عرض بوربوينت تقديمي للدرس الثاني المتغيرات والتكرارات من الوحدة الرابعة البرمجة بواسطة المايكروبت لمقرر التقنية الرقمية

تم تحميل هذا الملف من موقع المناهج السعودية	
موقع المناهج ⇦ المناهج السعودية ⇦ الصف الأول ⇦ المهارات الرقمية ⇦ الفصل الثاني ⇦ عروض بوربوينت ⇦ الملف	
تاريخ إضافة الملف على موقع المناهج: 2026-01-08 00:50:09	
ملفات ا كتب للمعلم ا كتب للطالب ا اختبارات الكترونية ا اختبارات ا حلول ا عروض بوربوينت ا أوراق عمل منهج انجليزي ا ملخصات وتقارير ا مذكرات وبنوك ا الامتحان النهائي ا للمدرس	المزيد من مادة المهارات الرقمية:
إعداد: نجود دحمان	

التواصل الاجتماعي بحسب الصف الأول					
صفحة المناهج السعودية على فيسبوك					
المواد على تلغرام	التربية الاسلامية	اللغة العربية	اللغة الانجليزية	الرياضيات	

المزيد من الملفات بحسب الصف الأول والمادة المهارات الرقمية في الفصل الثاني	
1	عرض بوربوينت تقديمي للدرس الثاني المتغيرات والتكرارات من الوحدة الرابعة البرمجة بواسطة المايكروبت لمقرر التقنية الرقمية

