الوحدة الثالثة:
البرمجة بواسطة المايكروبت (Micro:bit)

أهلاً بك ستتعلم في هذه الوحدة كيفية برمجة متحكم دقيق باستخدام لغة نصية. ستتعرف على أداة مايكروسوفت ميك كود (MakeCode) لبرنامج المايكروبت (Micro:bit) وستتعلم كيفية البرمجة باستخدام لغة بايثون.

بالإضافة إلى ذلك، ستتعلم كيفية إنشاء أكواد أكثر تعقيدًا باستخدام المتغيرات والدوال والحلقات واتخاذ القرارات من أجل إكمال المهام المعقدة.

أهداف التعلم

ستتعلم بنهاية هذه الوحدة:

- ماهية المايكروبت ومكوناته.
- استخدام مايكروسوفت ميك كود.
- أنواع المتغيرات والعمل عليها.
- التعامل مع الأرقام والإحداثيات بلغة بايثون.
- التكرارات في مايكروبت بلغة بايثون وكيفية استخدامها.
- اتخاذ القرارات في مايكروبت بلغة بايثون.

الأدوات

- مايكروسوفت ميك كود للمايكروبت (Microsoft MakeCode for Micro:bit)

رابط الدرس الرقمي

www.ien.edu.sa

الدرس الأول:

مقدمة إلى المايكروبت (Micro:bit)

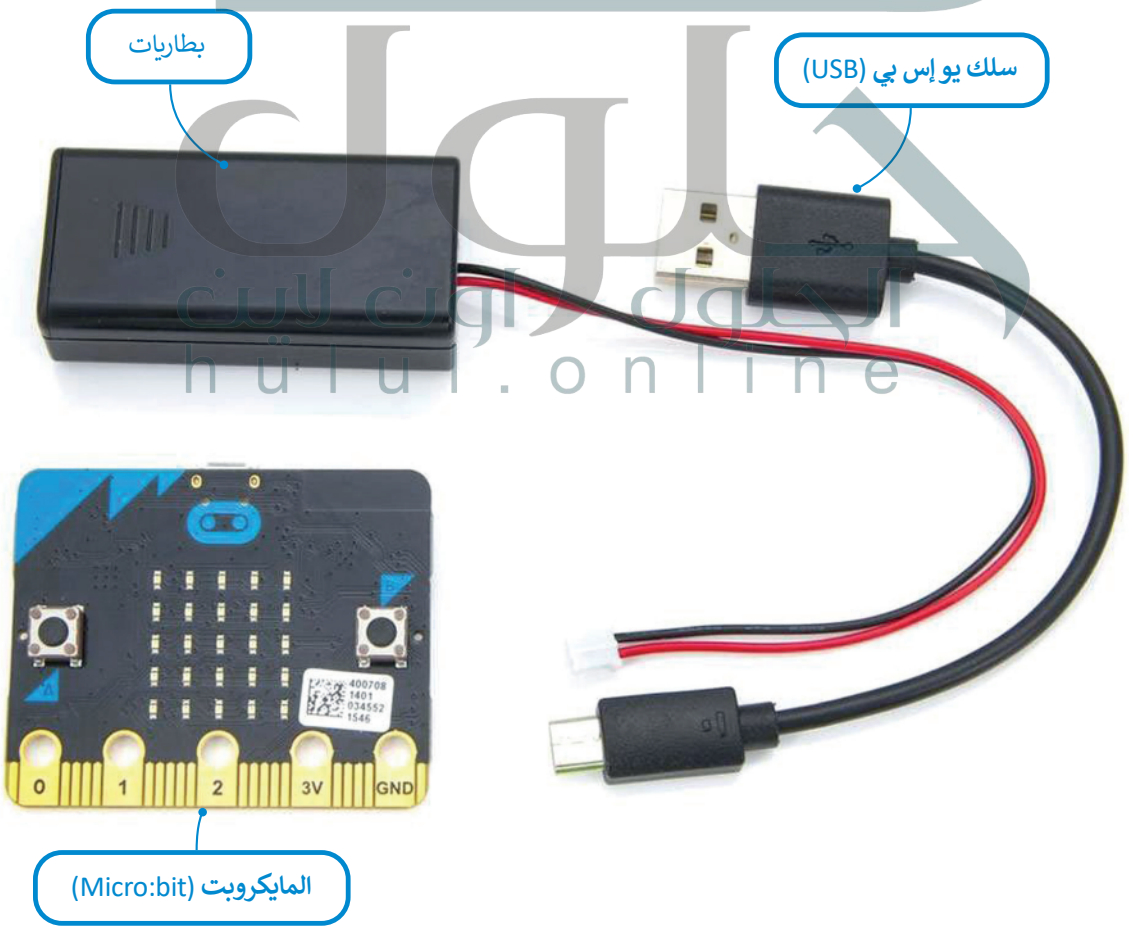
تمارس البرمجة دورًا مهمًّا في التقدم التقني وترتبط بجميع المجالات في الحياة، كما تسهم في تطوير مهارات التفكير المختلفة. ستتعرف في هذا الدرس على إحدى التقنيات المخصّصة لتطوير المهارات البرمجيّة بشكل سلس وسريع، وهي المايكروبت (Micro:bit) وستستخدم لغة البرمجة بايثون (Python) لكتابة برامجك في مايكروسوفت ميك كود (Microsoft Makecode)، وستتعلم أيضًا كيفية التعامل مع المتغيرات في البرمجة.

لتتعرف على المايكروبت (Micro:bit)

المتحكمات الدقيقة هي دوائر إلكترونية متكاملة تحتوي على معالج دقيق إلى جانب الذاكرة، وتدعم مختلف الأجهزة الطرفية القابلة للبرمجة والمستخدمة للإدخال والإخراج وتتحكم في وظائف الجهاز أو النظام الإلكتروني. تعدّ المتحكمات الدقيقة حاسوبًا صغيرًا مبسّطًا على شكل رقاقة صغيرة يمكن أن يعمل بأدنى حد من المكونات الخارجية نظرًا لأنظمته الفرعية العديدة المدمجة.

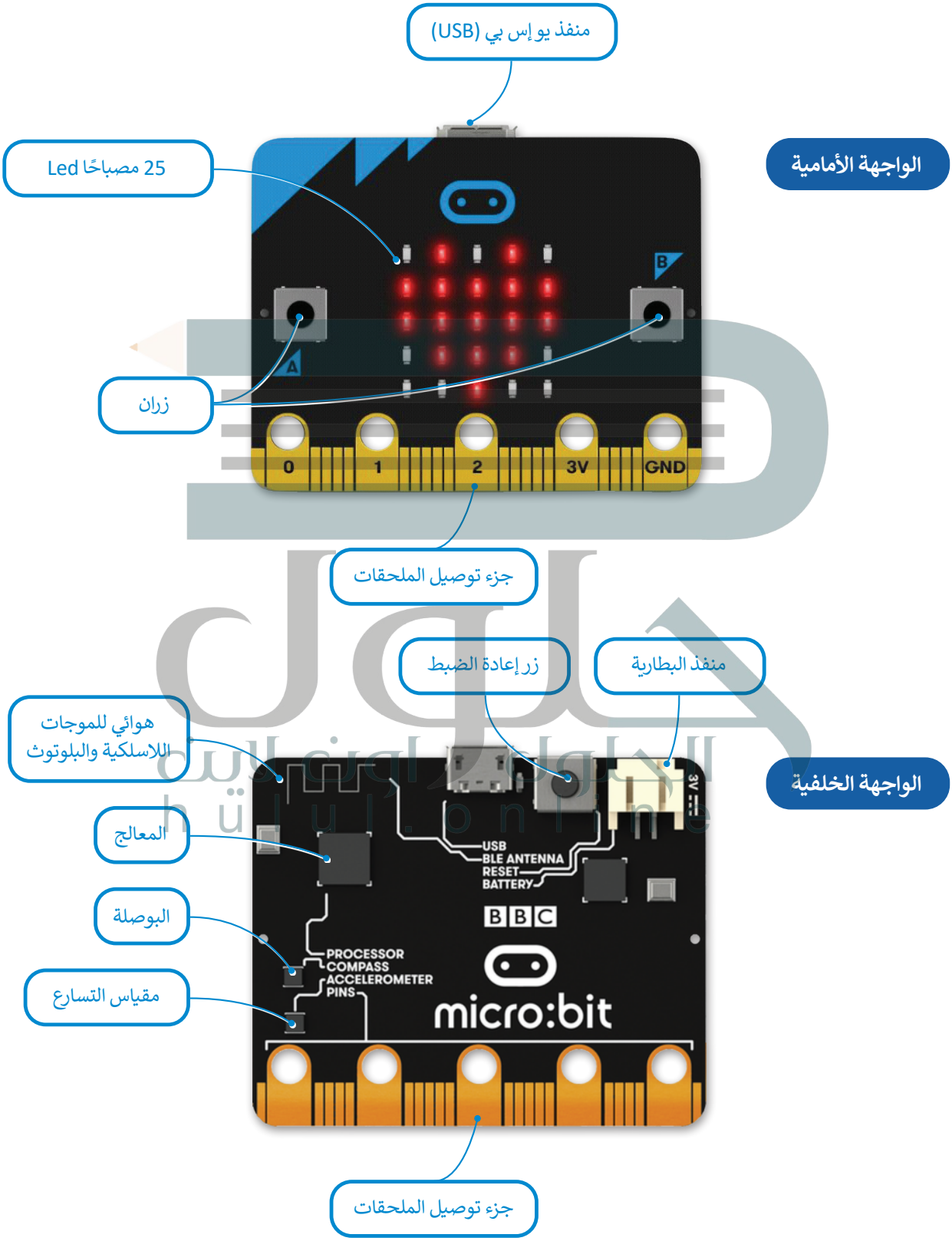
يمكن العثور على المتحكمات الدقيقة في مجموعة كبيرة من الأنظمة والأجهزة، وتستخدم على نطاق واسع في جميع الأنظمة المدمجة مثل الساعات الذكية، والكاميرات الرقمية للبوابات الذكية، والأجهزة الكهربائية، وجميع أنواع المركبات ذاتية القيادة، كما يمكن أيضًا استخدامها في بناء الروبوتات.

يُعدُّ **المايكروبت** (Micro:bit) حاسب صغير الحجم تم إنشاؤه من قِبل هيئة الإذاعة والتلفزيون BBC. يمكنك استخدامه لإنشاء مشاريع رائعة، وذلك من خلال توظيف مهاراتك البرمجية.



مكونات المايكروبت

يتكون المايكروبت من واجهة أمامية وواجهة خلفية يوجد عليهما مجموعة من المكونات موضحة فيما يلي:

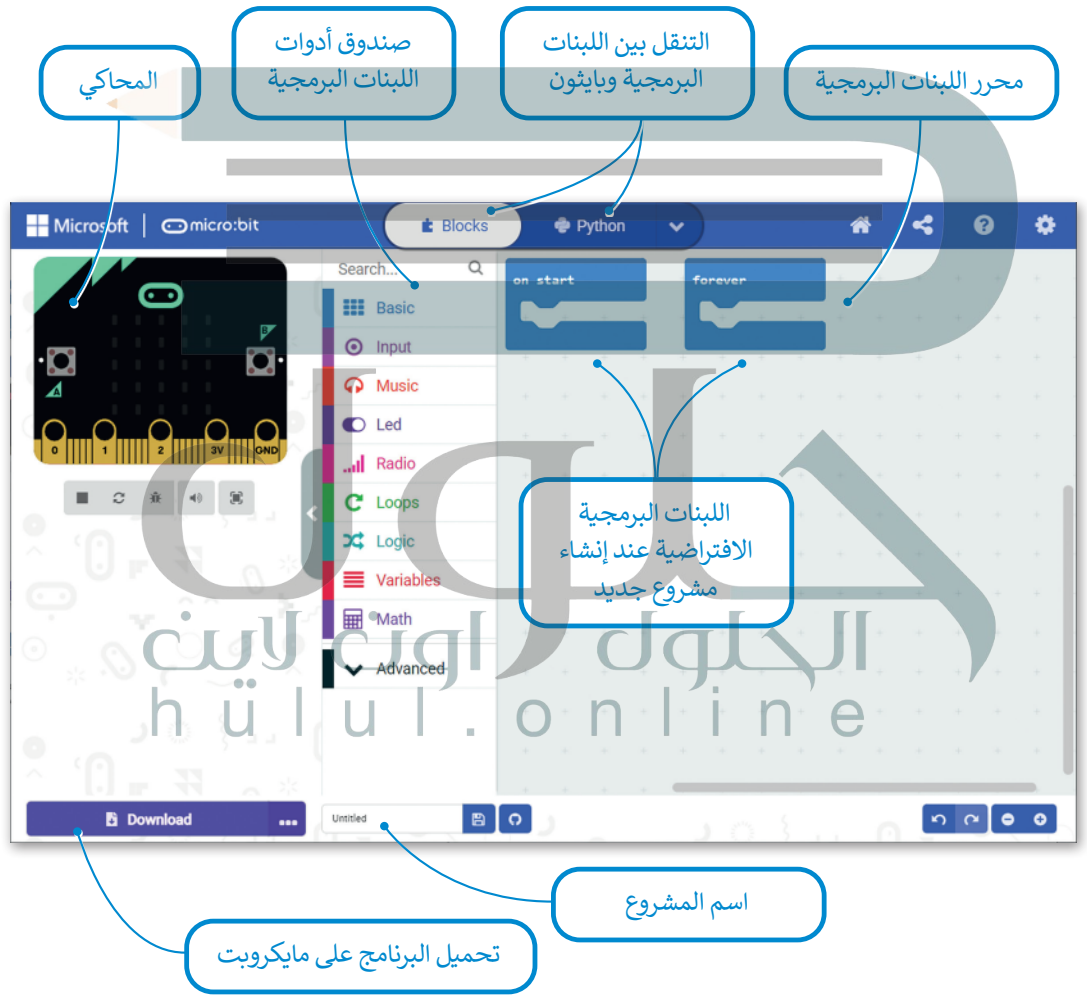


مايكروسوفت ميك كود (Microsoft MakeCode)

يمكنك استخدام لغات برمجة مختلفة لبرمجة المايكروبت، كلغة برمجة **مايكروسوفت ميك كود** (Microsoft MakeCode) القائمة على اللبنات البرمجية، أو لغة **بايثون** (Python) للبرمجة النصية. ستستخدم في هذه الوحدة مايكروسوفت ميك كود.

يتوافر محرر ميك كود عبر الإنترنت، وللبدء بإنشاء مشاريعك عليك زيارة موقع الويب: https://makecode.microbit.org/#editor.

وفيما يلي توضيح لمكونات الواجهة الرئيسة لمحرر ميك كود:



لمحة تاريخية

تم ابتكار لغة **بايثون** (Python) بواسطة **جيودو فان روسوم** (Guido van Rossum)، وكان إصدارها الأول في العام 1991 م. وهي لغة برمجة عالية المستوى مفتوحة المصدر وسهلة التعلم.

إنشاء برنامج في مايكروبت

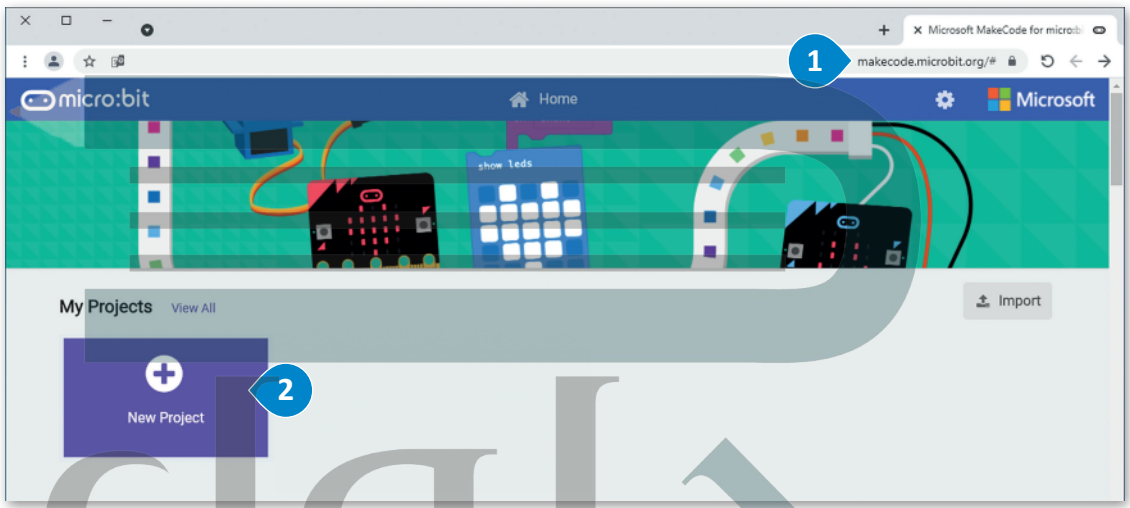
حان الوقت لإنشاء مشروعك الأول في مايكروبت، وسيكون على شكل برنامج صغير تستخدم فيه مصابيح Led الموجودة في مايكروبت لإضاءة رسالة ترحيبية. إذا لم يتوافر لديك جهاز مايكروبت حقيقي لاختبار برامجنا، يمكنك إنشاء محاكاة بديلة عبر الإنترنت.

إنشاء برنامج جديد:

> اكتب **makecode.microbit.org** 1

> اضغط على **New Project** (مشروع جديد) لإنشاء مشروع جديد. 2

> الآن أصبح مشروعك جاهزًا للبدء بالبرمجة. 3



إضافة لبنات إلى برنامجك

لقد أنشأت مشروعًا في مايكروبت وستقوم الآن بإضافة اللبنات المناسبة لجعل برنامجك يعرض رسالة ترحيب.

لإنشاء برنامج باستخدام اللبنات البرمجية:

- اضغط على فئة لبنات **Basic** (أساسي). 1
- اسحب وأفلت لبنة **show string "Hello!"** (إظهار السلسلة "Hello!") داخل لبنة **on start** (بداية). 2
- اسحب وأفلت لبنة **show icon** (إظهار الرمز) داخل لبنة **forever** (للأبد). 3
- اضغط على زر التشغيل وسيعرض المحاكي رسالتك. 4
- اضغط على زر التوقف لإيقاف المحاكي. 5

تعتبر لغات **بايثون** (Python) **وفيجوال بيسك** (Visual Basic) **وجافا سكريبت** (JavaScript) لغات برمجة عالية المستوى. تستخدم كلمات وحروف ورموز عادية من اللغة. تتضمن لغة البرمجة عالية المستوى كلمات يجب تعلمها، وكذلك قواعد لبناء الجمل البرمجية يجب اتباعها، كما في اللغات التي يتحدثها البشر.

توفر بيئة التطوير المتكاملة (IDE) وظائف أكثر تعقيدًا لمساعدة المطور على كتابة التعليمات البرمجية المعقدة بسهولة أكبر.

الانتقال من اللبنات البرمجية إلى لغة بايثون

تُعدُّ لغة بايثون واحدة من مئات لغات البرمجة الموجودة حاليًا، وتُستخدم في هذه اللغة كلمات من اللغة الإنجليزية وتراكيب خاصة لوصف التعليمات في الحاسب وهي لغة برمجة نصية عالمية، وتستخدم للأغراض العامة، حيث يمكنك العثور عليها في مجموعة متنوعة من التطبيقات المختلفة. ستساعدك بيئة مايكروسوفت ميك كود في كتابة برنامجك الأول وبرمجة المايكروبت من خلال سحب وإفلات اللبنات في محرر لغة البرمجة. إبدأ بإنشاء برنامجك وذلك بإضافة اللبنات البرمجية.

