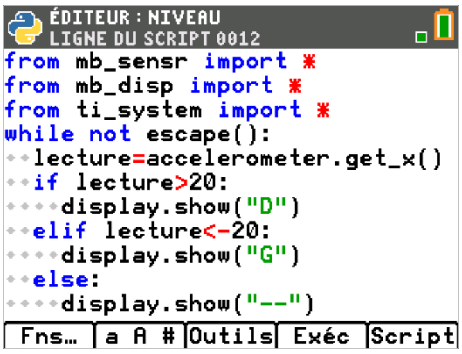
11. Créer un nouveau script appelé « niveau » et importer les modules nécessaires (**micro:bit**, **ti_system, mb_sensor**, **mb_disp**).

Astuce : Dupliquer le script précèdent à l'aide de

[Script] ↓ selection du script [Gérer] 1 : Dupliquer

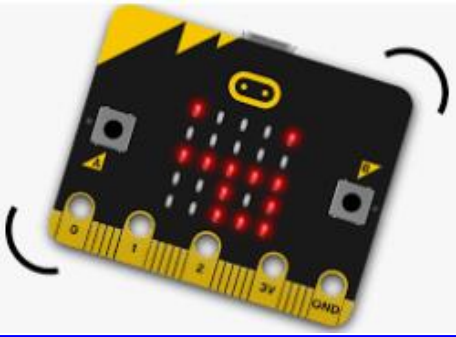
Effacer ensuite les lignes que vous ne souhaitez pas conserver, ou les commenter.

Mettre la carte BBC micro:bit sur une surface plane et exécuter le script. Vérifier ainsi la planéité d'une surface.



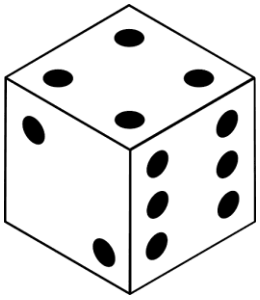
Si l'appareil est tenu horizontalement il devrait afficher -- en revanche, si on l'incline vers la gauche ou vers la droite il devrait afficher G ou D respectivement.

12. Les boutons et les déplacements spatiaux sont deux façons d'obtenir une entrée sur la carte micro:bit et de produire des résultats sur l'écran de la calculatrice TI-83 Premium CE, l'écran micro:bit... ou les deux.
La partie suivante de cette leçon crée une activité qui produit des données à l'aide de la carte micro:bit pour un apprentissage plus approfondi avec la TI-83 Premium CE.



13. **Partie 2 :** Jetons un dé (un cube numéroté de 1 à 6 sur ses faces). Lorsque le **bouton A est enfoncé**, un entier aléatoire de 1 à 6 est assigné à une variable. Vous pouvez utiliser le bouton A, le bouton B ou un ~~geste~~ déplacement de votre choix.
Afficher la valeur sur la carte micro:bit uniquement. Essayez-vous même avant de regarder l'étape suivante. Nous utiliserons le programme actuel et ajouterons du code pour simuler le lancer de dé.

Pouvez-vous déterminer quel numéro se trouve au bas du *dé illustré ci-contre ?*



Conseil à l'enseignant :

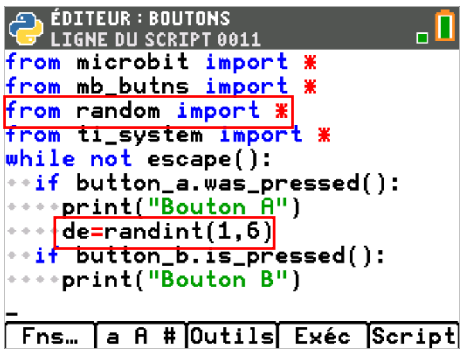
Réponse : la valeur sur la face inférieure est 3. Les faces opposées d'un dé s'additionnent toujours à 7.