مقرر التقنية الرقمية ١-٣ أول ثانوي

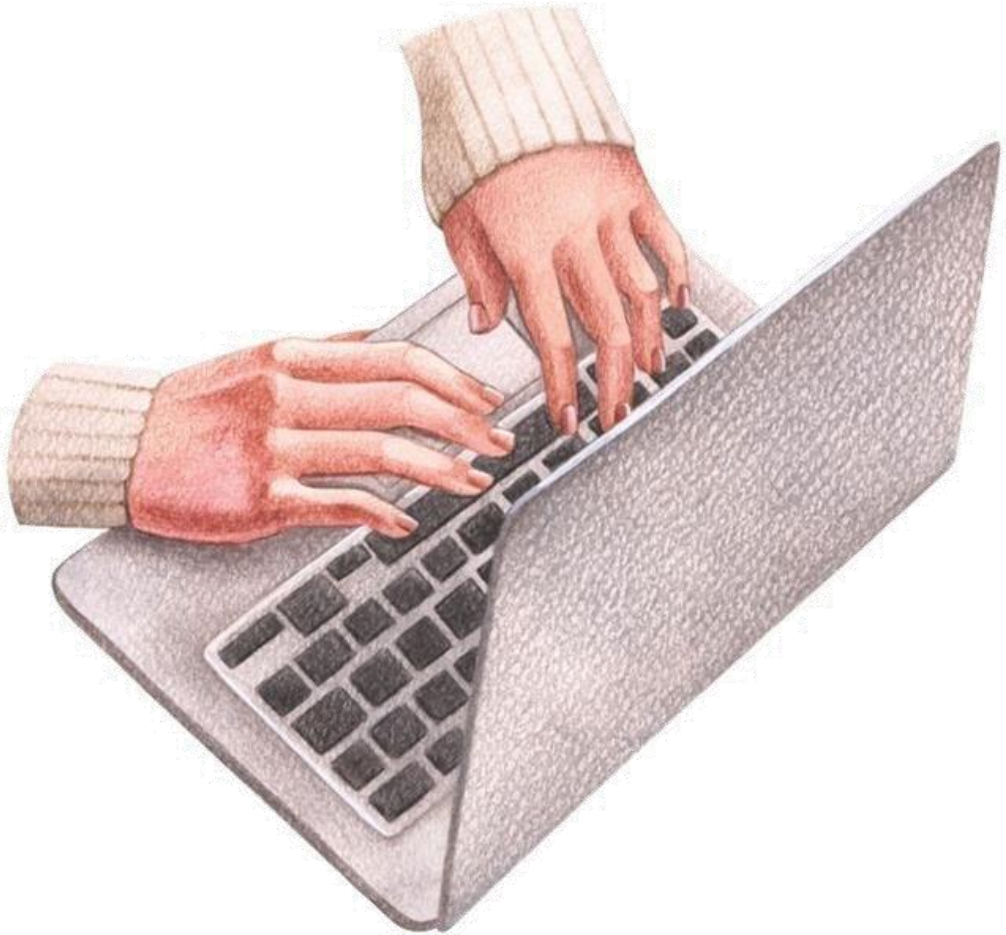
الفصل الدراسي الثالث

المعلمة

نجود دحمان

النشيد الوطني

محتويات المنهج

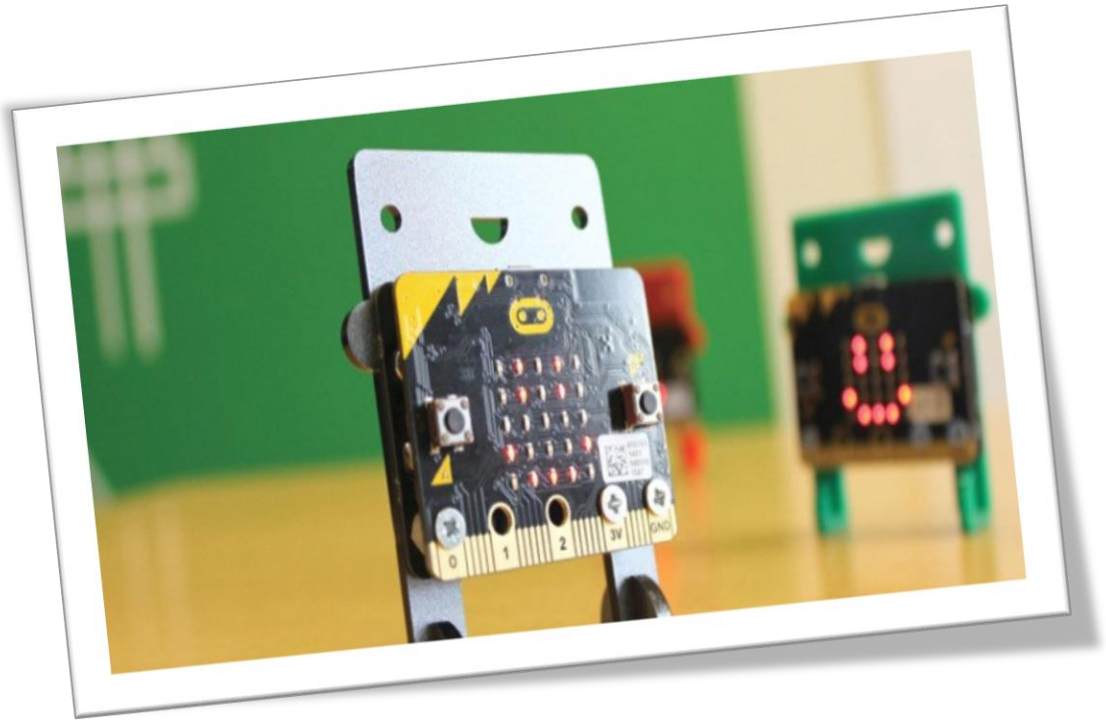


١ مستندات ونماذج وتقارير الأعمال

٢ الشبكات

٣ البرمجة بواسطة المايكروبت

المادة : التقنية الرقمية ١-٣	التاريخ : / ١١ /	الوحدة الثالثة :البرمجة بواسطة المايكروبت



الوحدة الثالثة

البرمجة بواسطة المايكروبت

الدروس السابقة

١	and يعتبر من أسماء المتغيرات التي لا يمكن استخدامه	
٢	المتغيرات المحلية يتم تعريفها خارج أي دالة ويمكن الوصول إليها من أي مكان في البرنامج	
٣	يتم تعريف الدالة باستخدام الأمر if	

محتويات الوحدة

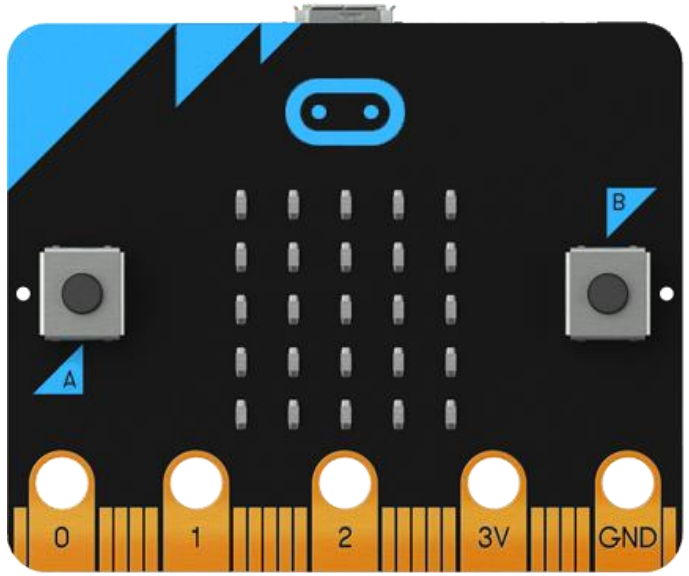
- مقدمة إلى المايكروبت .
- المتغيرات والتكرارات . 
- اتخاذ القرارات .
- المشروع