شاهد كيف يمكنك الانتقال إلى لغة بايثون من خلال مايكروسوفت ميك كود:

لتحويل البرنامج إلى لغة بايثون:

> اضغط على القائمة المنسدلة الخاصة بلغات البرمجة. 1

> حدد لغة **Python** (بايثون). 2

> سيظهر البرنامج بلغة بايثون. 3

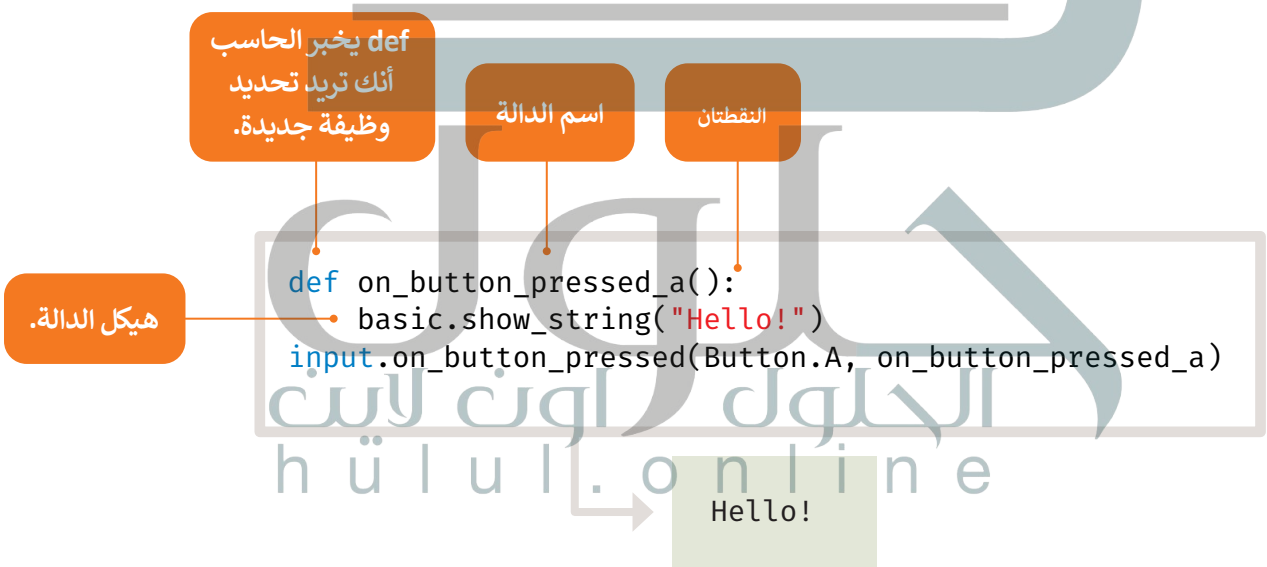
الدوال في لغة البايثون

في البرمجة، تكون الدالة عبارة عن جزء من التعليمات البرمجية التي تُستخدم لمساعدتك في مهمة أو حدث متكرر ومحدد ، مثل الضغط على زر. الميزة الرئيسة هي إمكانية استدعائها بشكل متكرر في البرنامج الرئيسي.

محتوى جملة الدالة في بايثون:

- > يستخدم الجزء الأول من الدالة كلمة **def** ويحتوي على تعريف الدالة.
- > الجزء الثاني هو اسم الدالة.
- > يوجد في نهاية رأس الدالة نقطتان.
- > يأتي بعد ذلك هيكل الدالة، ويجب وضع مسافة بادئة لها.

فيما يلي دالة تطبع رسالة "**Hello!** " ("مرحبا!") عند الضغط على زر المايكروبت.



في هذه الوحدة سوف تستخدم الدوال التالية:

الدالة	الوصف
on_forever( )	تنفذ الدالة جزء من الكود بشكل لا نهائي في الخلفية.
on_button_pressed_a( )	تنفذ الدالة جزءًا من الكود عندما يتم الضغط على زر المايكروبت وتحريره مرة أخرى.
on_gesture_shake( )	تنفذ الدالة جزءًا من الكود عندما تهز المايكروبت.

يمكنك أيضًا إنشاء التعليمات البرمجية باستخدام اللبنات البرمجية وتحويلها للغة بايثون أو العكس. لتغيّر الآن البرنامج بلغة بايثون وترى نتيجة هذا التغيير على اللبنات البرمجية.

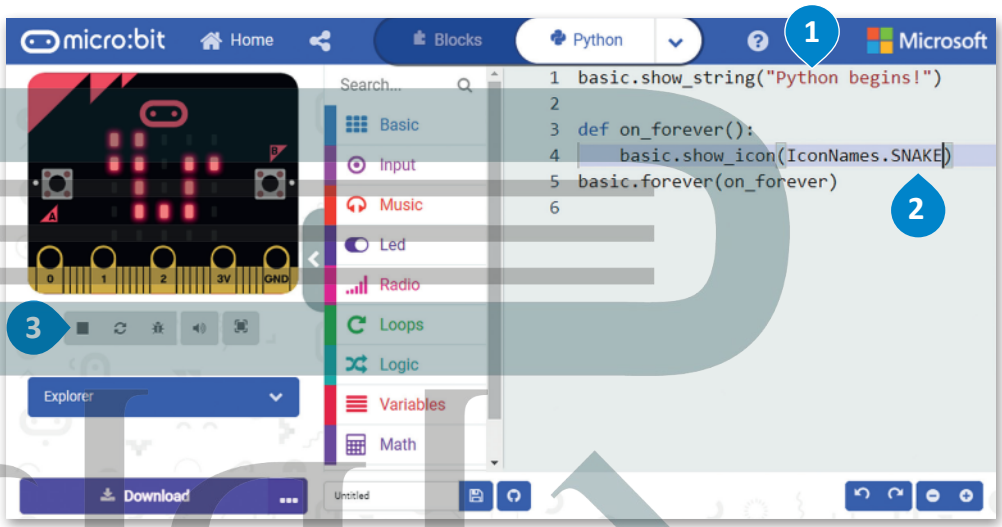
للبرمجة باستخدام بايثون:

> اضغط ضغطة مزدوجة على الأمر **show_string** (إظهار السلسلة) واستبدل كلمة **"Hello!"** ("مرحبا!") بعبارة **"Python begins!"** ("بايثون يبدأ!"). 1

> اضغط ضغطة مزدوجة على الأمر **show_icon** (إظهار الرمز) واستبدل كلمة **HEART** (قلب) بعبارة **SNAKE** (ثعبان). 2

> اضغط على زر التشغيل لبدء المحاكاة. 3

في حال ظهور رسالة خطأ، يجب أن تتحقق من كتابة البرنامج بشكل صحيح. تأكد أولًا من عدم نسيان أي أقواس أو علامات اقتباس، وتحقق أيضًا من عدم وجود أي أخطاء إملائية.



للانتقال إلى اللبنات البرمجية

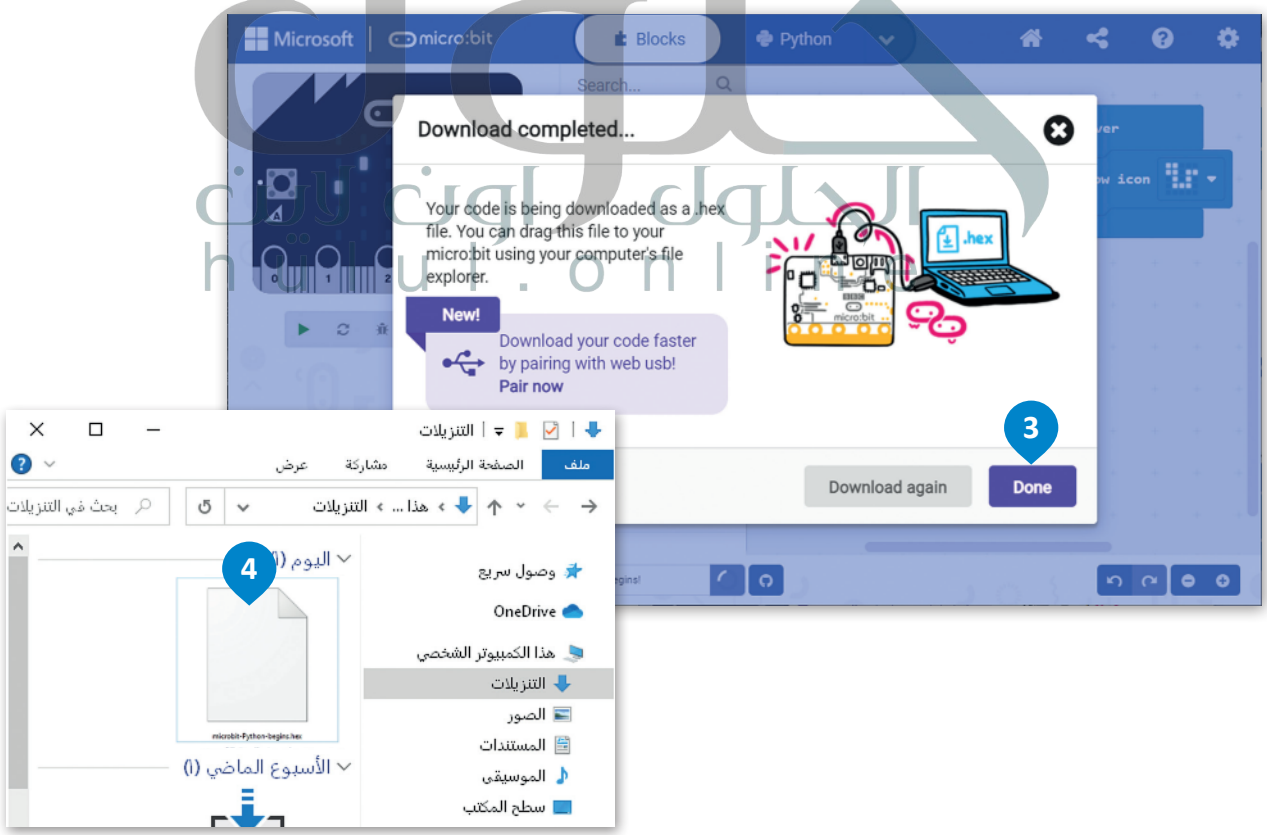
حفظ البرنامج

لحفظ البرنامج على الحاسب:

> اكتب اسمًا لبرنامجك. 1

> اضغط على أيقونة حفظ. 2

> اضغط على **Done** (تم) من النافذة المنبثقة 3، يتم حفظ البرنامج في مجلد **التنزيلات**. 4



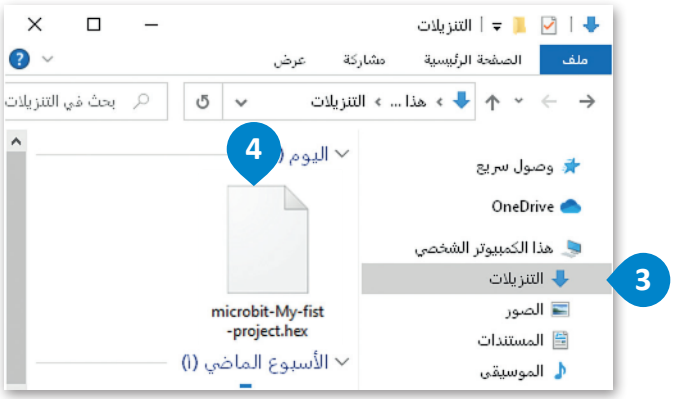
تنزيل البرنامج على جهاز المايكروبت عبر سلك يو إس بي

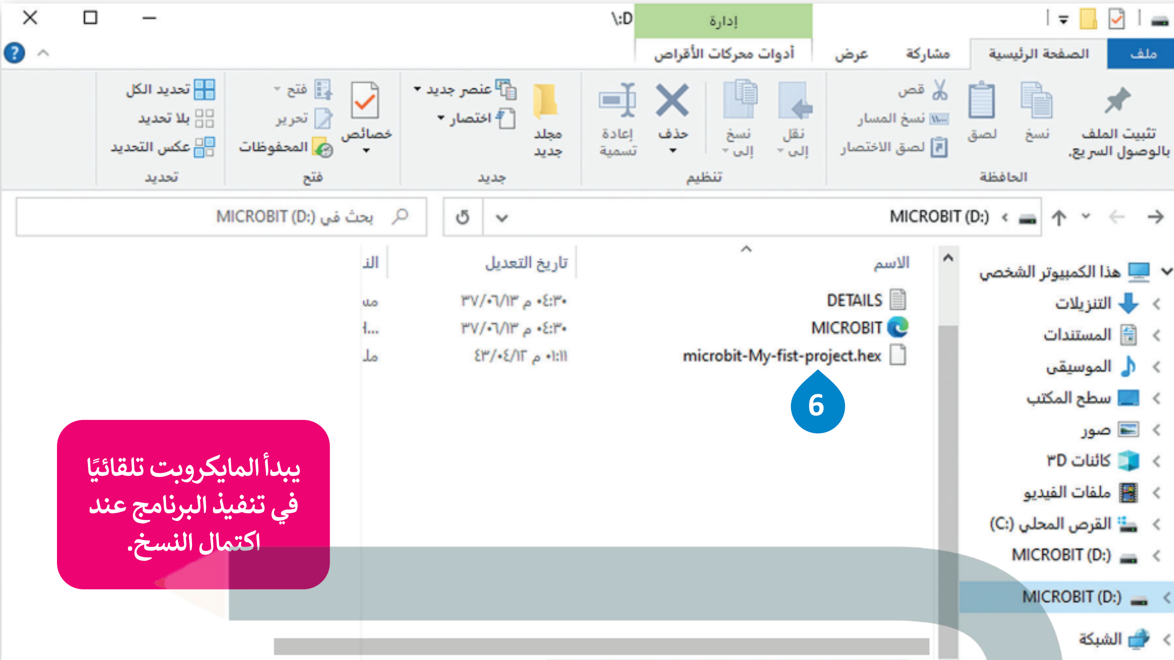
لتنزيل البرنامج على جهاز المايكروبت، عليك أولًا توصيل المايكروبت بجهاز الحاسب الخاص بك باستخدام سلك يو إس بي. بعد ذلك سيظهر كمحرك أقراص يو إس بي محمول.

لتنزيل البرنامج على المايكروبت :

- اكتب اسمًا لبرنامجك. 1
- اضغط على **Download** (تنزيل). 2
- افتح مجلد **التنزيلات** (Download) 3 وانسخ الملف بامتداد **hex**. الخاص برنامجك. 4
- افتح محرك اقراص **MICROBIT** (مايكروبت) 5 والصق الملف بامتداد **hex**. 6

سيضيء المصباح الموجود على الجزء الخلفي من المايكروبت لإظهار أن البرنامج يقوم بالنسخ. عندما يتوقف عن الوميض، سيعمل البرنامج على المايكروبت الخاص بك.



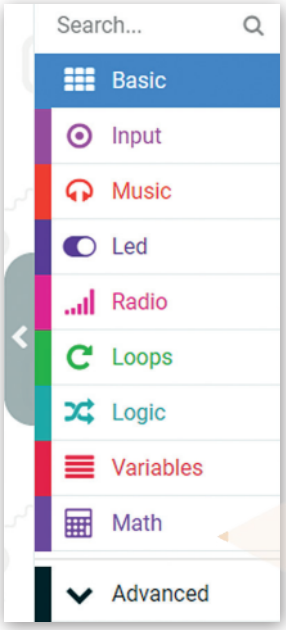


حذف اللبنات

لحذف لَبِنة أو مجموعة من اللبنات، عليك سحبها وإفلاتها مرة أخرى في مربع أدوات **اللبنات** (Blocks).

صندوق أدوات الأوامر

يتم تنظيم دوال مايكروبت في نطاقات بأسماء مطابقة لأسماء التبويبات، وبنفس طريقة تنظيم اللبنات البرمجية ضمن فئات (تبويبات). يُعد استدعاء إحدى دوال بايثون المضمنة في مايكروبت أسهل الطرق لبدء استخدام مايكروبت في بايثون.



لإضافة أمر في محرر اللغة يتعين عليك فقط سحبه وإفلاته.

أزل كافة الأوامر السابقة من المحرر وابدأ بإضافة الأوامر النصية لإنشاء البرنامج بلغة بايثون.

لإنشاء برنامج بايثون:

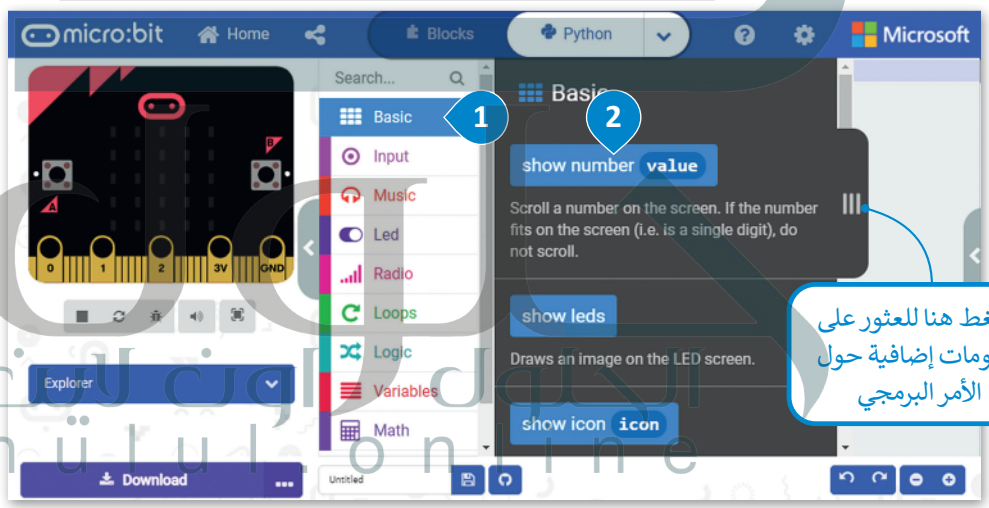
> اضغط على فئة **Basic** (أساسي) الأساسية. 1

> اسحب وأفلت أمر **show number** (إظهار الرقم) في المحرر. 2

> اكتب الرقم الذي تريد إظهاره. 3

> اضغط على زر التشغيل لبدء المحاكي الذي سيعرض الرسالة السابقة على الشاشة. 4

عند سحبك لدالة بايثون وإفلاتها في المحرر، سيتم تنفيذها عند الضغط على أيقونة التشغيل بصورة افتراضية.



اضغط هنا للعثور على معلومات إضافية حول الأمر البرمجي

يُعبِّر الجزء الأول قبل النقطة عن فئة الأوامر التي ينتمي إليها الأمر البرمجي

احفظ عملك دائمًا.

أمثلة برمجية

أزرار مايكروبت

حان الوقت لترى كيف يمكنك استخدام الأزرار الموجودة في المايكروبت. ستنشئ مشروعًا جديدًا ينتج منه إظهار الحرف **A** على شاشة المايكروبت عند الضغط على زر **A**، وإظهار الحرف **B** عند الضغط على زر **B**.

ابدأ بإنشاء مشروع جديد.

لبرمجة زر A:

> اضغط على فئة أوامر **input** (الإدخال). 1

> اسحب وأفلت أمر **run code on button pressed** (عندما يكون زر run code مضغوط). 2

> من فئة أوامر **Basic** (أساسي)، اسحب وأفلت أمر **show leds** (إظهار المصابيح). 3

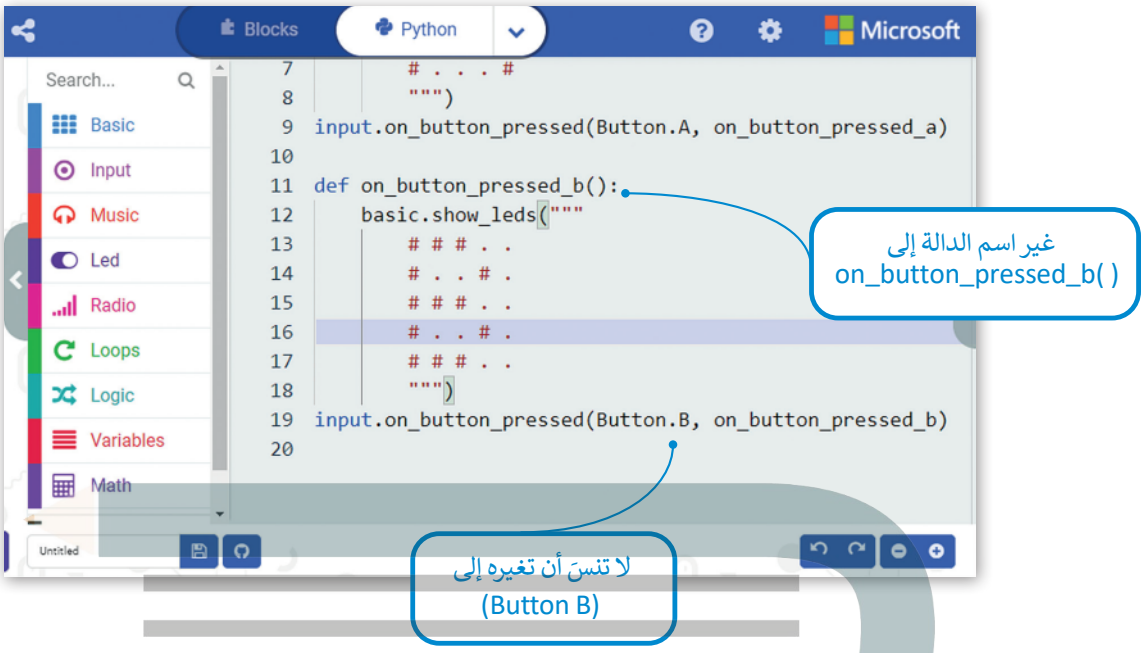
> داخل الأمر إظهار المصابيح، أنشئ الحرف **A** في مصابيح **Led**، **#** لإضاءة المصباح، و **.** لعدم إضاءته. 4

> اضغط على زر التشغيل لبدء البرنامج. 5

> اضغط على الزر **A** في المحاكي لمعاينة النتيجة. 6

عند البرمجة بلغة بايثون يمكنك استخدام رمز **#** لتشغيل مصابيح **Led** أو إيقاف تشغيله.