14. Reprendre le script « boutons ». Ajoutez les deux instruction mises en valeur selon les indications de la capture ci-contre :

From random import * et **randint( )** se trouvent tous deux sur
[Fns] < Modul > 2 : Random

Le reste de l'instruction ♦♦♦♦**de = randint(1, 6)** est tapé manuellement et est soigneusement indenté, car il fait partie de la condition **If button_a.was_pressed( )**.

Encore une fois, faites attention à l'indentation.

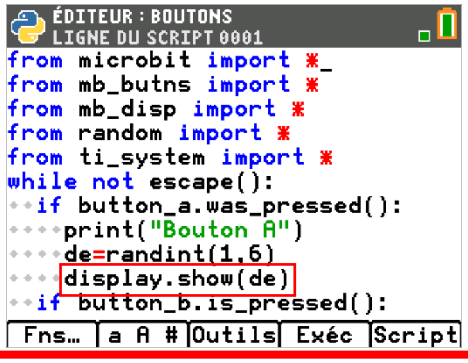


15. Une fois que la valeur de la variable a été établie, ajouter l'instruction :

display.show (de)

pour afficher la valeur sur la matrice de l'écran de la carte micro:bit, sans oublier de charger le module d'affichage **mb_disp**.

Exécuter le programme à nouveau. Lorsque vous appuyez sur le bouton A, vous voyez « Bouton A » sur l'écran de la calculatrice et le numéro sur l'affichage de la carte micro:bit change... mais pas à chaque fois ! Parfois, le nombre aléatoire sélectionné est le même que le précédent. Les appuis successifs sur le bouton A sont des « événements indépendants ».

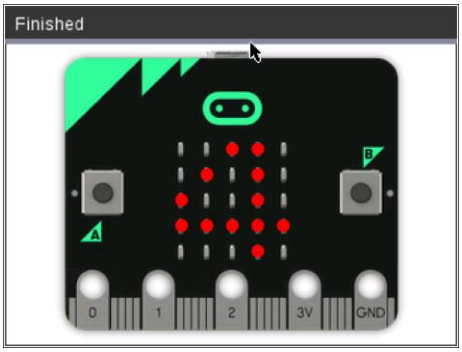


16. **Collecte de données :** Lancer le dé avec juste une pression sur un bouton est intéressant, mais pour une étude plus approfondie, il serait utile de stocker toutes ces valeurs dans des listes afin que vous puissiez interpréter les données : Un nombre a-t-il une occurrence plus fréquente que les autres ? Quel est le nombre moyen ? Et ainsi de suite…

Ajouter les instructions au programme afin de :

- Créer une liste vide.
- Ajouter (**.append**) la valeur d'un lancer à la liste.
- Stocker la liste de Python vers le système TI-83 Premium CE pour l'analyse.

Chacune de ces trois tâches se traduit par des instructions placées à des endroits particuliers dans le programme. Essayez-le vous-même avant de passer à l'étape suivante.



17. L'affectation de liste vide est à mettre au début du programme :

lancers = [ ]

Les crochets sont sur le clavier et dans la table de caractères **[a A #]**.

Lorsqu'un lancer est effectué, il est ajouté à la liste avec :

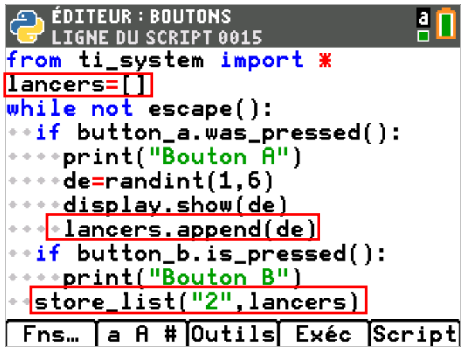
lancers.append(de)

La méthode **.append()** se trouve dans **[Fns] > List > 6.append(x)**

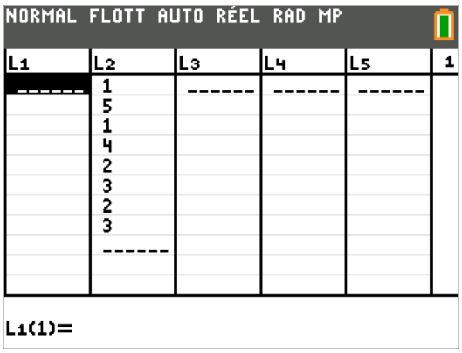
A la fin du programme, la liste finale est stockée dans une liste de la TI-83 Premium CE :

store_list("2", lancers)

La fonction **store_list** est à **[Fns] < Modul > 4 : ti_system**.