التقويم قبلي

استراتيجية النقاش والحوار

- هل يمكن إجراء أي عمليات حسابية باستخدام لغة البايثون؟
- هل هناك اختلاف في صيغة كتابة العمليات الحسابية بين البرمجة و العمليات الرياضية ؟

المادة : التقنية الرقمية ١-٣	التاريخ : / ١١ /	الوحدة الثالثة :البرمجة بواسطة المايكروبت



الدرس الثاني

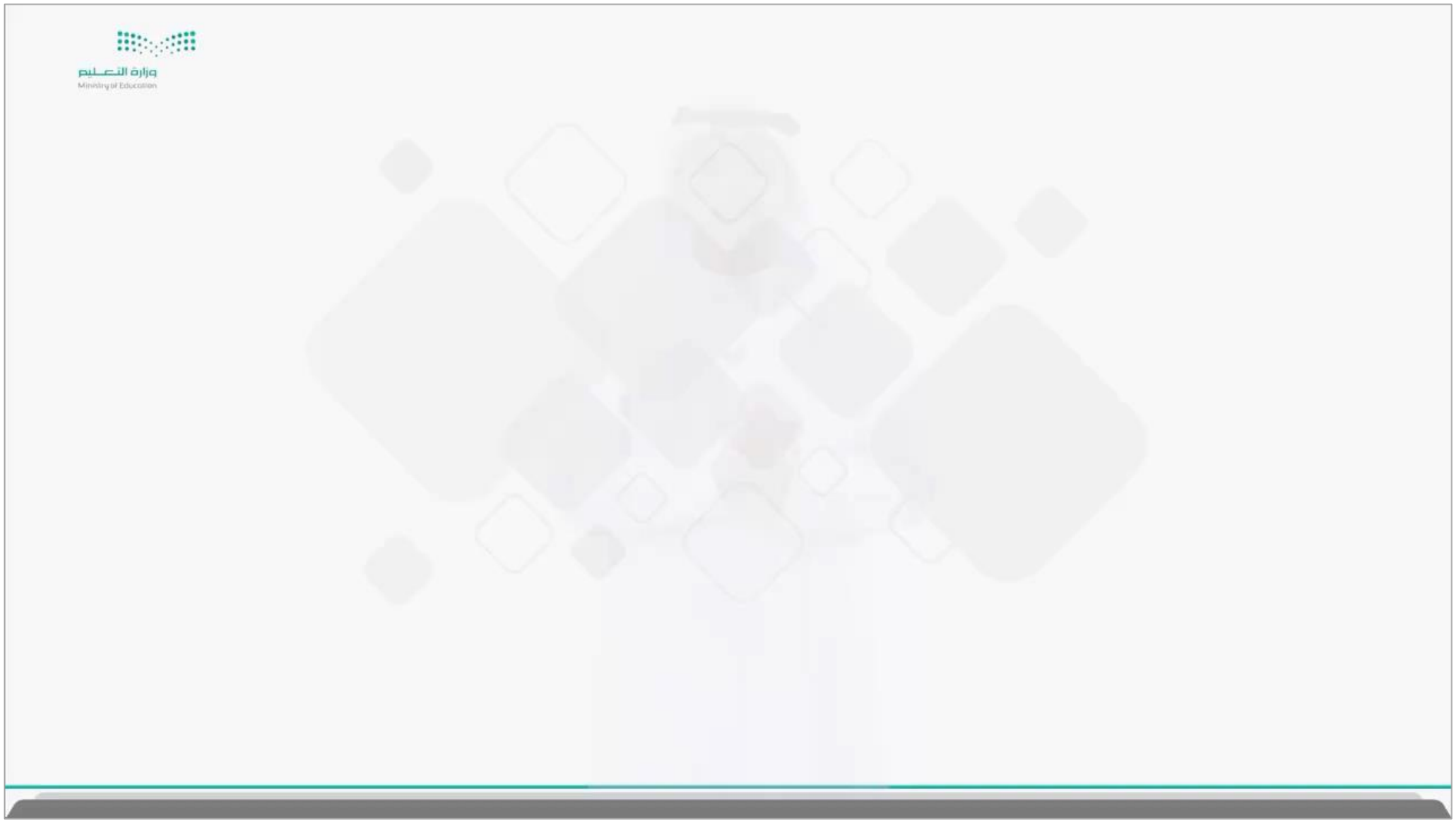
المتغيرات والتكرارات

أهداف الدرس الجزء الأول

ستتعلم في هذا الدرس

- ❑الحسابات والأرقام.
- ❑الإحداثيات في بايثون.
- ❑أوامر اللعب.
- ❑التكرارات.