كرر نفس الخطوات لبرمجة الزر **B**.



هناك خيار آخر من خلال الضغط على زري **A** و **B** في نفس الوقت.

دالة الاهتزاز (Shake)

يستخدم مايكروبت مقياس التسارع الخاص به لاكتشاف أي تغيرات في الحركة. ستنشئ برنامجًا يعرض رقمًا مختلفًا على شاشة المايكروبت كل مرة يهتز بها المايكروبت.

استخدام دالة الاهتزاز (Shake):

> من فئة **Input** (الإدخال)، اسحب وأفلت دالة **run code on Gesture.Shake** (run code عند Gesture.Shake). 1

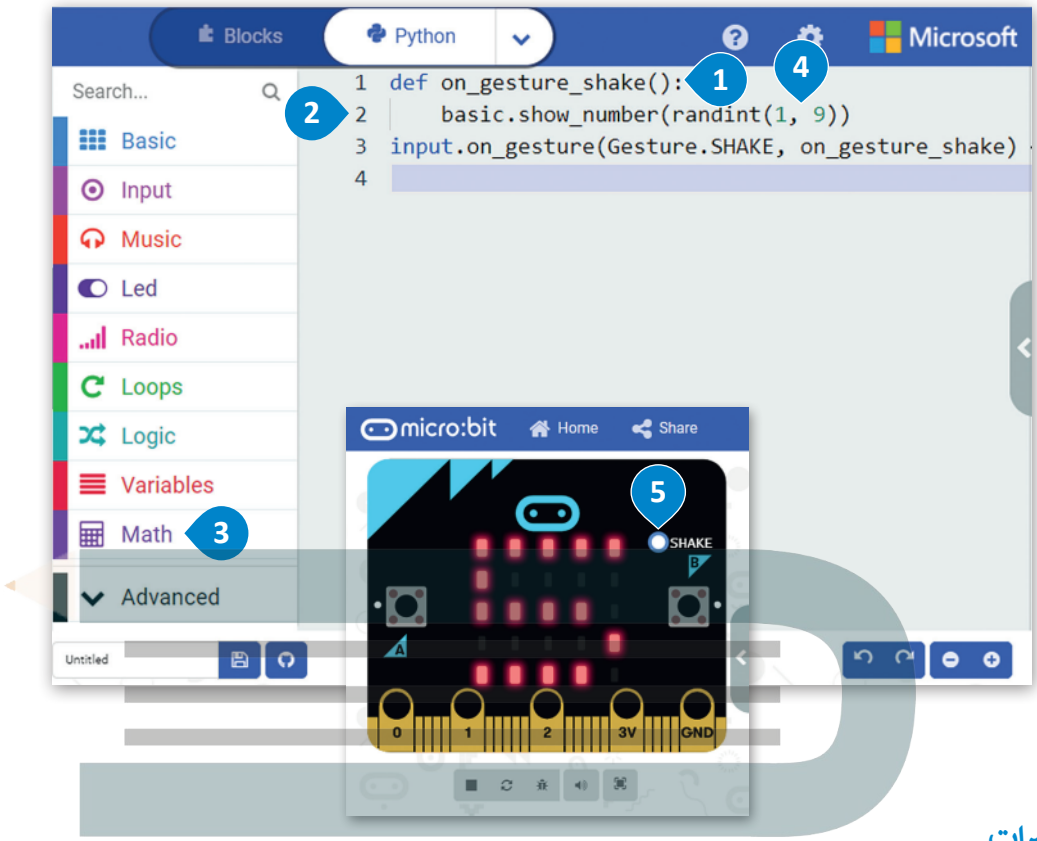
> من فئة **Basic** (أساسي)، اسحب أمر **show number** (إظهار الرقم) وأفلته. 2

> اضغط على فئة أوامر **Math** (رياضيات). 3

> حدد أمر **randint**، ضعه داخل الأمر **show number** (إظهار الرقم) واضبط نطاق القيم إلى **(1,9)**. 4

> اضغط على زر **SHAKE** (اهتزاز) في المحاكي لاختبار برنامجك. 5

يقوم الأمر randint بوضع رقم عشوائي داخل النطاق المحدد (بين أدنى وأعلى قيمة في النطاق).



المتغيرات

ترتبط المتغيرات بمواقع تخزين البيانات، ويتم منح كل متغير اسمًا رمزيًا يسمح باستخدامه بشكل مستقل عن المعلومات التي يمثلها. يمكن أن تتغير قيمة المتغير أثناء تنفيذ البرنامج، ويمكن أن تمثل المتغيرات أنواعًا مختلفة من البيانات. الفئتان الرئيستان للمتغيرات هما: الأرقام والنصوص. تدعم لغة بايثون نوعين من الأرقام، وهما: الأعداد الصحيحة والأعداد العشرية. وكما تعرفت سابقًا في سكراتش فإن المتغيرات النصية تسمى **سلاسل نصية** (Strings).

يمكن أن يكون للمتغير اسمٌ مختصر مثل (x أو y)، أو اسمٌ وصفيٌ مثل (total_volume، CarModel، age).

الأعداد (القيم العددية)

```
MyAge=12
level=3
score=1200
```

النصوص (السلاسل النصية)

```
MyName="Salman"
EmailAddress="salmansa.bl@outlook.com"
color="Green"
```

لا يمكن استخدام بعض الكلمات لتسمية المتغيرات لكونها كلمات خاصة أو مفتاحية مستخدمة بواسطة لغة البرمجة، ويُطلق على هذه الكلمات اسم الكلمات المحجوزة مثل:

def	and
return	not
while	True
else	False
global	None
if	import

الإعلان عن المتغيرات

الإعلان عن المتغير هو عملية تعيين قيمة ومعرِّف (اسم فريد) للمتغير. عليك استخدام علامة المساواة (=) للإعلان عن متغير.

يجب الانتباه إلى أن استخدام علامة المساواة (=) في البرمجة يختلف عن استخدامها في الرياضيات والعمليات الحسابية، فعلى سبيل المثال يشير استخدام علامة المساواة بهذا الشكل (**MyAge = 12**) إلى أنك تريد تمرير القيمة **12** كرقم ليتم تعيينها إلى المتغير المسمى **MyAge**. يمكنك أيضًا القيام بعمليات حسابية على الجانب الأيمن من علامة المساواة ثم إسناد النتيجة إلى المتغير الموجود على الجانب الأيسر.

لتستعرض مثالًا على ذلك.

لكل متغير في البرمجة اسم وقيمة فريدة.

يمكنك أثناء برمجتك بلغة بايثون كتابة الأوامر التي تتذكرها، ولا يُعدُّ ضروريًا اختيارها من فئات الأوامر مرة أخرى.

لتعيين قيمة متغير عددي:

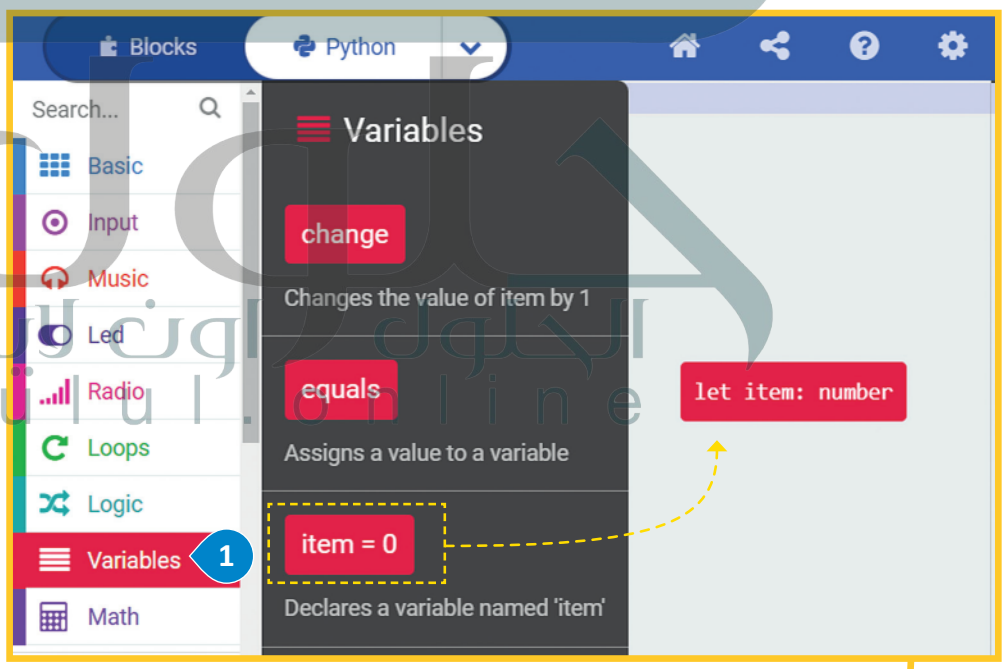
> اضغط على فئة أوامر **Variables** (متغيرات). 1

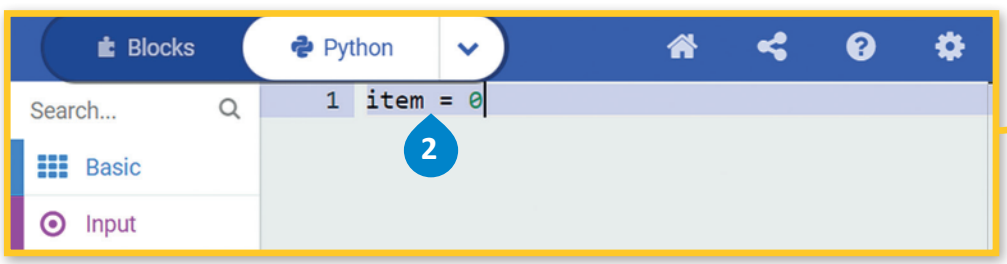
> اسحب وأفلت أمر **item = 0** (العنصر = 0) داخل محرر التعليمات البرمجية. 2

> اكتب واضبط اسم المتغير ليكون **MyAge =12** (عمري =12) . 3

> من فئة أوامر **Basic** (أساسي) اسحب وأفلت أمر **show number** (إظهار الرقم). 4

> اكتب اسم المتغير داخل الأقواس. 5





المتغيرات النصية

لا يقتصر استخدام المتغيرات على تخزين الأرقام فقط، بل يمكنك استخدامها لتخزين النصوص أيضًا. تسمى المتغيرات التي تخزن النصوص متغيرات نصية، ولتعيين نص إلى متغير كل ما عليك هو وضع النص داخل علامات الاقتباس.

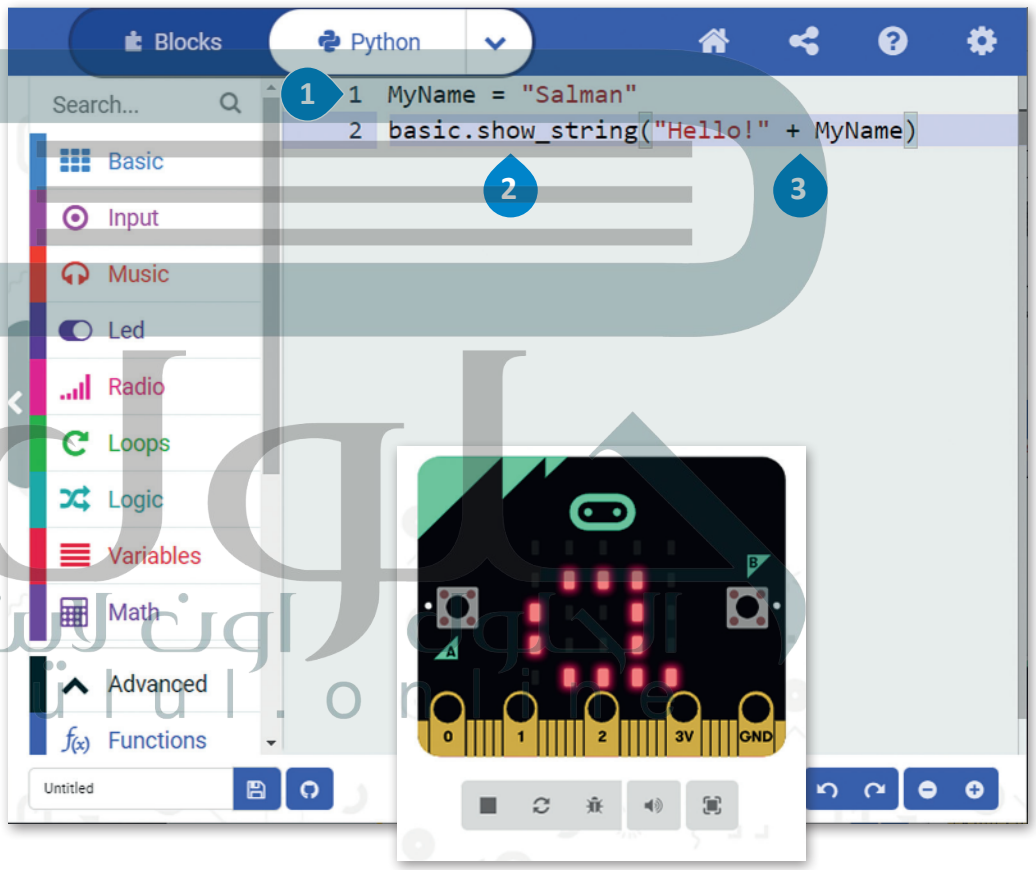
لتعيين قيمة لمتغير نصي:

> من فئة الأوامر **Variables** (المتغيرات) اسحب الأمر **item = 0** (العنصر = 0) وأفلته، اكتب اسم المتغير وقيمته. 1

> من فئة **Basic** (أساسي)، اسحب وأفلت أمر **show string** (إظهار السلسلة). 2

> اضغط بالفأرة داخل القوسين واحذف علامة التعجب، ثم اكتب "+" واسم المتغير **MyName** (اسمي). 3

يجب دائمًا عند استخدام المتغيرات النصية وضع النص بين علامتي الاقتباس " ".



تغيير الأمر

يمكن استخدام المتغيرات لأداء مجموعة متنوعة من المهام. ويقوم الأمر **تغيير** (change) الموجود في فئة أوامر **المتغيرات** (Variables) بتغيير قيمة المتغير المحدد بالقيمة المعينة التي يتم إدخالها. يقتصر استخدام هذه الطريقة على المتغيرات العددية.

تقليل قيمة المتغير	زيادة قيمة المتغير
`item -= عدد`	`item += عدد`

في المثال التالي يقوم البرنامج زيادة قيمة المتغير **العنصر** (item) بمقدار 1:

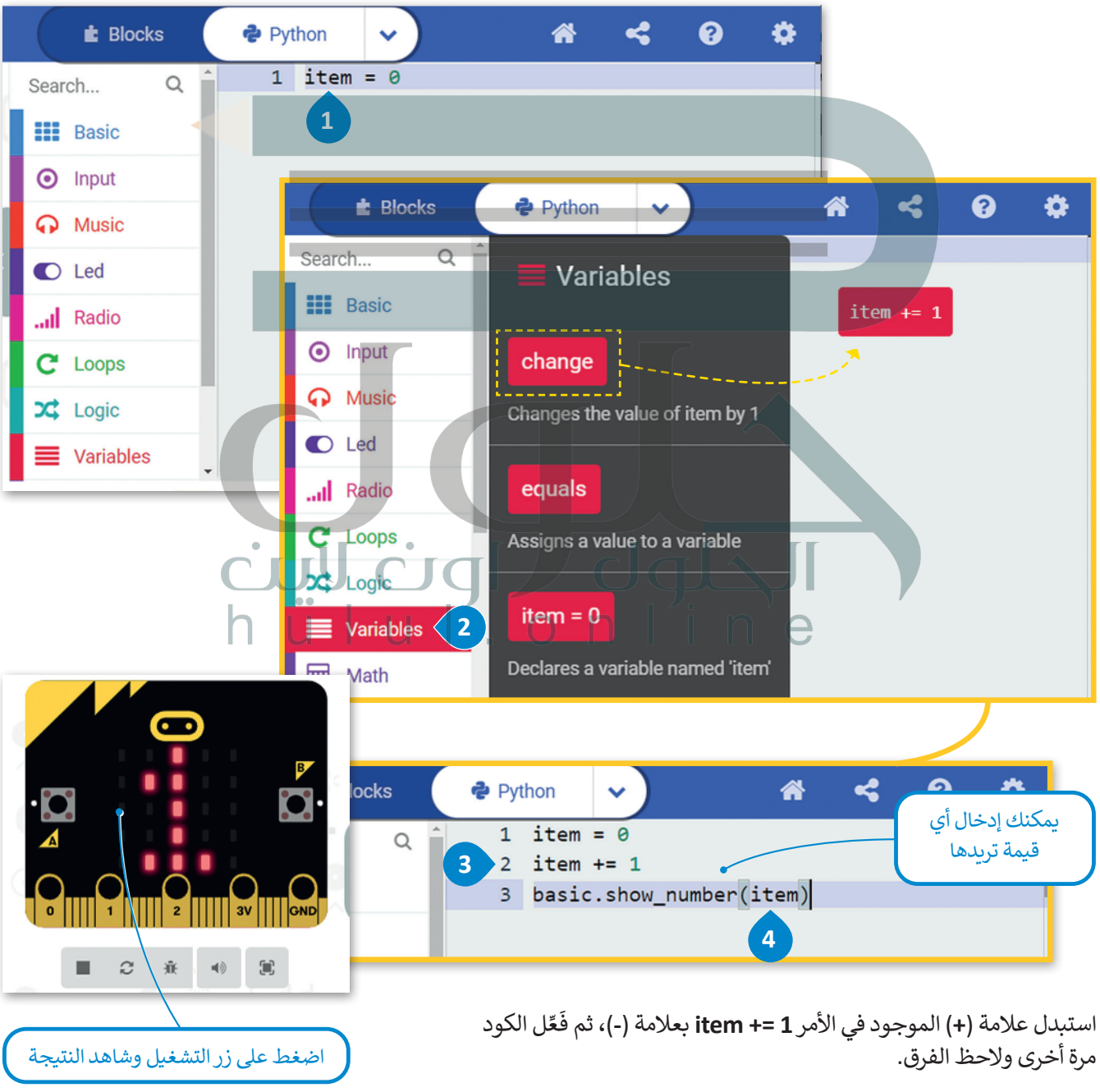
لتغيير قيمة متغير عددي:

> من فئة الأوامر **Variables** (المتغيرات) اسحب الأمر **item = 0** (العنصر = 0) وأفلته. 1

> اضغط على فئة الأوامر **Variables** (المتغيرات). 2

> اسحب وأفلت الأمر **change** (تغيير). 3

> من فئة الأوامر **Variables** (المتغيرات) اسحب وأفلت الأمر **show number** (إظهار الرقم) 4 واكتب داخل الأقواس اسم المتغير **item** (العنصر).



استبدل علامة (+) الموجود في الأمر **item += 1** بعلامة (-)، ثم فَعِّل الكود مرة أخرى ولاحظ الفرق.

المتغيرات المحلية والمتغيرات العامة

يتم تصنيف المتغيرات إلى متغيرات محلية ومتغيرات عامة بناءً على نطاقها. ونطاق المتغير هو الجزء من البرنامج الذي يمكن من خلاله الوصول إلى المتغير ورؤيته واستخدامه.