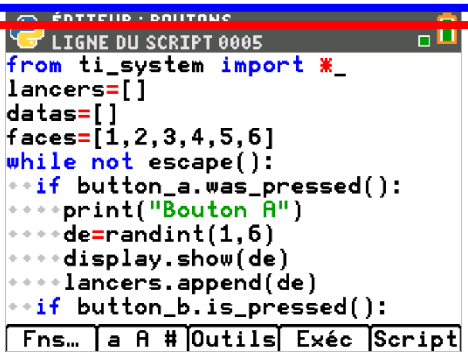


18. Lorsque vous exécutez le programme, appuyez plusieurs fois sur le bouton A, puis appuyez sur **[annul]** pour mettre fin au programme. Votre programme Python a une liste nommée « lancers » et maintenant les données de cette liste ont été exportées dans la liste L_2 de la calculatrice.



Mais ces données ne sont pas classées. Nous allons le faire en utilisant une boucle fermée (**For**) et l'instruction **.count(n° face)** effectuera le comptage des occurrences correspondant aux numéros des faces (1 à 6). Les données de ce comptage seront sauvegardées dans une liste **data( )** à initialiser aux côtés des autres (lancers( ) et faces=[1,2,3,4,5,6])

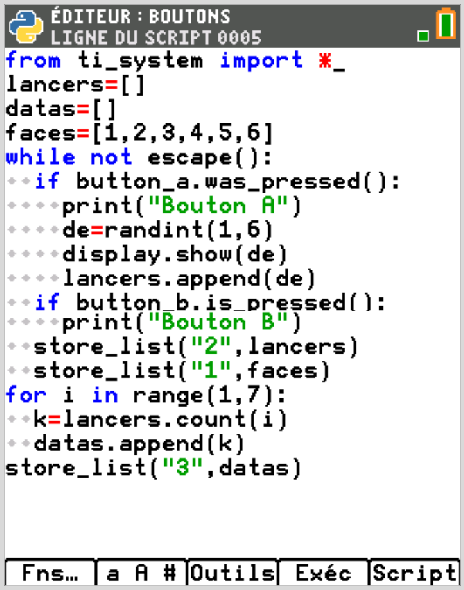
La liste faces sera exportée en L_1 et la liste data en L_3
Exécuter de nouveau votre script afin d'obtenir une cinquantaine de lancers, puis effectuer la représentation graphique des données sous la forme d'un diagramme en bâtons.

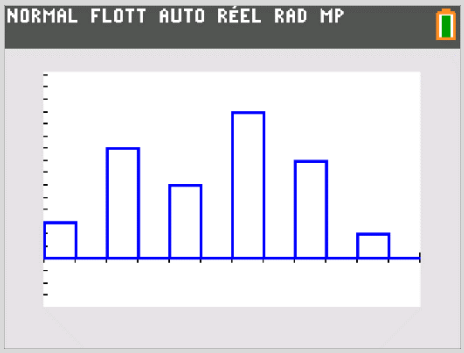


Remarque : le bouton B et votre geste n'ont aucun effet sur la matrice. Essayez de modifier votre code pour afficher un lancer du dé uniquement lorsque *vous « secouez » la carte micro:bit.*

N'oubliez pas d'enregistrer votre document !

Conseil à l'enseignant :
Le bouton B pourrait être utilisé comme un bouton de « réinitialisation », effaçant les données et recommençant avec une liste vide.





19. Extensions possibles :

- Modifier votre script afin que la face de chaque lancer soit affichée.
- Utiliser l'accéléromètre afin de déclencher la réalisation d'un lancer.