فيديو تعريفي للمتغيرات والتكرارات



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

يمكن استخدام لغة البايثون لإجراء أي نوع من العمليات الرياضية

(جمع – طرح – ضرب – قسمة)

تكتب العمليات الرياضية في البرمجة

بطريقة مختلفة عن التي تكتب بها في العمليات الرياضية (الحسابية)

العملية الحسابية	بلغة بايثون	رياضياً
الجمع	4+2	4+2
الطرح	4-2	4-2
الضرب	4*2	4×2
القسمة	4/2	٤÷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 :

يتم تنفيذ عوامل التشغيل بالترتيب من اليسار إلى اليمين.

```
X = a * * 2 + 2 * a * b + b * * 2
```

يُحدد ترتيب العمليات في بايثون سابقاً 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 مثلاً :

أن ناتج 4+2*5 هو 14 وليس 30

يتم تنفيذ العمليات من اليسار الي اليمين التي تكون في نفس مستوى الترتيب.

يمكن العثور على العمليات الرياضية في مايكروسوفت ميك كود من خلال فئة حساب (Math).

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

التطبيق العملي

الدَّرس الثَّاني: المتغيِّراتُ والتَّكراراتُ


لإضافة عملية الجمع

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

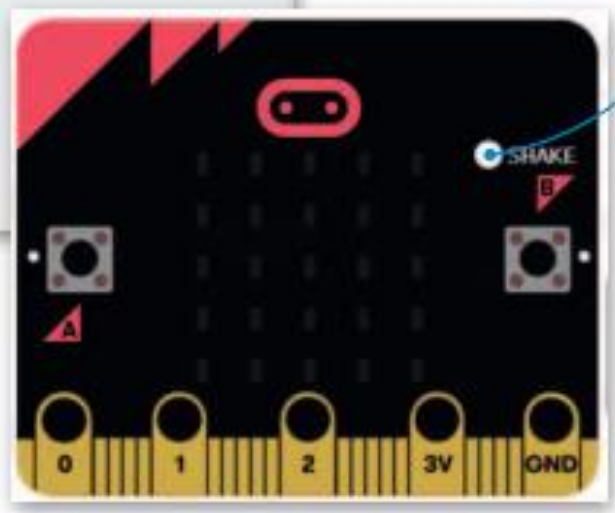
> من فئة **Math** (حساب)، اسحب وأفلت أمر الجمع داخل الجملة البرمجية ثم اكتب الأرقام التي تريد جمعها. 5

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

Python

```
1  add = 0
2
3  def on_gesture_shake():
4      global add
5      add = 5 + 10
6      basic.show_number(add)
7  input.on_gesture(Gesture.SHAKE, on_gesture_shake)
```

اضغط على زر (Shake) للتحقق من نتيجة المقطع البرمجي.



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


الإحداثيات في بايثون

الدَّرس الثَّاني: المتغيِّراتُ والتَّكراراتُ

يتم تمثيل مصابيح Led الموجودة مايكروبت على شكل شبكة إحداثيات بمحور أفقي سيني (X) ومحور عمودي صادي (Y).

تحتوي على 5 صفوف أفقية و 5 صفوف عمودية من المصابيح.