المتغيرات المحلية	المتغيرات العامة
يتم تعريف المتغيرات المحلية داخل دالة ولذا تنتمي فقط إلى هذه الدالة المحددة، ولا يمكن الوصول إليها إلا من خلال تلك الدالة التي تم تعريفها داخلها.	يتم تعريف المتغيرات العامة خارج أي دالة ويمكن الوصول إليها بشكل عام في البرنامج بأكمله، وبمعنى آخر يمكن استخدامها في أي مكان في البرنامج وليس فقط في النطاق الذي تم تحديده، كداخل الدالة على سبيل المثال.

نطاق المتغير العام

```
variable = 0

def name():
    global variable
    command
close.def()
```

نطاق المتغير المحلي

```
variable = 0

def name():
    command
close.def()
```

أنشئ برنامجًا بحيث تتغير قيمة المتغير myVar بمقدار 1 في كل مرة تضغط فيها على زر A من المايكروبت.

ستستخدم الأمر **عام** (global) للدلالة على أن myVar هو متغير عام، مما يعني أن تعيين قيمة myVar داخل الدالة يغير ما سيعرض عند استخدام القيمة myVar في القسم الرئيس من البرنامج.

أنشئ البرنامج التالي:

يمكن الوصول للمتغيرات العامة في البرنامج من جميع الدوال.

عرّف المتغير قبل استخدامه

Blocks | Python | Microsoft

```
1  myVar = 0
2
3  def on_button_pressed_a():
4      global myVar
5      myVar += 1
6      basic.show_number(myVar)
7  input.on_button_pressed(Button.A, on_button_pressed_a)
```

نطاق المتغير العام myVar

لنطبق معًا

تدريب 1

ما لغة البرمجة عالية المستوى؟

تُعد لغات بايثون وفيجوال بيسك وجافا سكربت لغات برمجة عالية المستوى، لغة البرمجة عالية المستوى، لغة برمجة تستخدم عناصر عادية من اللغة كالكلمات والحروف وتتضمن لغة البرمجة عالية المستوى كلمات يجب تعلمها وكذلك قواعد لبناء الجمل البرمجية يجب اتباعها، كما في اللغات التى يتحدثها البشر

تدريب 2

ما الذي سيعرض على شاشة LED عند تشغيل البرنامج التالي وفقًا للأزرار التي ستضغط عليها؟ اكتب الإجابة الصحيحة.

Python Microsoft

```
1  def on_button_pressed_a():
2      basic.show_string("Left")
3  input.on_button_pressed(Button.A, on_button_pressed_a)
4
5  def on_button_pressed_ab():
6      basic.show_icon(IconNames.HAPPY)
7  input.on_button_pressed(Button.AB, on_button_pressed_ab)
8
9  def on_button_pressed_b():
10     basic.show_string("Right")
11 input.on_button_pressed(Button.B, on_button_pressed_b)
12
```

A	Left
B	Right
A+B	Happy أيقونة

تدريب 3

أنشئ برنامجًا برنامج يعرض عند بدء تشغيله الرسالة "Hello KSA" على شاشة المايكروبت، ثم يعرض أيقونة قلب.

```
basic.show_string("Hello KSA")
basic.show_icon(IconNames.HEART)
```

تدريب 4

هناك كلمات لا يمكن استخدامها كأسماء للمتغيرات، حدد الكلمات التي يمكن استخدامها كاسم للمتغير والتي لا يمكن استخدامها.

الأسماء	يمكن استخدامها	لا يمكن استخدامها
1. global		✓
2. MyAge	✓	
3. False		✓
4. LEDColor	✓	
5. def		✓
6. import		✓

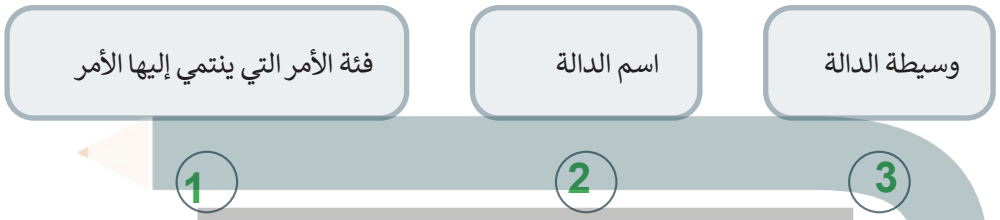
لماذا لا يمكن استخدام هذه الأسماء كاسم متغير في بايثون؟ اشرح إجاباتك.

لا يمكن استخدام بعض الكلمات لتسمية المتغيرات لكونها كلمات خاصة أو مفتاحية مستخدمة بواسطة لغة البرمجة، ونطلق على هذه الكلمات اسم الكلمات المحجوزة.

تدريب 5

اقرأ الكود واكتب الرقم الصحيح في المربعات من أجل تحديد كل مكون من سطر الأوامر.

```
1     2          3
basic.show_string("Hello KSA")
```



تدريب 6

أنشئ برنامجًا يعرض عند البدء أيقونة **HAPPY** على شاشة المايكروبت وعندما يتم تفعيل مستش

```
basic.show_icon(IconNames.HAPPY)
def on_gesture_shake():
    basic.show_icon(IconNames.CONFUSED)
input.on_gesture(Gesture.SHAKE, on_gesture_shake)
```

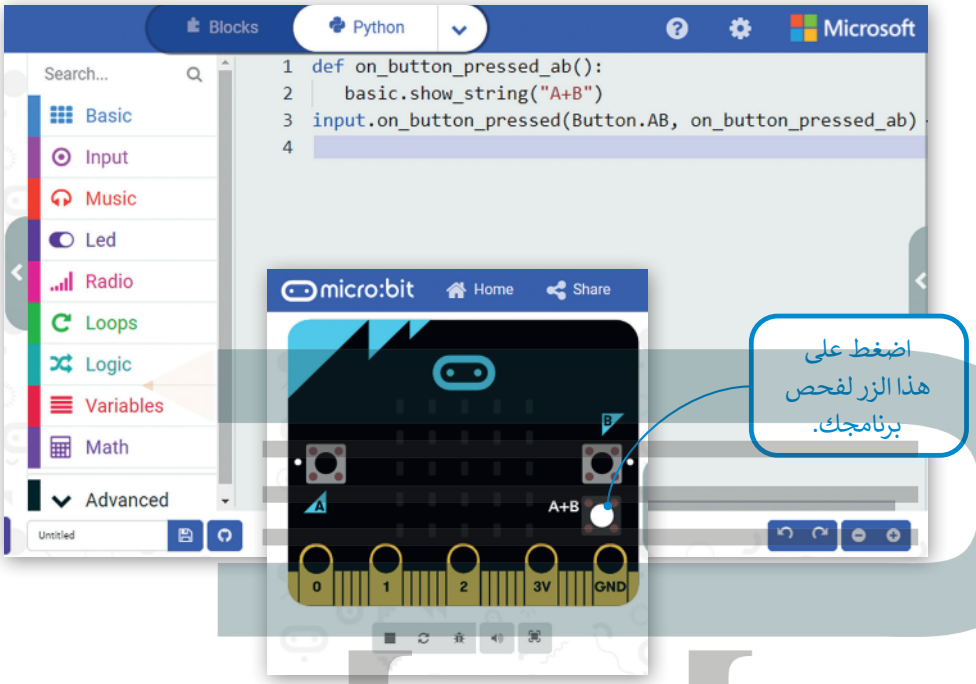
تدريب 7

أنشئ بعض التعليمات البرمجية يتم فيها تقليل قيمة المتغير بمقدار 1 في كل مرة يضغط بها المستخدم على الزر B.

```
myVar = 0
def on_button_pressed_b():
    global myVar
    myVar -= 1
    basic.show_number(myVar)
input.on_button_pressed(Button.B, on_button_pressed_b)
```

تدريب 8

أنشئ البرنامج التالي في مايكروسوفت ميك كود (Microsoft MakeCode).



صِف دالة البرنامج المحدد.

عند الضغط على الزر A+ B فإن السلسلة "A + B" تظهر على شاشة LED الخاصة بالمايكروبت

تدريب 9

أنشئ برنامجًا

> عندما يتم الضغط

> عندما يتم الضغط

```
def on_button_pressed_a():
        basic.show_string("Left")
input.on_button_pressed(Button.A, on_button_pressed_a)

def on_button_pressed_b():
        basic.show_string("Right")
input.on_button_pressed(Button.B, on_button_pressed_b)
```

رابط الدرس الرقمي

www.ien.edu.sa

الدرس الثاني:

المتغيرات والتكرارات

بعد أن تعرفت في الدرس السابق على بيئة مايكروسوفت ميك كود، ستتعلم في هذا الدرس كيفية إجراء العمليات الرياضية باستخدام الأرقام، وكيفية التعامل مع الإحداثيات، كما ستتعرف على كيفية تنفيذ التكرار أثناء البرمجة، وعملية التكرار من المزايا الموجودة في معظم لغات البرمجة.

الحسابات والأرقام

يمكنك استخدام بايثون لإجراء أي نوع من العمليات الرياضية، ولكن يجب ملاحظة أن العمليات مثل: الجمع والطرح والضرب والقسمة تُكتب في البرمجة بطريقة مختلفة عن تلك التي تُكتب بها في العمليات الرياضية (الحسابية)، حيث تستخدم المعاملات الرياضية التالية لتمثيل العمليات الحسابية الأساسية.

العمليات الحسابية	بلغة بايثون	رياضيًا
الجمع	4+2	4 + 2
الطرح	4–2	4 – 2
الضرب	4*2	4 × 2
القسمة	4/2	4 ÷ 2
الأس	x**2	x^2

على سبيل المثال، يجب أن تتم كتابة المعادلة الرياضية التالية:

$$x = a^2 + 2ab + b^2$$

في بايثون كما يلي:

```
x=a**2+2*a*b+b**2
```

يتم تنفيذ عوامل التشغيل بالترتيب من اليسار إلى اليمين.

يُحدد ترتيب العمليات في بايثون سابقًا، وتنطبق عليها نفس القواعد التي سبق أن تعلمتها في مايكروسوفت إكسل بشأن استخدام الأقواس.

يتم حساب عمليات الضرب والقسمة قبل عمليات الجمع والطرح، وهذا يعني مثلًا أن ناتج 4 + 2 * 5 هو 14 وليس 30.

في حال أردت تغيير أولوية العمليات الحسابية، يتعين عليك استخدام الأقواس. يظهر ترتيب العمليات الحسابية كما في الجدول المجاور، حيث يتم تنفيذ المعاملات في نفس المستوى بالترتيب من اليسار إلى اليمين.

يمكنك العثور على المعاملات الرياضية في مايكروسوفت ميك كود في فئة **حساب** (Math).

أنشئ برنامجًا في مايكروسوفت ميك كود بايثون يجمع رقمين عند اهتزاز المايكروبت.

أولوية العمليات الحسابية	
( )	الأقواس
**	الأس
* /	الضرب والقسمة
+ -	الجمع والطرح

لإضافة عملية الجمع:

> من فئة **Variables** (متغيرات)، اسحب وأفلت أمر **item = 0** (العنصر =0)، اكتب اسم المتغير **add** (إضافة). 1

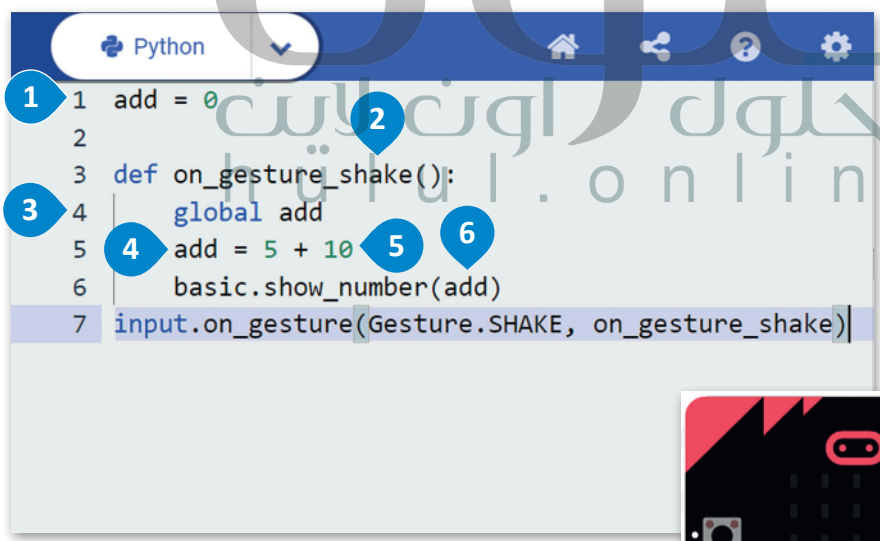
> من فئة **Input** (**الإدخال**)، اسحب وأفلت دالة **run code on Gesture.Shake** (run code عند Gesture.Shake). 2

> اكتب الأمر **global add** (إضافة عامة). 3

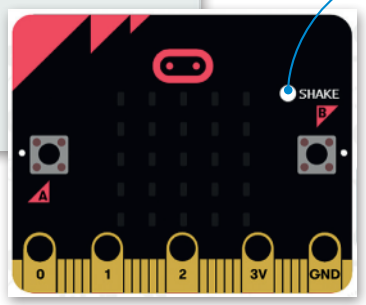
> من فئة **Variables** (المتغيرات)، اسحب وأفلت أمر المساواة، واكتب **add** (إضافة) على الجانب الأيسر. 4

> من فئة **Math** (حساب)، اسحب وأفلت أمر الجمع داخل الجملة البرمجية ثم اكتب الأرقام التي تريد جمعها. 5

> من فئة **Basic** (أساسي)، اسحب وأفلت أمر **show number** (إظهار الرقم)، واكتب **add** (إضافة) داخل الأقواس. 6



اضغط على زر (Shake) للتحقق من النتيجة من الكود.

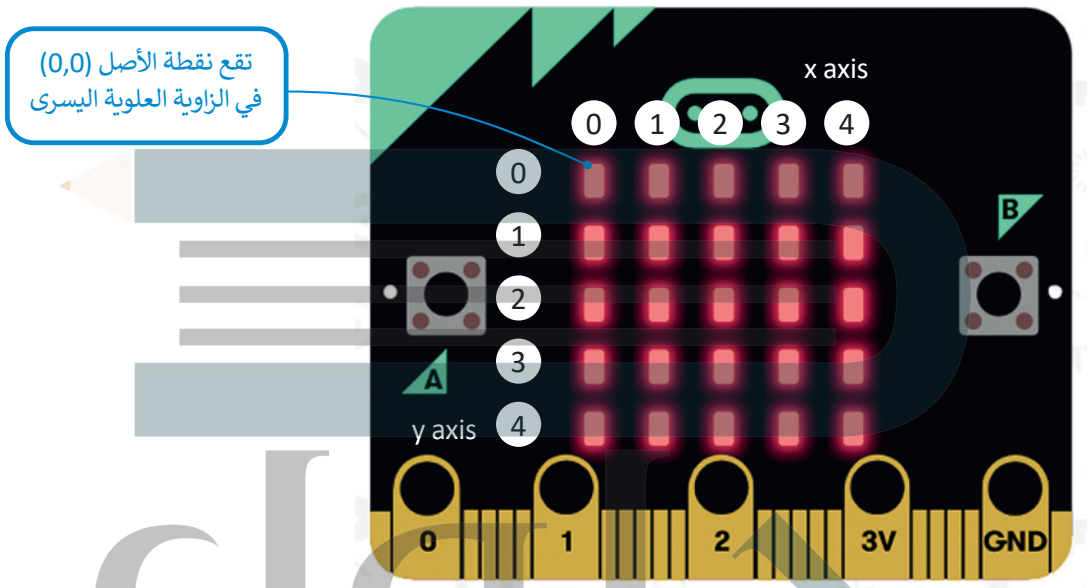


يُطلق على الرموز التي تساعدك على إجراء العمليات الرياضية اسم المعاملات الرياضية.

الإحداثيات في بايثون

يتم تمثيل مصابيح Led الموجودة في مايكروبت على شكل شبكة إحداثيات بمحور سيني (x) أفقي ومحور صادي (y) عمودي، وتحتوي هذه الشبكة على خمسة صفوف وخمسة أعمدة من المصابيح. يختلف نظام التمثيل هذا عن نظام الإحداثيات الديكارتي المعتاد المستخدم في الرياضيات، حيث يشبه نظام إحداثيات مقلوبة.

توجد النقطة (0،0) في الزاوية اليسرى العلوية وتسمى نقطة الأصل التي تُمكنك من تحديد موضع مصابيح Led باستخدام الإحداثيات الثنائية. وتتراوح قيم إحداثيات x بين 0 إلى 4 تمامًا كما هو الحال في شبكة الإحداثيات المستخدمة في الرياضيات، وتزداد قيمها من اليسار إلى اليمين. بينما إحداثيات y تتراوح قيمها بين 0 إلى 4 وتزداد قيمها من الأعلى إلى الأسفل.



أوامر اللعب

حان الوقت لتتعرف على كيفية إنشاء لعبة بسيطة باستخدام المايكروبت. ستكون "شخصية" لعبتك هي كائن ضوئي، ويتم تحديد موقعه والتحكم في حركته باستخدام نظام الإحداثيات. ستنشئ برنامجًا يتحرك فيه الكائن إلى اليسار عند الضغط على الزر **A**.

لمحة تاريخية

يُعدُّ رينيه ديكارت (1596-1650) الفيلسوف وعالم الرياضيات الفرنسي أول من طور نظام الإحداثيات المستخدم في أيامنا هذه، وقد حدث ذلك حين كان مستلقيًا على سريره وأراد إيجاد طريقة دقيقة لتحديد موضع الذبابة التي لاحظها على سقف الغرفة.

لإنشاء الكائن الرسومي:

> من فئة **Variables** (متغيرات)، اسحب وأفلت أمر **item = 0** (العنصر = 0)، واكتب **player** (لاعب) على الجانب الأيسر. 1

> اضغط على فئة **Advanced** (متقدم). 2

> من فئة **Game** (اللعبة)، اسحب وأفلت الأمر **create sprite at x:2 y:2** (إنشاء كائن رسومي في x:2 و y:2). 3

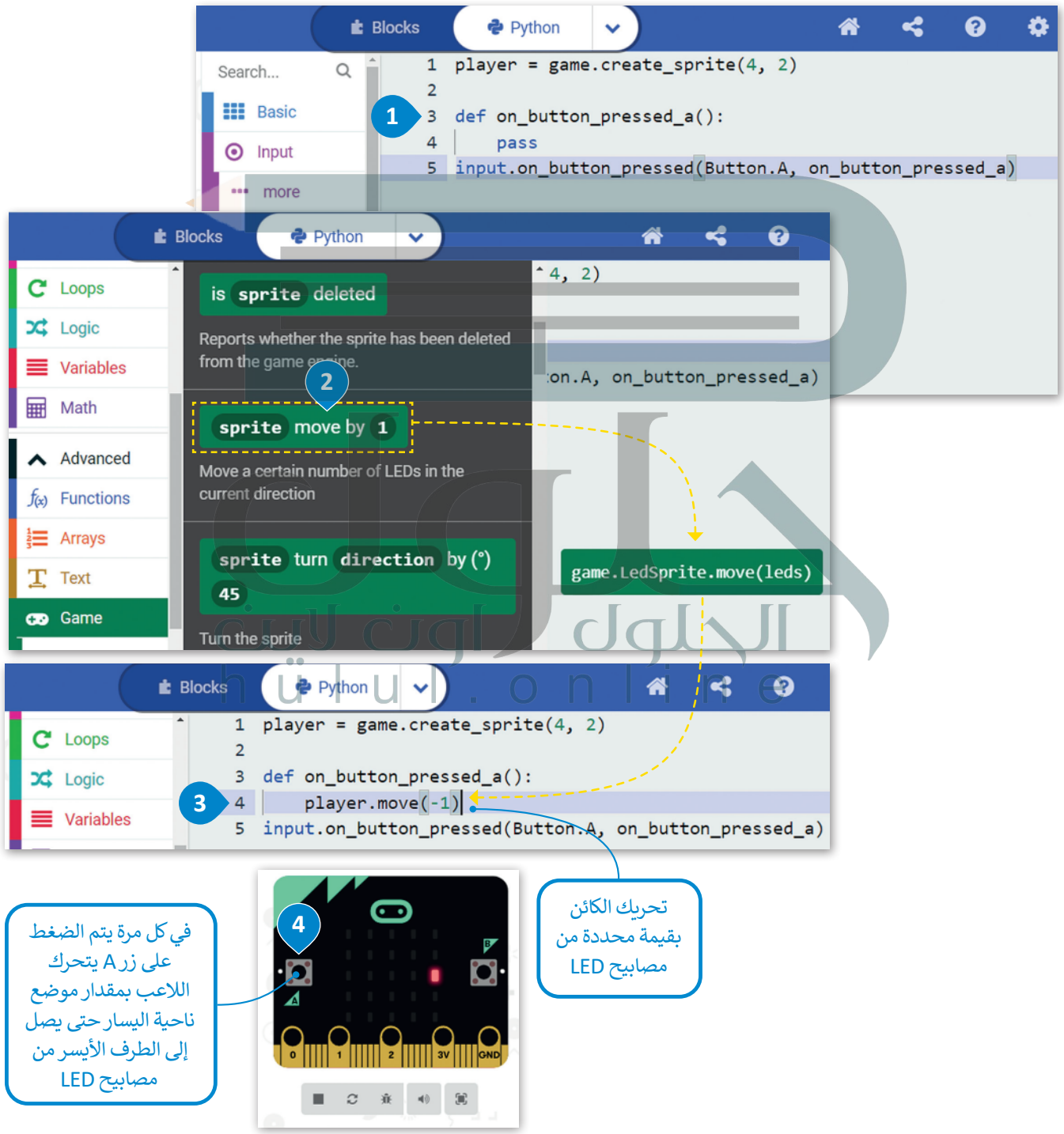
> اضبط موضع اللاعب على إحداثيات **(4, 2)** من شاشة **LED**. 4

لجعل الكائن الرسومي يتحرك في شاشة LED:

> من فئة **Input** (الإدخال)، اسحب وأفلت أمر **run code on button pressed** (عندما يكون زر run code مضغوط). 1

> من فئة **Game** (اللعبة)، اسحب وأفلت أمر **sprite move by 1** (نقل الكائن الرسومي بمقدار 1) 2، واكتب **player** (لاعب) على الجانب الأيسر وأضف القيمة **-1** داخل الأقواس. 3

> اضغط على زر **A** في المحاكي للتحقق من النتيجة. 4

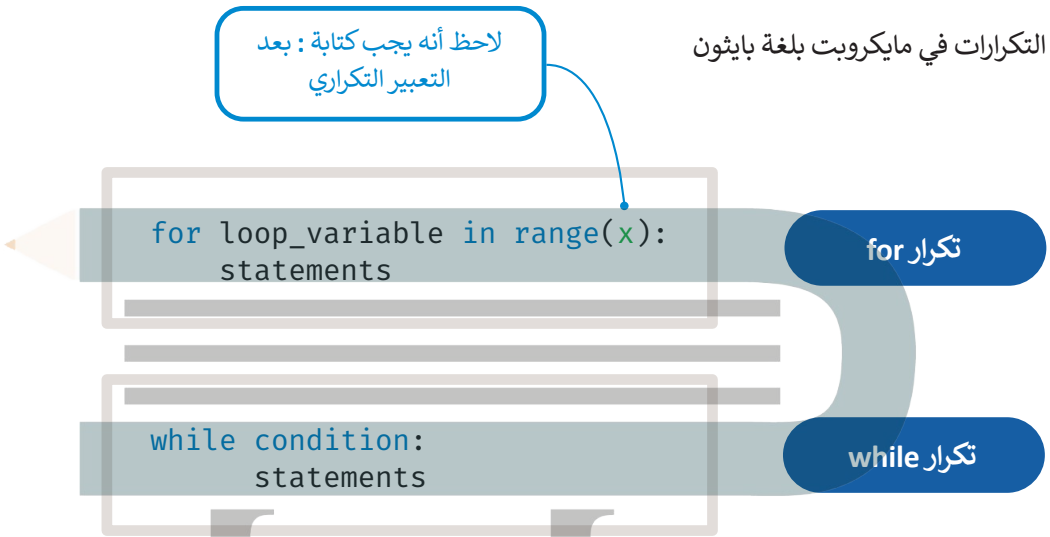


التكرارات

تحتاج أحيانًا إلى تكرار جزء من البرنامج عدة مرات في البرمجة، ولهذا السبب فإن معظم لغات البرمجة توفر دوال مختلفة خاصة بالتكرارات البرمجية. تسمح لك التكرارات بتنفيذ سطر واحد أو مجموعة من التعليمات البرمجية لعدة مرات. توفر بايثون عددًا من أوامر التكرار التي تساعدك على تجنب إعادة كتابة أوامر التعليمات البرمجية، وتدعم بايثون نوعين من التكرارات: تكرار **for** وتكرار **while**.

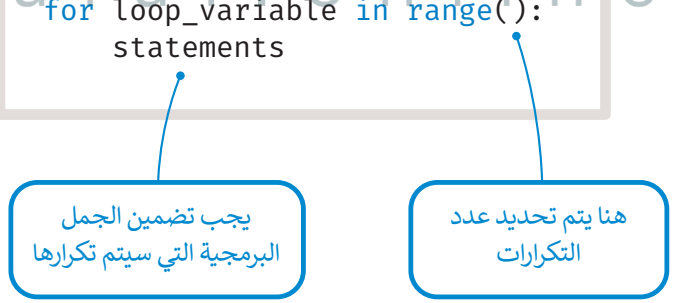
الفرق بين تكرار **for** وتكرار **while** هو أنه في تكرار **for** يكون عدد التكرارات التي يتعين إجراؤها محدد بالفعل ويستخدم للحصول على نتيجة محددة بينما يعمل الأمر أثناء تكرار **while** حتى يتم الوصول إلى حالة معينة ويتم إثبات العبارة خاطئة.

التكرارات في مايكروبت بلغة بايثون



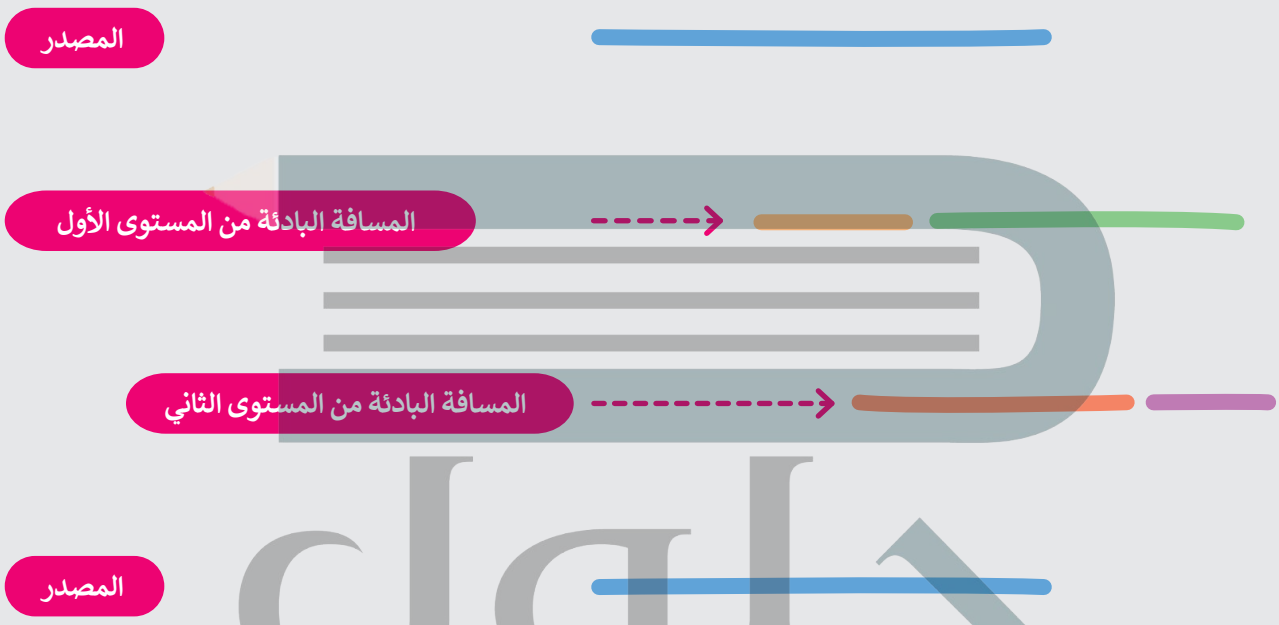
تكرار for

يتم استخدام تكرار **for** إذا أردت تكرار مجموعة من الأوامر لعدد محدد من المرات. يتم تحديد عدد التكرارات في **نطاق** (range).



كن حذرًا عند استخدام المسافة البادئة.

تعدّ المسافة البادئة مهمة جدًا في بايثون وهي إضافة مسافة (فراغ) قبل العبارة. وتشبه ترقيم صفحات الكتاب بالنسبة للقارئ، فبدون أرقام الصفحات لا يعرف القارئ مكان مواصلة القراءة وقد يختلط عليه الأمر. بنفس الطريقة يعمل بايثون، فبدون المسافة البادئة لا يعرف أي عبارة تالية سيقوم بتنفيذها أو أي عبارة تنتمي إلى أي لبنة ولن يتم تنفيذ الكود.



المسافة البادئة من المستوى الأول

```
def on_forever():
    for i in range (10):
        basic.show.number(i)
basic.forever(on_forever)
```

المسافة البادئة من المستوى الثاني

مثال برمجي: سقوط المطر

تعرفت في الدرس السابق على مثال يحرك به اللاعب كائنًا يسارًا بالضغط على الزر **A**. سترى في هذا المثال كيف يمكنك تطبيق تكرار **for** لجعل الكائن يبدو كأنه يسقط من الأعلى.

ستنشئ برنامجًا يُمثل سقوط المطر على شاشة المايكروبت.

لإنشاء كائن رسومي للمطر:

> من فئة **Basic** (أساسي)، اسحب وأفلت دالة **run code forever** (للأبد run code). 1

> عرّف متغير باسم **rain** (مطر) ومن فئة **Game** (اللعبة)، اسحب وأفلت **create sprite at x:2 y:2** (إنشاء كائن رسومي في x:2 و y:2) على الجانب الأيمن. 2

> من فئة **Math** (حساب)، اسحب وأفلت أمر **randint** وعيّن القيم داخل الأقواس كالتالي **(0,4),0)**. 3

يتيح لك تكرار "للأبد" (forever) تشغيل جزء من البرنامج بشكل مستمر في الخلفية. وفي كل تكرار يسمح بتشغيل الأكواد الأخرى في نفس الوقت، حيث أن الكود الموجود داخل تكرار "للأبد" (forever) سينتج عن الكود الآخر الموجود في برنامجك.

اتبع الخطوات التالية لإكمال برنامج سقوط المطر.

لإنشاء الكائن الرسومي باستخدام التكرارات:

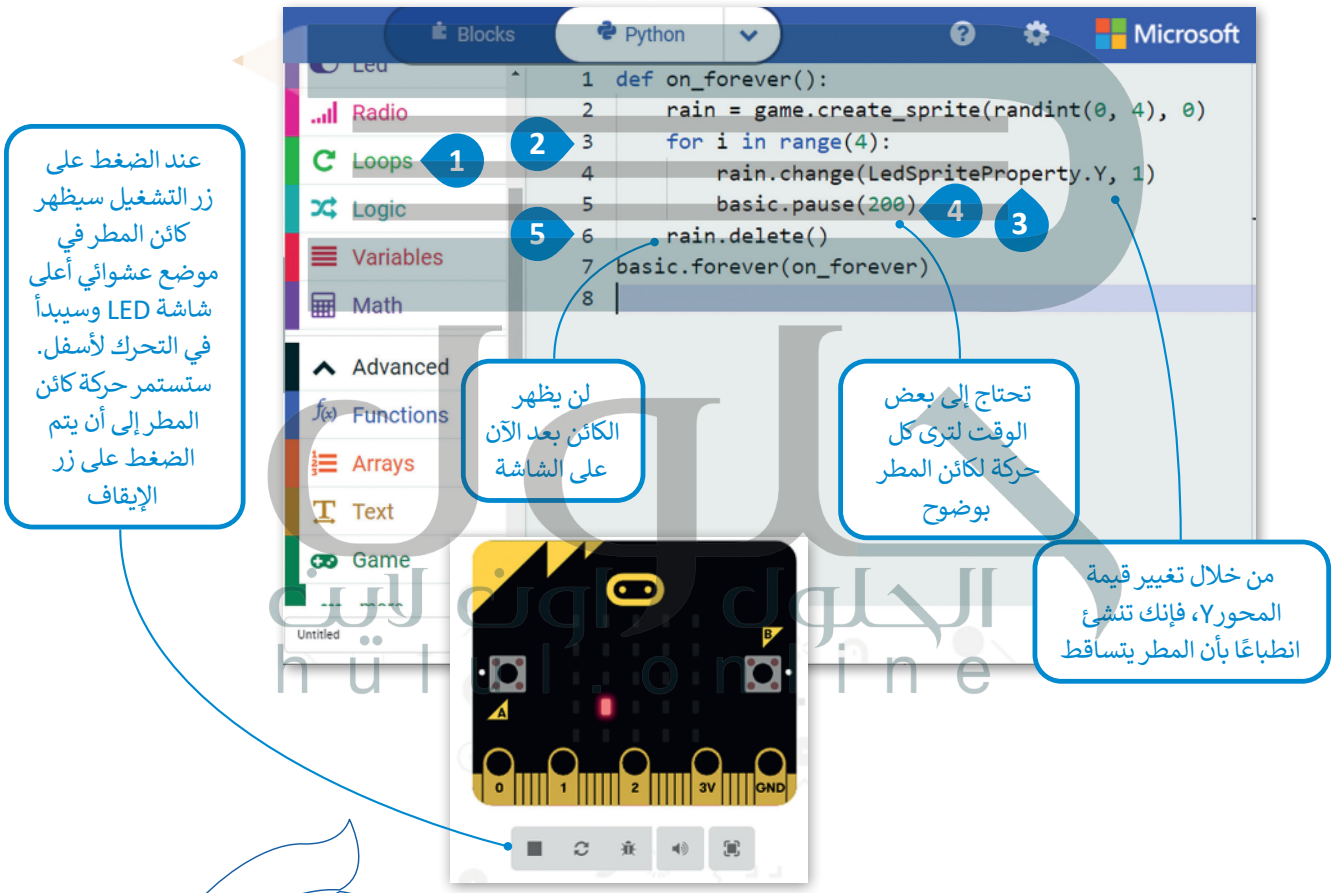
> اضغط على فئة **Loops** (حلقات). 1

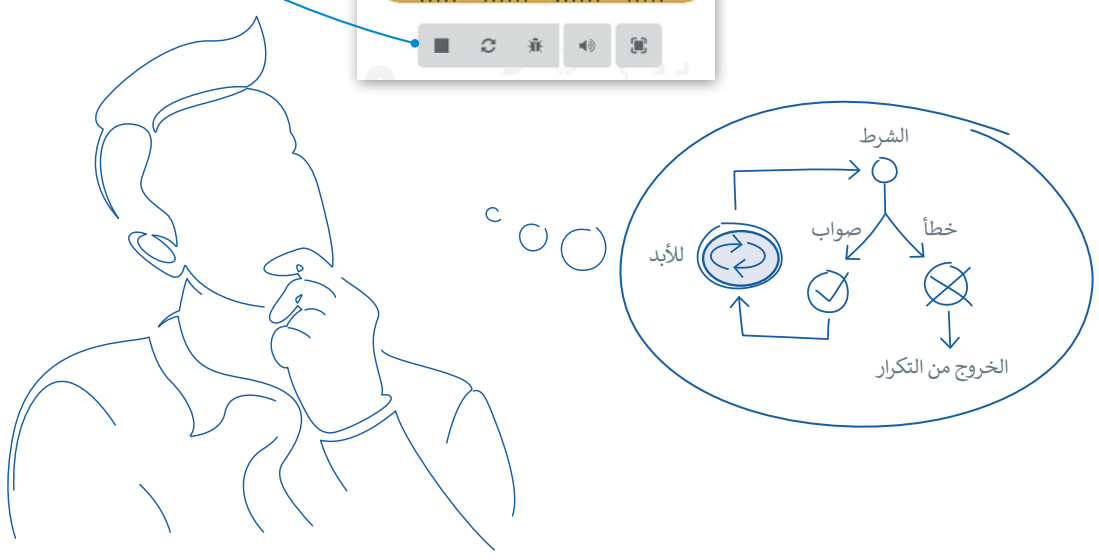
> حدد دالة **for** وضعها داخل دالة **run code forever** (للأبد run code). 2

> من فئة **Game** (اللعبة)، اسحب وأفلت **sprite change property by 1** (تغيير خاصية الكائن الرسومي بمقدار 1)، واضبط الكائن إلى **rain** (مطر) و **property** (خاصية) إلى **Y**. 3

> من فئة **Basic** (أساسي)، اسحب وأفلت أمر **pause (ms)** (إيقاف مؤقت (مللي ثانية)) واضبط **time** (الوقت) إلى **200**. 4

> من فئة **Game** (اللعبة)، اسحب وأفلت أمر **delete sprite** (حذف الكائن الرسومي) واضبط الكائن الرسومي إلى **rain** (مطر). 5





يتم استخدام تكرار **for** عندما يكون عدد التكرارات محددًا منذ البداية. ماذا تفعل عندما يكون هذا الرقم غير معروف ويعتمد التكرار على شرط؟ في مثل هذه الحالا يقدم بايثون لك تكرار **while**.

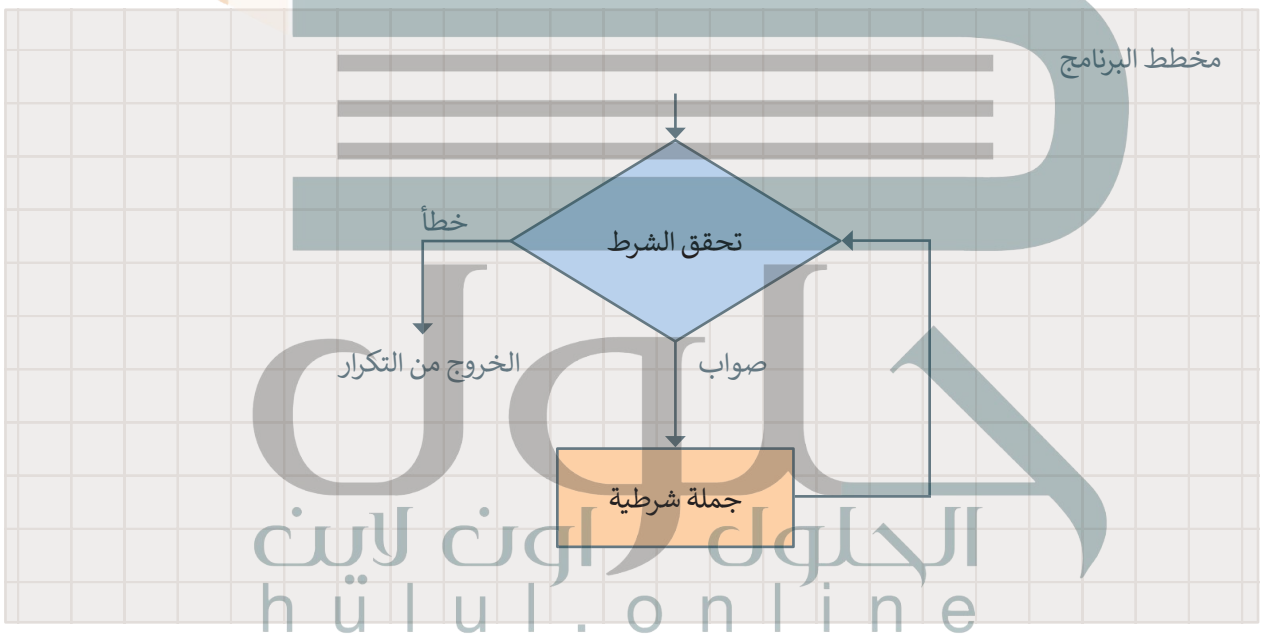
تكرار While

يتم استخدام تكرار while عندما يكون عدد التكرارات غير معروف (أو محدد) سابقًا.

كلما كان الشرط متحقّقًا يستمر التكرار في عمله لفحص الحالة بصورة مستمرة بعد كل تكرار، وعند عدم تحقق الشرط، فإن التكرار يتوقف ليمرر التحكم في البرنامج إلى السطر الذي يلي التكرار. أما عند عدم تحقق الشرط منذ البداية، فإن عبارات التكرار لن يتم تنفيذها إطلاقًا.

يجب إضافة مسافة بادئة لجمل التكرار

```
while condition:
    statements
```



لتلق نظرة على مثال مع تكرار **while**. سيظهر في هذا المثال الحرف "**A**" على الشاشة طالما استمر المستخدم بالضغط على الزر **A**، وسينتهي التكرار عند توقف المستخدم عن الضغط على زر **A**.

```
def on_forever():
    while input.button_is_pressed(Button.A):
        basic.show_string("A")
    basic.show_icon(IconNames.NO)
basic.forever(on_forever)
```

إذا لم يتم الضغط على الزر A باستمرار، فلن يكون الشرط متحقّقًا وبالتالي لن يتم تنفيذ الأوامر داخل التكرار

التكرار اللانهائي

حلقة التكرار اللانهائي في بايثون هي حلقة شرطية متكررة ومستمرة يتم تنفيذها حتى يتدخل عامل خارجي في عملية التنفيذ مثل: الذاكرة غير الكافية أو الضغط على زر الإيقاف.

إذا لم تصبح حالة تكرار **while** غير متحققة، يصبح لديك تكرار لا نهائي، وهو التكرار الذي لا يتوقف أبدًا. عند استخدام تكرار **while**، يجب عليك تضمين أمر أو مجموعة من الأوامر التي تغير حالة الشرط من متحقق إلى غير متحقق.

لتطبق الجملة البرمجية التالية، ما الذي تلاحظه؟

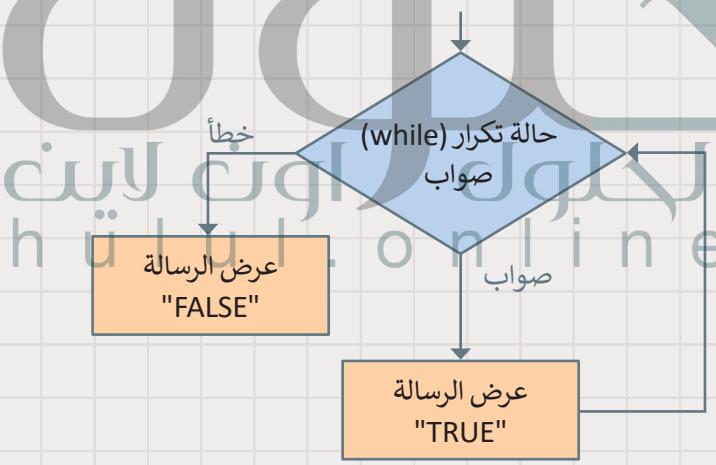
```
while True:
    basic.show_string("TRUE")
basic.show_string("FALSE")
```

ستعرض الشاشة ما يلي:

TRUE

في المثال السابق ستعرض الرسالة **TRUE** بشكل مستمر (إلى الأبد)، بينما لن تعرض رسالة **FALSE** على الشاشة نهائيًا.

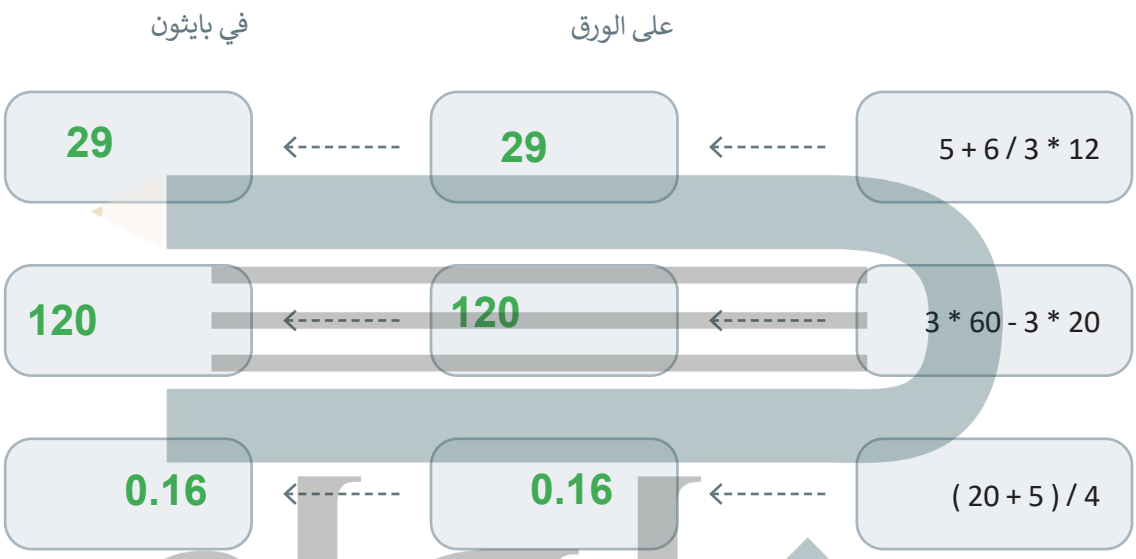
مخطط البرنامج



لنطبق معًا

تدريب 1

احسب نتيجة العمليات الحسابية على الورق أولًا ثم طبق هذه العمليات في بايثون.



تدريب 2

أنشئ بعض التعليمات البرمجية التي تجعل الكائن يتحرك إلى اليمين عند الضغط على الزر B.
ما القيمة التي يجب وضعها للأمر player.move( ) يتحرك اللاعب؟

```
player = game.create_sprite(2, 2)
def on_button_pressed_b():
        player.move(1)
input.on_button_pressed(Button.B, on_button_pressed_b)
```

تدريب 3

املأ الفراغات في العبارات التالية بالكلمات المناسبة مما يلي، مع ملاحظة أنه يمكن استخدام بعض الكلمات عدة مرات:

True، False، لانهائية، for، while، النطاق، التكرارات، مرات، الشرط.

1. عندما تريد تكرار مجموعة من الأوامر، يتم استخدام عدد محدد من ___مرات___ الحلقة ___for___. تم تحديد عدد ___التكرارات___ في معلمات ___النطاق___ ( ).

2. عندما يكون عدد ___التكرارات___ غير معروف سابقًا، يتم استخدام الحلقة ___while___. طالما أن الشرط ___false___ فإن الحلقة تتكرر. بعد كل تكرار يتم فحص ___النطاق___. عندما تصبح الحالة ___True___ ، يتوقف التكرار ويمر التحكم في البرنامج إلى السطر الذي يلي الحلقة.

3. إذا كان الشرط مبدئيًا ___false___ ، فلن يتم تنفيذ عبارات حلقة while على الإطلاق.

4. إذا لم يصبح ___الشرط___ الحلقة while ___false___ ، فسوف ينتهي بك الأمر بحلقة ___لا نهائية___. الحلقة اللانهائية هي حلقة لا تنتهي أبدًا.

5. عند استخدام حلقة while، يجب عليك تضمين أمر أو مجموعة أوامر من شأنها تغيير الحالة من ___True___ إلى ___false___.

تدريب 4

جرب البرنامج التالي، واكتب ما يظهر على الشاشة ومتى يحدث ذلك.

```
def on_forever():
    while input.is_gesture(Gesture.SHAKE):
        basic.show_string("Earthquake!")
    basic.show_icon(IconNames.SQUARE)
basic.forever(on_forever)
```

بعد الضغط على زر التشغيل يتم عرض أيقونة مربع على شاشة LED إلى الأبد، وعند اهتزاز المايكروبت يعرض في شاشة LED رسالة زلزال

تدريب 5

كم مرة سينفذ الأمر basic.show_number( )؟ اختر الإجابة الصحيحة:

```
def on_forever():
    for index in range(5):
        basic.show_number(index)
basic.forever(on_forever)
```

- ◯ لن يعمل البرنامج لأن بناء جملة الأوامر غير صحيح.
- ◯ تعرض "1" و"2" و"3" و"4" و"5" على الشاشة.
- ✓ تعرض "0" و"1" و"2" و"3" و"4" على الشاشة.

```
def on_forever():
    for index in range(3):
        index = 0
        basic.show_number(index)
basic.forever(on_forever)
```

- ✓ تعرض "0" على الشاشة.
- ◯ تعرض "0" و"1" و"2" و"3" على الشاشة.
- ◯ تعرض "0" و"3" على الشاشة.

تدريب 6

شغِّل البرنامج وصِف وظيفته.

```
player = game.create_sprite(0, 0)
for i in range(5):
    for j in range(5):
        player.set(LedSpriteProperty.Y, i)
        player.set(LedSpriteProperty.X, j)
        basic.pause(400)
```

ينشئ البرنامج كائن باسم اللاعب في الموضع (0.0) أعلى يسار الشاشة الخاصة بالمايكروبت، ثم استخدم إحداثيات الكائن (X.Y) لإعطاء الصورة الوهمية التي تتحرك على شاشة LED يعد ذلك يجعل البرنامج الكائن يتحرك في موضع واحد في كل مرة من اليسار إلى اليمين وعندما يصل الكائن إلى الموضع الصحيح لخط LED فإنه يستمر إلى الخط التالي

تدريب 7

اكتب برنامجًا يعرض باستمرار رمز البطة على الشاشة، كما يعرض الرسالة "Quack" عند الضغط على الزر B.


```
def on_forever():
    basic.show_icon(IconNames.DUCK)
basic.forever(on_forever)

def on_button_pressed_b():
    basic.show_string("Quack")
input.on_button_pressed(Button.B, on_button_pressed_b)
```

رابط الدرس الرقمي

www.ien.edu.sa

الدرس الثالث:

اتخاذ القرارات

في معظم البرامج التي أنشأتها حتى الآن تم تنفيذ الأوامر بالتتابع واحدًا تلو الآخر، ولكن في بعض الأحيان يكون ترتيب عمليات التنفيذ وفقًا لطبيعة المشكلة. ستتعلم في هذا الدرس كيفية إنشاء برامج تستجيب لمدخلات المستخدم أثناء تنفيذها وتعطي نتائج مختلفة لمدخلات مختلفة. لتحقيق ذلك، ستتعرف على أنواع المعاملات والمستشعرات الشرطية.

المعاملات الشرطية في بايثون

تُستخدم المعاملات الشرطية لاتخاذ القرارات في البرمجة، حيث تقارن بين القيم وتُعيد نتيجة واحدة من اثنتين: صواب أو خطأ. يمكنك في الشكل المجاور التعرف على المعاملات الشرطية في بايثون.

المعامل	المعنى
==	يساوي
>	أكبر من
<	أصغر من
>=	أكبر من أو يساوي
<=	أصغر من أو يساوي
!=	لا يساوي

عندما تريد اتخاذ قرار في بايثون، فإنك تستخدم جملة if. ستجد أوامر if في مايكروبت في فئة أوامر **المنطق** (Logic). هناك ثلاث طرق للتعبير عن جملة if كما في الشكل أدناه:

كن حذرًا عند استخدام الأقواس، وتذكر أنه يجب إغلاق كل قوس يتم فتحه.

أنواع الجمل الشرطية

```
if الشرط:
    العبارة
```

```
if الشرط:
    العبارة 1
else:
    العبارة 2
```

```
if الشرط:
    العبارة 1
elif:
    العبارة 2
else:
    العبارة 3
```

لاحظ أن النقطتين (:) اللتين تليان التعبير الشرطي ضروريتان

```
if الشرط:
    العبارة
```

معلومة

تجمع جملة if...elif...else بين جملة if وجملة if...else.

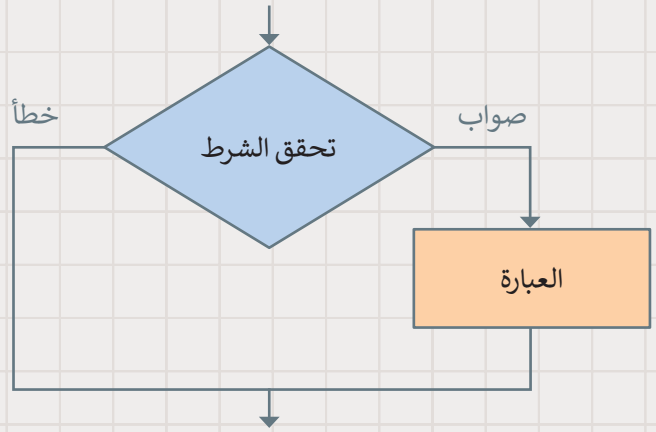
جملة if الشرطية البسيطة

في جملة if البسيطة. إذا تحقق الشرط فستنفّذ العبارة (العبارات) التي تتبع if.

إذا لم يتحقق الشرط فلن تنفّذ العبارة (العبارات).

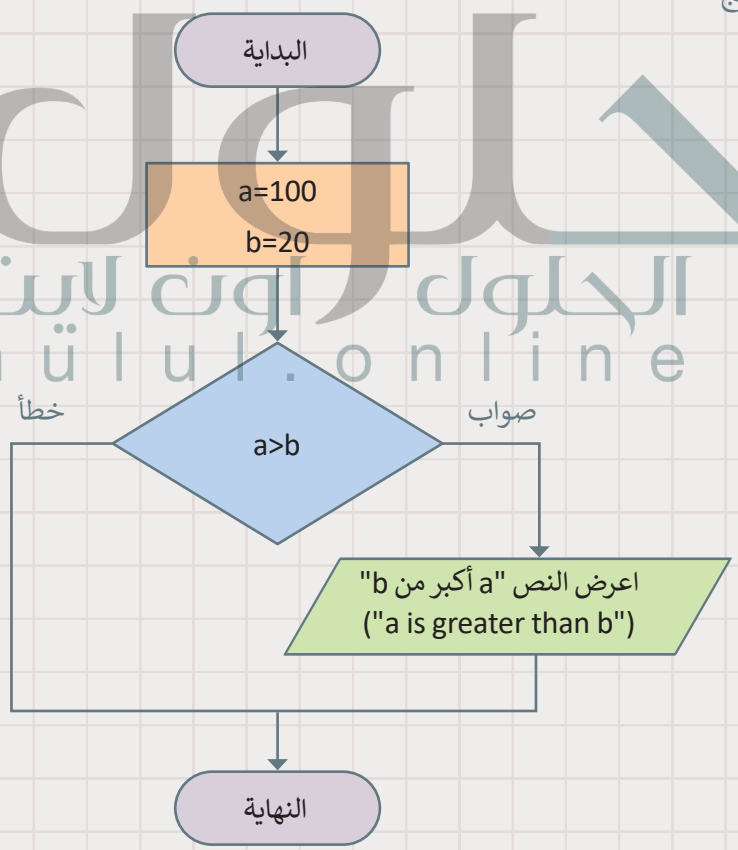
تستخدم بايثون المسافة البادئة للإشارة إلى العبارات المعتمدة على تحقق الشرط.

المخطط الانسيابي للجملة الشرطية



حان الوقت لتستعرض مثالًا.

المخطط الانسيابي للبرنامج



للمقارنة بين متغيرين:

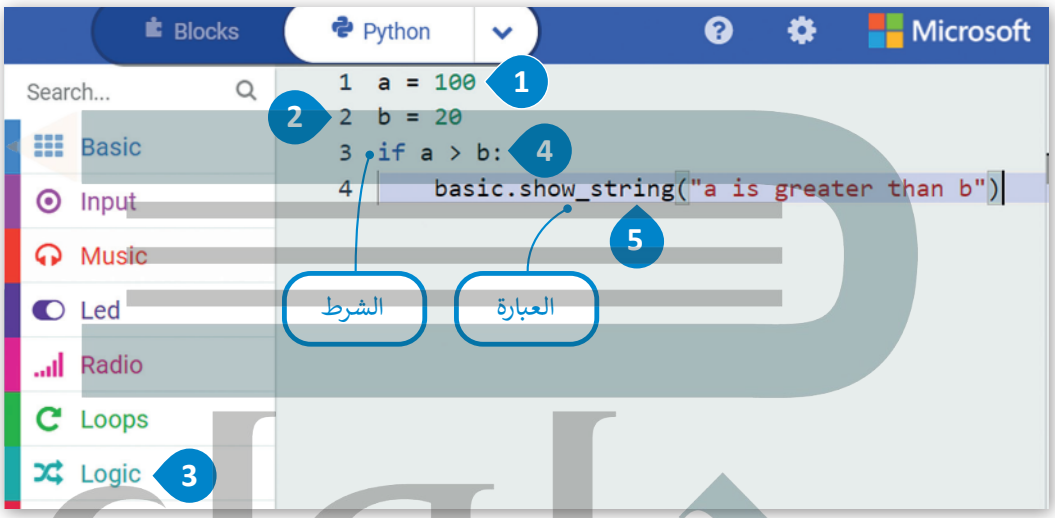
> أعلن عن المتغير **a** وعيّن قيمته إلى **100**. 1

> أعلن عن المتغير **b** وعيّن قيمته إلى **20**. 2

> اضغط على فئة **Logic** (المنطق). 3

> اسحب وأفلت دالة **if**، اكتب الشرط كالتالي: **a>b**. 4

> من فئة **Basic** (أساسي)، اسحب وأفلت أمر **show string** (إظهار السلسلة)، واكتب داخل النص جملة **"a is greater than b "** (" a أكبر من b ") 5



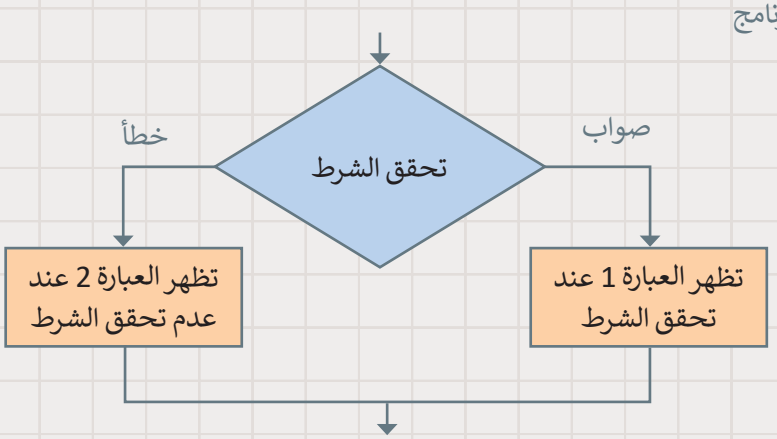
جملة if... else الشرطية

عند استخدامك جملة **if ... else** الشرطية. إذا تحقق الشرط، فستنفّذ العبارة (العبارات) التي تتبع **if**، أما إذا لم يتحقق الشرط، فستنفّذ العبارة (العبارات) الموجودة ضمن شرط آخر.

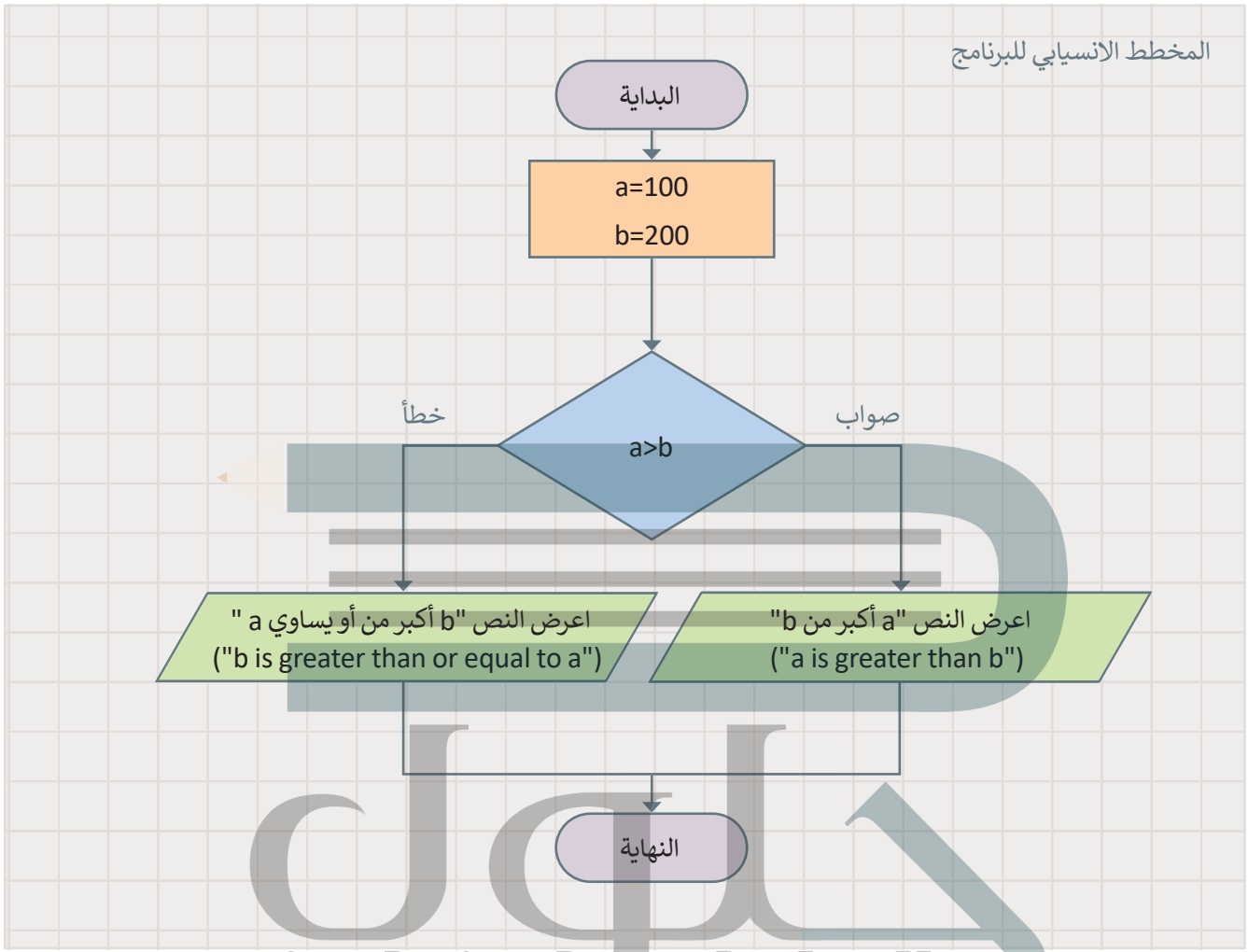
كما في الحالة السابقة، يتم استخدام المسافة البادئة للإشارة إلى العبارات التي ستنفّذ كل مرة.

if الشرط:
العبارة 1
else:
العبارة 2

المخطط الانسيابي للبرنامج



وإليك مثال: أنشئ البرنامج التالي. ستجد الأمر **if ... else** في فئة أوامر **المنطق** (Logic).




```
1  a = 100
2  b = 200
3  if a > b:
4      basic.show_string("a is greater than b")
5  else:
6      basic.show_string("b is greater than or equal to a")
7
```

الشرط

جملة If

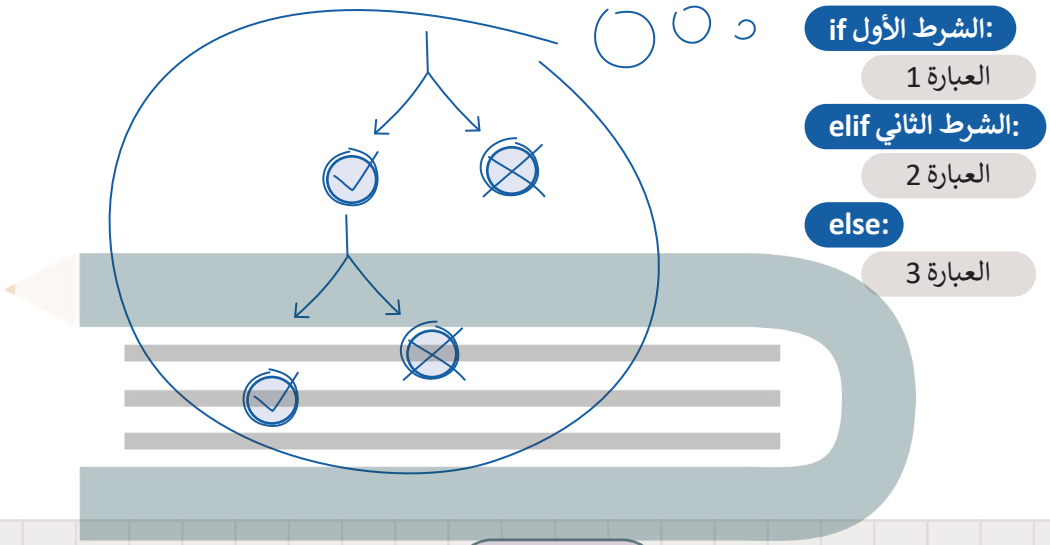
جملة else

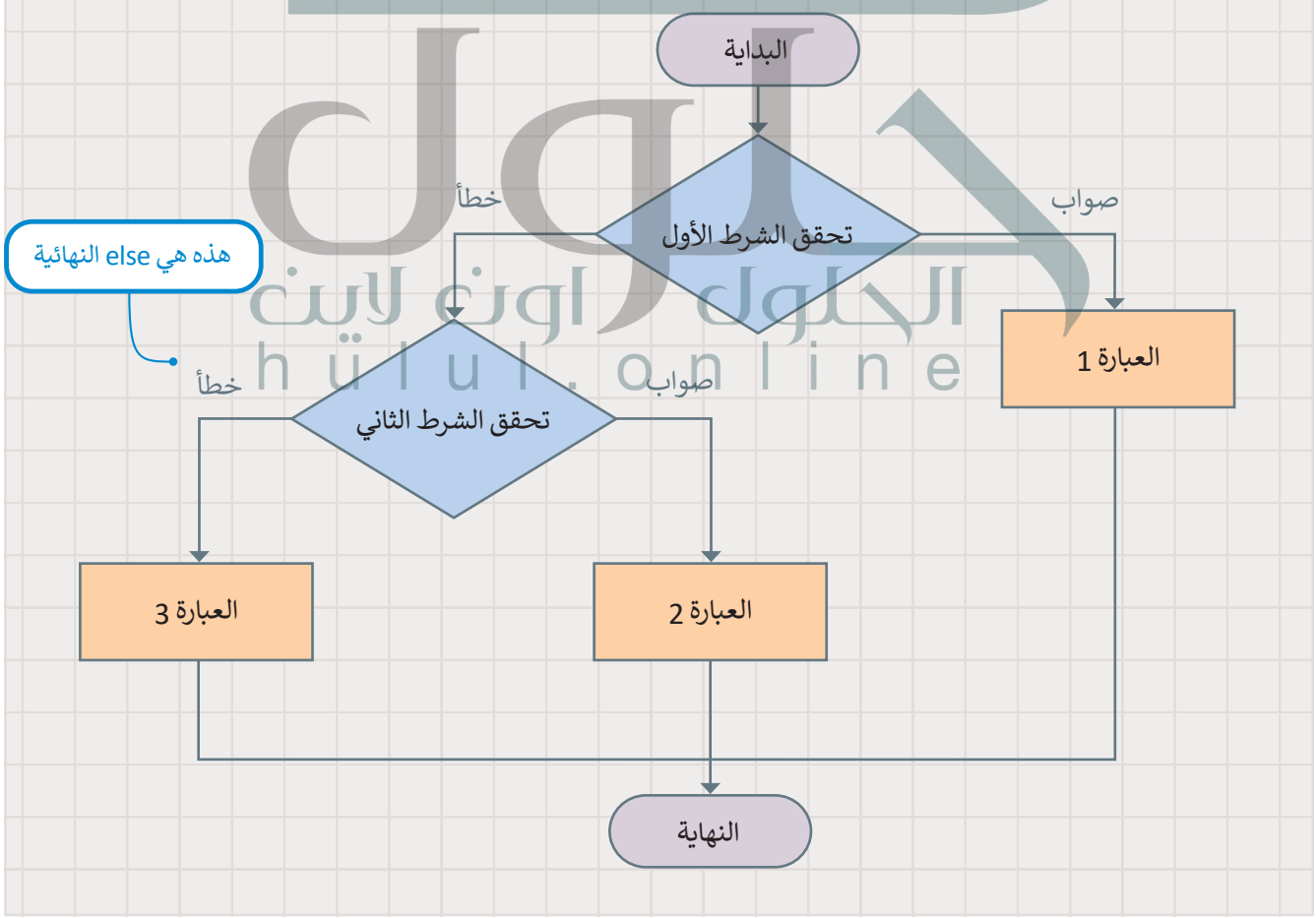
b أكبر من أو يساوي a

جملة if...elif

في الجمل الشرطية السابقة كان على المستخدم اختيار أحد خيارين، أما في هذا النوع من جمل if الشرطية، فإن المستخدم يجب أن يحدد خيارًا من بين خيارات متعددة. تنفذ عبارات if من الأعلى إلى الأسفل.

يتحقق البرنامج من الشروط واحدًا تلو الآخر، فإذا تحقق أحد الشروط، تنفذ العبارة تحت هذا الشرط ويتجاوز باقي الشروط، اما إذا لم يتحقق أي من الشروط، فستنفّذ جملة **else** النهائية.

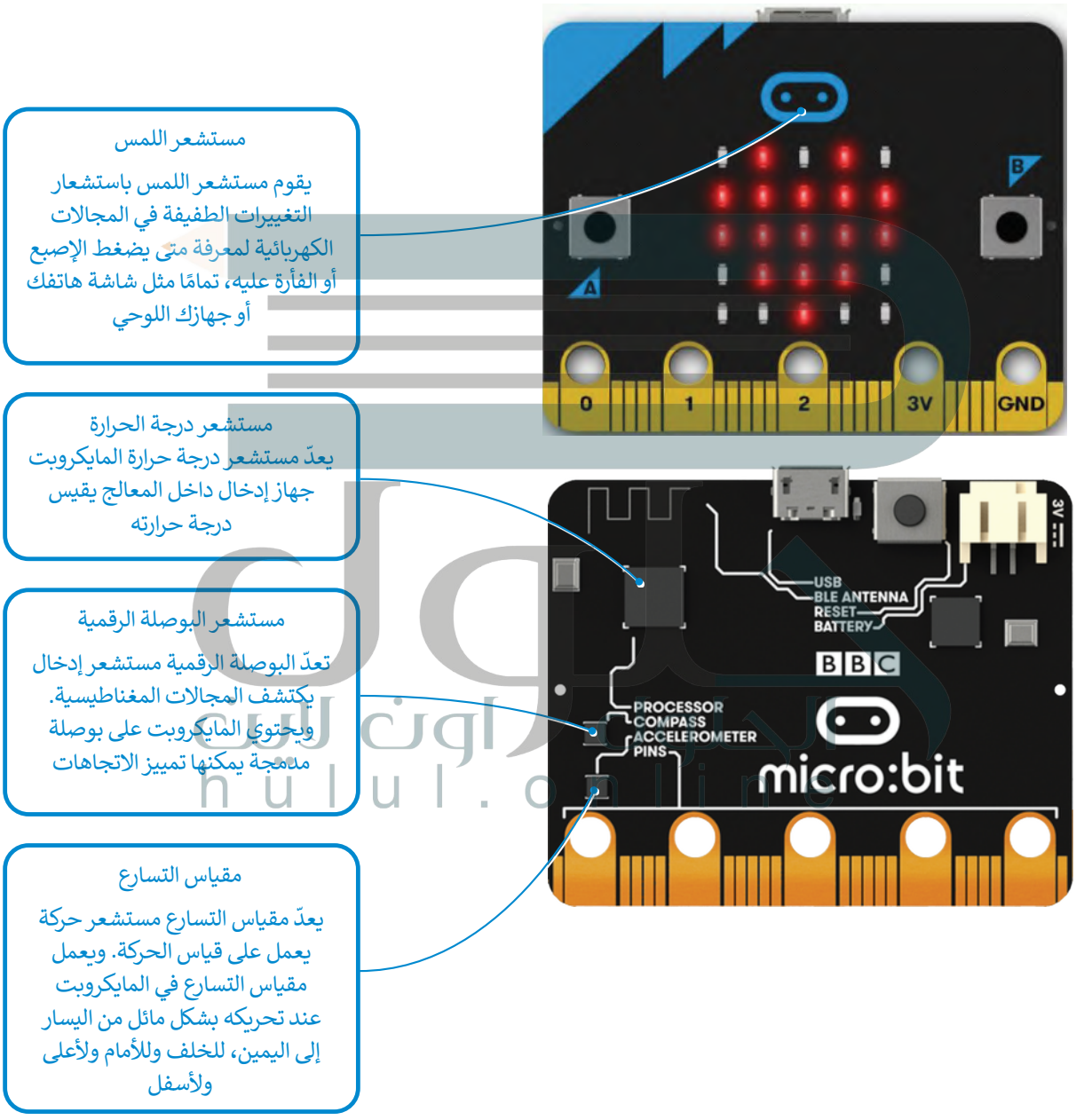




الإدخال

لقد تعلمت حتى الآن كيفية تعيين قيم لمتغيرات البرنامج. هناك طريقة أخرى لتعيين قيمة متغير وهي الحصول على بيانات الإدخال والمعلومات من بيئة الجهاز الذي تبرمجه. يقدم بايثون فئة **إدخال** (Input) حيث يمكنك العثور على أوامر الإدخال، عند استدعاء إحدى هذه الدوال، يتوقف البرنامج وينتظر إدخال البيانات، من الأمثلة على البيانات المدخلة الضغط على زر معين.

تستخدم جمل if الشرطية المدخلات كشروط. تتضمن مدخلات مايكروبت أحداثًا وبيانات من أجهزة الاستشعار والأزرار المختلفة.



تطبيقات المستشعرات في الحياة

مستشعر اللمس

تُستخدم مستشعرات اللمس بشكل كبير كبديل للمفاتيح الآلية رغم أن لها استخدامات أخرى متعددة. ويمكن ملاحظة التطبيقات الأكثر شيوعًا لمستشعرات اللمس في صناعة الإلكترونيات الاستهلاكية التي تشمل: أجهزة الحاسب، والهواتف المحمولة، والأجهزة الطرفية، والأجهزة المنزلية، وأنظمة قفل الأبواب، ووحدات التحكم في الألعاب، فقد كان هذا القطاع من أولى القطاعات التي شهدت انتشارًا عالميًا. وهناك مجال آخر تُستخدم فيه مستشعرات اللمس بصورة متزايدة وهو مجال صناعة السيارات، فالمفاتيح الذكية، ومفاتيح التحكم، وأجهزة التحكم عن بعد، والشاشات التي تعمل باللمس تعدّ ميزات أساسية في السيارات ذات التقنية الحديثة.

مستشعر الحرارة

تُستخدم مستشعرات الحرارة في العديد من الأجهزة الكهربائية داخل المنازل مثل: الثلاجات للمساعدة على تنظيم درجات الحرارة الباردة والحفاظ عليها وتُستخدم كذلك داخل المواقد والأفران لضمان ارتفاع درجة حرارتها إلى المستويات المطلوبة للطبخ أو التدفئة. وتستخدم أيضًا في مبرد المركبات للتحذير عندما ترتفع درجة حرارة المحرك بشكل خطير، إضافة إلى استخدامها في نظام التحكم بالمناخ داخل السيارة. بالإضافة إلى ذلك، تعتمد الدوائر المتكاملة على مستشعرات درجة حرارة السيليكون المدمجة في وحدات التحكم الدقيقة والإلكترونيات الأخرى. ويمكن العثور على هذه المستشعرات في مجموعة كبيرة من الأجهزة الإلكترونية مثل: أجهزة الحاسب المكتبية، والمحمولة، واللوحية، والهواتف المحمولة وغيرها من الأجهزة الإلكترونية الأخرى.

مستشعر البوصلة الرقمية

يعدّ مستشعر البوصلة الرقمية الجهاز الأكثر فاعلية في التنقل وتحديد الموقع والتعرف على الاتجاهات، وهو مفيد جدًا للرحالة في العثور على اتجاهاتهم، كما يُستخدم في الملاحة الجوية والتطبيقات العسكرية والروبوتات الخاصة بالمركبات ذاتية القيادة. هناك العديد من التطبيقات المتاحة والخاصة بمستشعرات البوصلة الرقمية لنظام أندرويد. على سبيل المثال: أثناء استخدام **نظام تحديد المواقع العالمي** (GPS) على الهواتف الذكية يمكن استخدم مستشعر البوصلة الرقمية الخاص بها لتحديد جهة الشمال والتدوير التلقائي لخريطة جوجل وفقًا لاتجاهها على أرض الواقع.

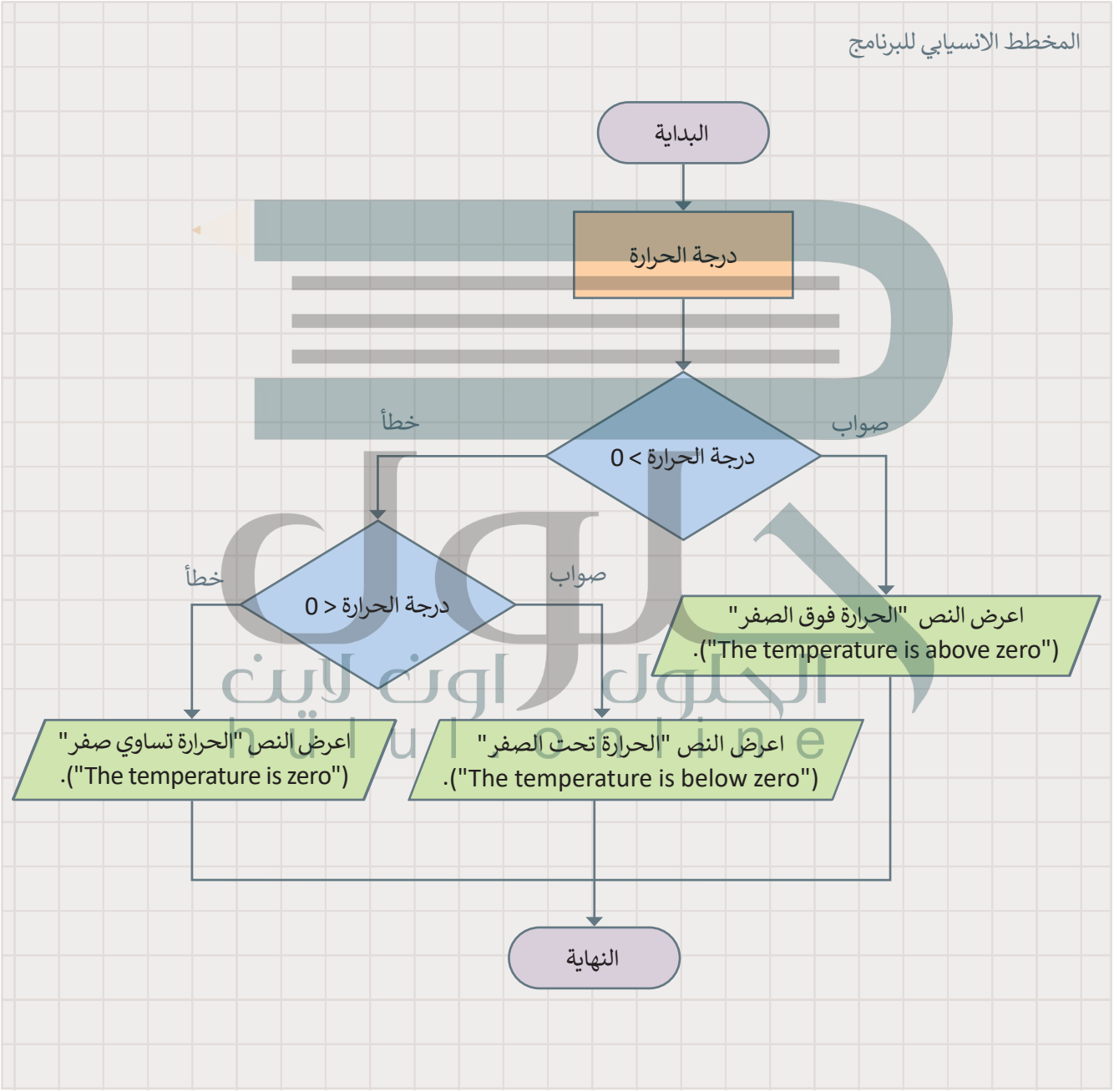
مستشعر مقياس التسارع

يمكن لمستشعرات مقياس التسارع قياس التسارع، أو السرعة، أو الإمالة، أو الاهتزاز أو الصدمة. ويستخدم هذا النوع من المستشعرات في أنظمة التثبيت. فعلى سبيل المثال: تستخدم الطائرات بدون طيار مقياس التسارع حيث يتكون من مستشعر الحركة القائم على المحور لتحديد اتجاهها والقدرة على الطيران بثبات. يستخدم مقياس التسارع أيضًا ضمن عوامل الأمان في أجهزة الحاسب المحمولة داخل الأقراص الصلبة. فعلى سبيل المثال: إذا سقط الحاسب المحمول فجأة أثناء استخدامه، فسيكتشف مقياس التسارع هذا السقوط المفاجئ ويوقف محرك القرص الصلب على الفور لتجنب حدوث أي تلف. تحتوي بعض الهواتف الذكية وأجهزة الحاسب اللوحية وغيرها من الأجهزة على مقياس تسارع للتحكم في واجهة المستخدم، حيث تُستخدم لتغيير وضع الشاشة أفقيًّا أو رأسيًّا بناءً على طريقة حمل الجهاز.

مثال برمجي: درجة الحرارة

يكتشف الأمر **حرارة** ((°) temperature) درجة الحرارة المحيطة ويقيسها بالدرجة المئوية. يحدد مايكروبت درجة الحرارة المحيطة من خلال فحص درجة حرارة المعالج. ونظرًا لأن درجة حرارة مايكروبت لا تكون مرتفعة في العادة، فإن درجة حرارة وحدة المعالجة المركزية عادةً ما تكون قريبة من درجة الحرارة في أي مكان محيطٍ بها.

لتلقِ نظرة على بعض الأمثلة باستخدام جمل **if ... elif**.



قد ترتفع درجة حرارة المايكروبت قليلًا عند عمله لمدة طويلة.

للتحقق من درجة الحرارة:

> من فئة **Logic** (المنطق)، اسحب وأفلت دالة **if**. 1

> من فئة **Input** (الإدخال)، اسحب وأفلت أمر **temperature (°)** (درجة الحرارة (درجة مئوية)) ، كشرط في جملة **if** واكتب **0 <**. 2

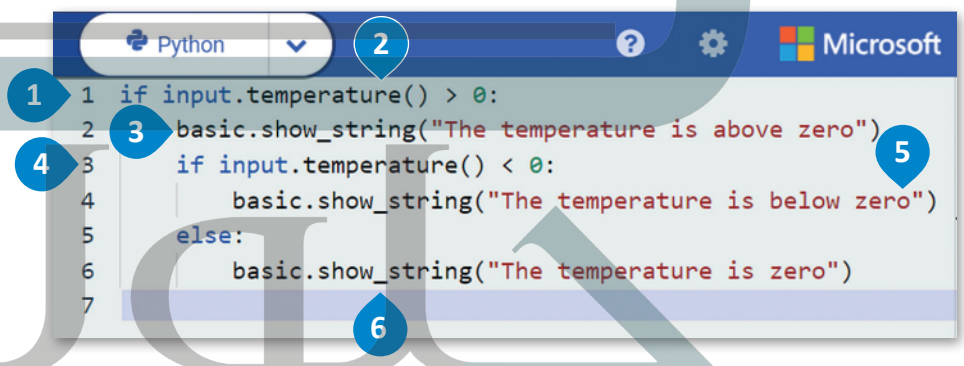
> من فئة **Basic** (أساسي)، اسحب وأفلت أمر **show string** (إظهار السلسلة) وعيّن النص إلى **"The temperature is above zero"** ("الحرارة فوق صفر"). 3

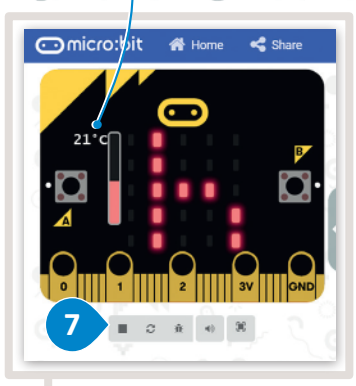
> من فئة **Logic** (المنطق)، اسحب وأفلت الأمر **if else** وعيّن **input.temperature ( ) <0** (إدخال.الحرارة)كشرط لها. 4

> من فئة **Basic** (أساسي)، اسحب وأفلت الأمر **show string** (إظهار السلسلة) وعيّن النص إلى **"The temperature is below zero"** ("الحرارة أقل من صفر"). 5

> من فئة **Basic** (أساسي)، اسحب وأفلت الأمر **show string** (إظهار السلسلة) وعيّن النص إلى **"The temperature is zero"** ("الحرارة هي صفر"). 6

> اضغط على تشغيل لمعاينة النتيجة. 7



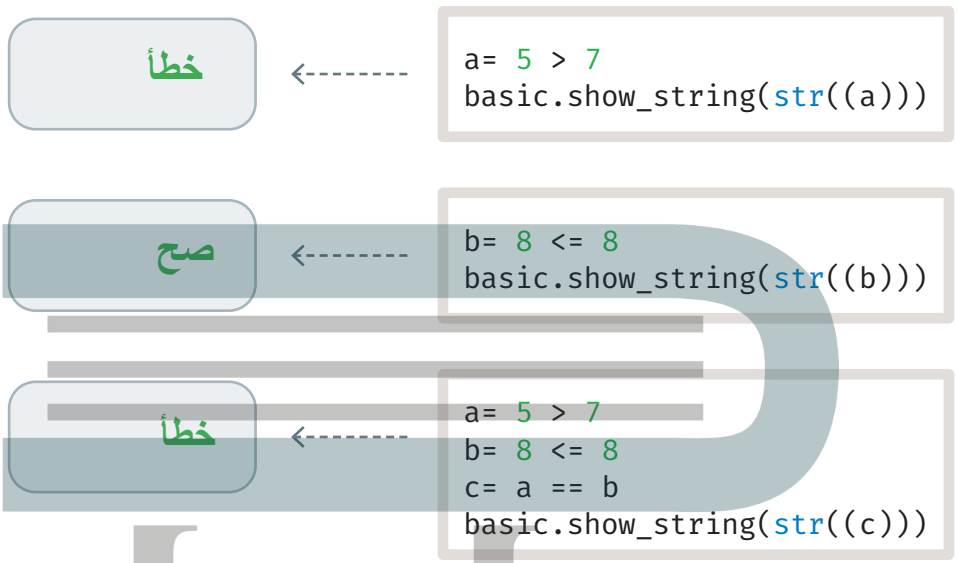


الحرارة فوق صفر

لنطبق معًا

تدريب 1

هل الأوامر التالية صحيحة أم خطأ؟



تدريب 2

املأ الفراغات في الجمل التالية بالكلمات المناسبة مما يلي، ويمكنك استخدام بعض الكلمات عدة مرات: تجاوز، True، False، تنفيذ، واحدًا تلو الآخر، else، الشرط.

1. في عبارة if:
إذا كان الشرط ___true___، فستنفذ العبارة (العبارات) التي تلي if. إذا كان ___الشرط___ هو ___false___، فلن تنفذ العبارة (العبارات).

2. في عبارة if ... else:
إذا كان الشرط ___true___، فستنفذ العبارة (العبارات) التي تلي if. إذا كان الشرط ___false___، فستنفذ العبارة (العبارات) الموجودة ضمن ___else___.

3. في عبارة if ... elif:
يتحقق البرنامج من الشروط ___واحداً تلو الآخر___، إذا كان أحد الشروط ___true___، فسيتم ___تنفيذ___ العبارة ضمن هذا الشرط.
سيتم ___تجاوز___ بقية العبارات. إذا لم يكن أي من الشروط ___true___، فستنفذ عبارة else النهائية.

تدريب 3

ما الذي يحدث عند تشغيل البرنامج التالي؟ اختر الإجابة الصحيحة.

```
number = 12
if number > 0 :
    basic.show_string("positive number")
```

- ◯ لن يعمل البرنامج لأن صيغة الأوامر غير صحيحة.
- ◯ لن تُعرض أي رسالة على الشاشة لأن المتغير لم يتم تعريفه.
- ✓ ستعرض الرسالة " positive number" (" رقم موجب") على الشاشة.

```
number = -10
if number < 0
    basic.show_string(negative number)
```

- ✓ لن يعمل البرنامج لأن صيغة الأوامر غير صحيحة.
- ◯ لن تُعرض أي رسالة على الشاشة لأن الشرط غير صحيح.
- ◯ ستعرض الرسالة " ative number

تدريب 4

أنشئ برنامجًا يسألك عن درجات
غير مجتاز.

```
grade = 0

basic.show_string("Enter your test score")

def on_button_pressed_a():
    global grade
    grade += 1
    basic.show_number(grade)
input.on_button_pressed(Button.A, on_button_pressed_a)

def on_button_pressed_ab():
    if grade >= 15:
        basic.show_string("Excellent")
    if grade <= 10:
        basic.show_string("Failed")
    else:
        basic.show_string("Good")
input.on_button_pressed(Button.AB, on_button_pressed_ab)
```

تدريب 5

شغّل البرنامج التالي وصِفْ وظيفته.

```
def on_forever():
    if input.button_is_pressed(Button.A):
        basic.show_icon(IconNames.HAPPY)
    else:
        basic.show_icon(IconNames.CONFUSED)
basic.forever(on_forever)
```

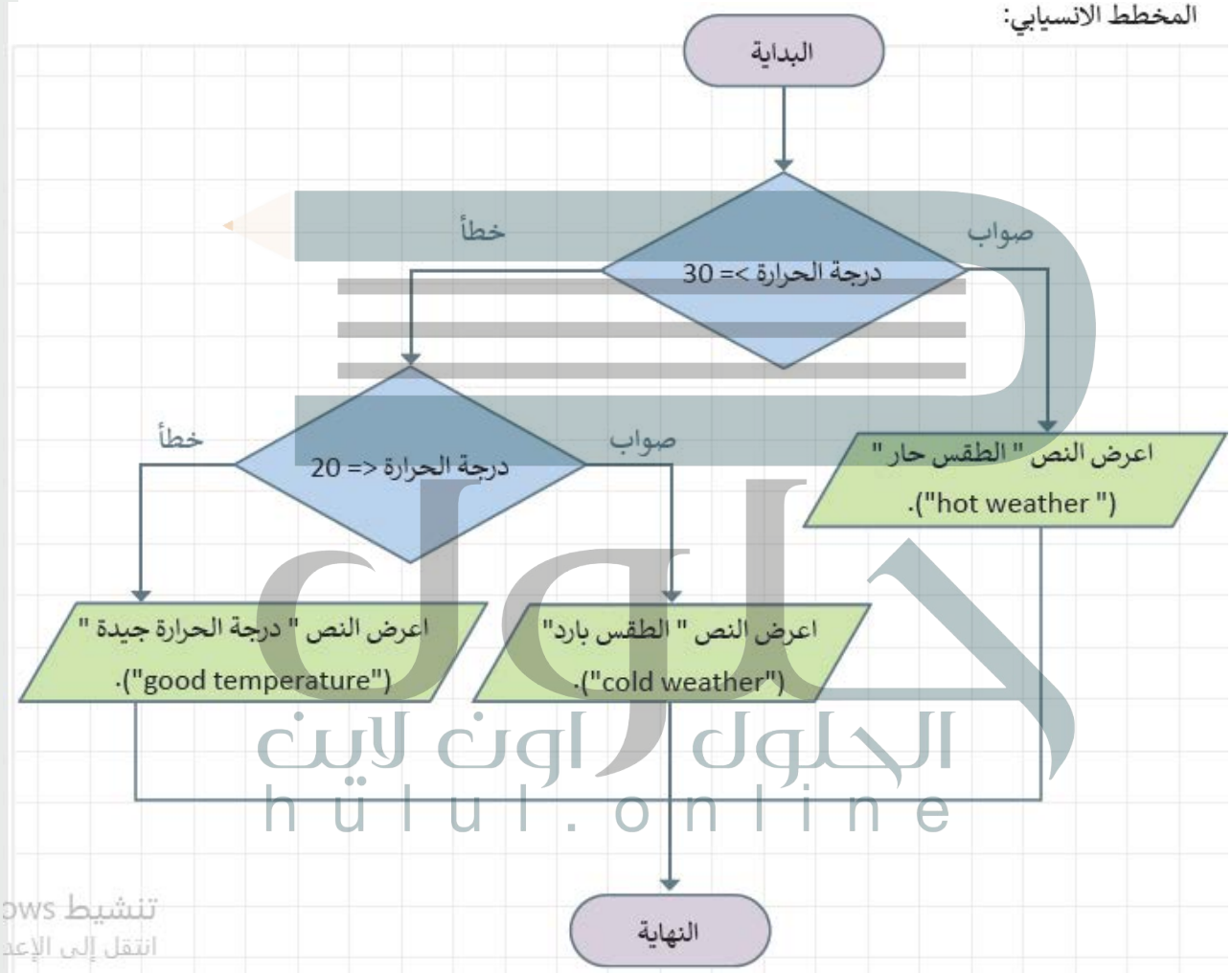
عند الضغط على زر التشغيل سيتم عرض أيقونة مرتبك على شاشة led باستمرار وعند الضغط على الزر A فإنه يتم عرض أيقونة سعيد على شاشة LED وعندما نحرر الزر A ستختفي أيقونة سعيد وسيتم عرض أيقونة مرتبك على شاشة LED مرة أخرى

تدريب 6

ارسم مخططًا انسيابيًا لبرنامج يقيس درجة حرارة البيئة المحيطة ثم أنشئ البرنامج:

- إذا كانت درجة الحرارة تساوي أو تزيد عن 30، فسَمِّ الرسالة **"hot weather"** (" الطقس حار").
- إذا كانت درجة الحرارة متساوية أو أقل من 20، فسَمِّ الرسالة **"cold weather"** ("الطقس بارد").
- إذا كانت درجة الحرارة بين 30 و 20 ، فسَمِّ الرسالة **"good temperature"** ("درجة الحرارة جيدة").

المخطط الانسيابي:



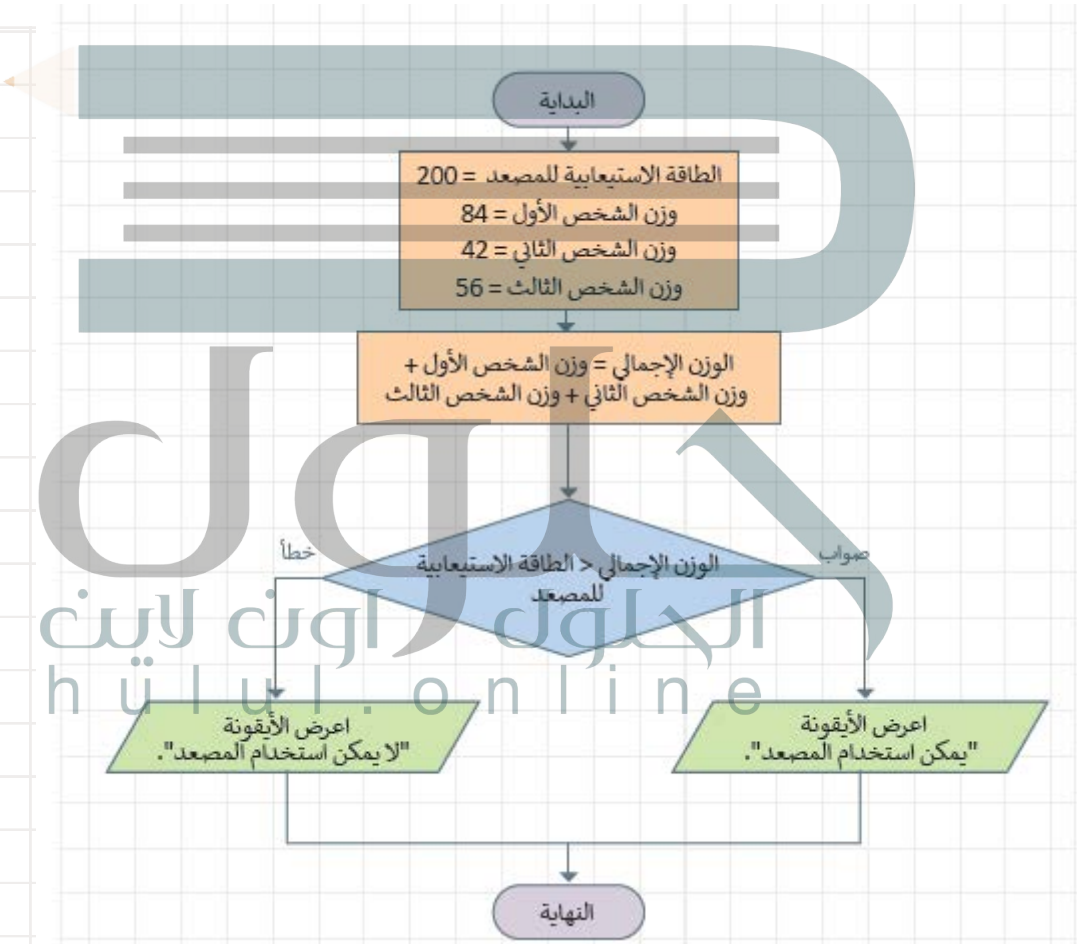

```
if input.temperature() >= 30:
    basic.show_string("hot weather")
if input.temperature() <= 20:
    basic.show_string("cold weather")
else:
    basic.show_string("good temperature")
```

تدريب 7

ارسم مخططًا انسيابيًا لبرنامج يتحقق مما إذا كان بإمكان ثلاثة أشخاص استخدام المصعد في نفس الوقت ثم أنشئ البرنامج:

- حدد الطاقة الاستيعابية للمصعد.
- حدد وزن كل شخص.
- احسب الوزن الإجمالي للأشخاص الثلاثة.
- أجرِ الفحص واعرض الرسالة المناسبة.

المخطط الانسيابي:




```
elevator_limit = 200
person1_w = 84
person2_w = 42
person3_w = 56
total_weight = person1_w + person2_w + person3_w
if total_weight < elevator_limit:
    basic.show_string("You can use the elevator")
else:
    basic.show_string("The elevator cannot be used")
```

رابط الدرس الرقمي

www.ien.edu.sa

مشروع الوحدة

أنشئ برنامجًا يقوم بحساب مربع سلسلة من الأرقام. مع العلم أنه يجب حساب مربع الأرقام على النحو التالي:

$1^2 = 1$ ، $2^2 = 4$ ، $3^2 = 9$ ، ... ، $N^2 = ...$

1. البرنامج سينفذ الآتي:
2. تعريف المتغير N.
3. إسناد قيمة للمتغير N.
4. اضبط قيمة المتغير N، للتحكم في قيمة المتغير N استخدم أزرار المايكروبت، ولزيادة قيمة المتغير استخدم الزر A ولتقليله استخدم الزر B.
5. اضبط قيمة المتغير N، للتحكم في قيمة المتغير N استخدم أزرار المايكروبت، ولزيادة قيمة المتغير استخدم الزر A ولتقليله استخدم الزر B.
6. عند الضغط على الزرين A + B:

.....

> احسب تسلسل مربع المتغير N (على سبيل المثال: إذا كان N=3 فسيكون تسلسل مربعه هو الأرقام 1، 4، 9).

.....

> اعرض تسلسل مربع المتغير N على شاشة المايكروبت.

7. شغّل البرنامج وتحقق من عدم وجود أي خطأ.

في الختام

جدول المهارات

المهارة	درجة الإتقان	
	أتقن	لم يتقن
1. إنشاء برنامج باستخدام مايكروسوفت ميك كود.		
2. إنشاء الأكواد بالتعامل مع المتغيرات.		
3. إجراء العمليات الرياضية باستخدام مايكروسوفت ميك كود بايثون.		
4. إنشاء الأكواد باستخدام جُمل التكرار.		
5. إنشاء الأكواد بتطبيق العوامل الشرطية المختلفة.		
6. إنشاء الأكواد لاتخاذ القرارات في مايكروبت بلغة بايثون.		

المصطلحات

زر	Button	المتحكم الدقيق	Microcontroller
معاملة شرطية	Conditional statement	برنامج	Program
الإدخال	Input	التكرارات	Repetition
شاشة LED	LED screen	مستشعر	Sensor
تكرار	Loop	متغير	Variable