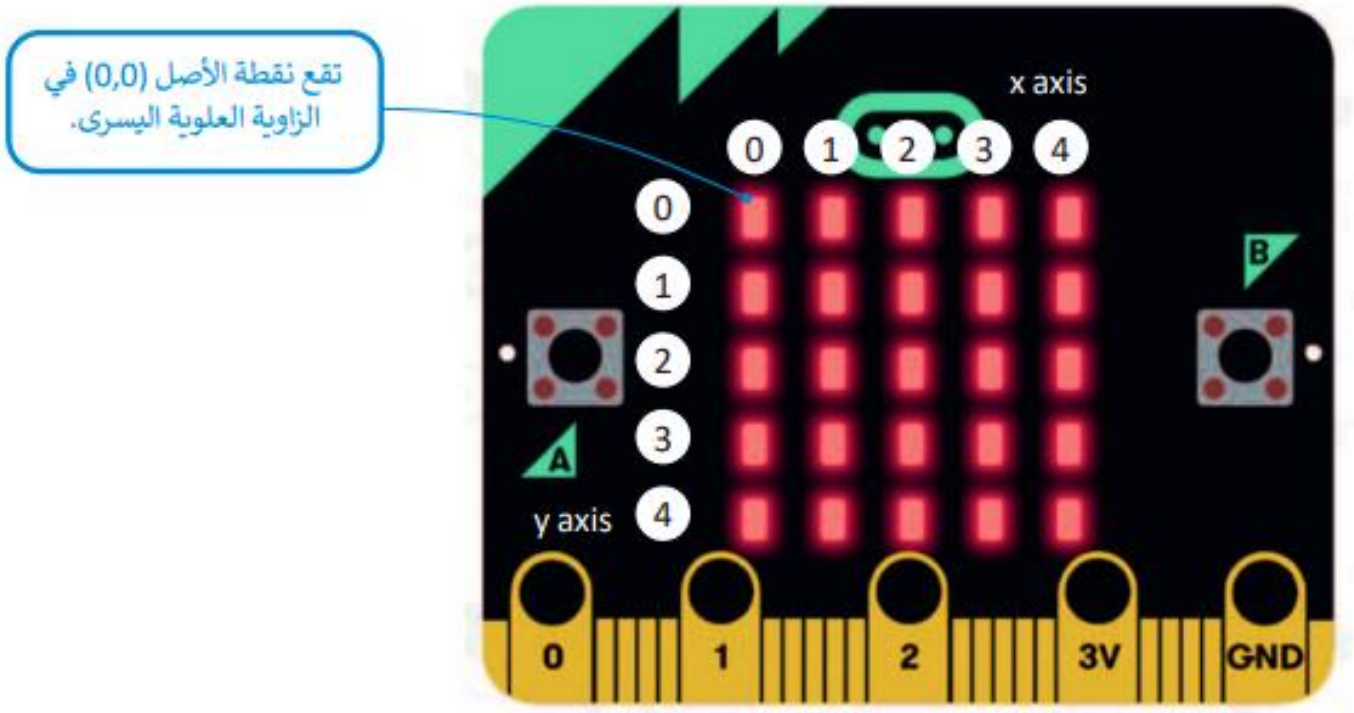
توجد النقطة (0،0) في الزاوية اليسرى العلوية (نقطة الأصل).

تتراوح قيم إحداثيات X بين (0 – 4)

وتزداد قيمتها من اليسار إلى اليمين.

تتراوح قيم إحداثيات Y بين (0 – 4)

وتزداد قيمتها 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 الاحداثيات.

ستنشئ مقطعاً برمجياً يتحرك فيه الكائن إلى اليسار عند الضغط على الزرA

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.


إنشاء الكائن الرسومي

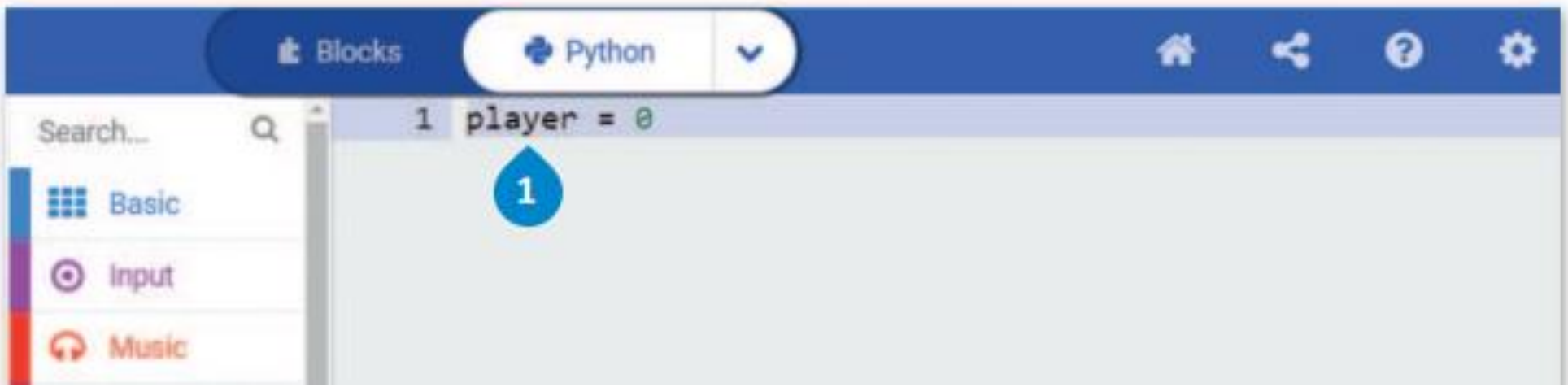
لإنشاء الكائن الرسومي:

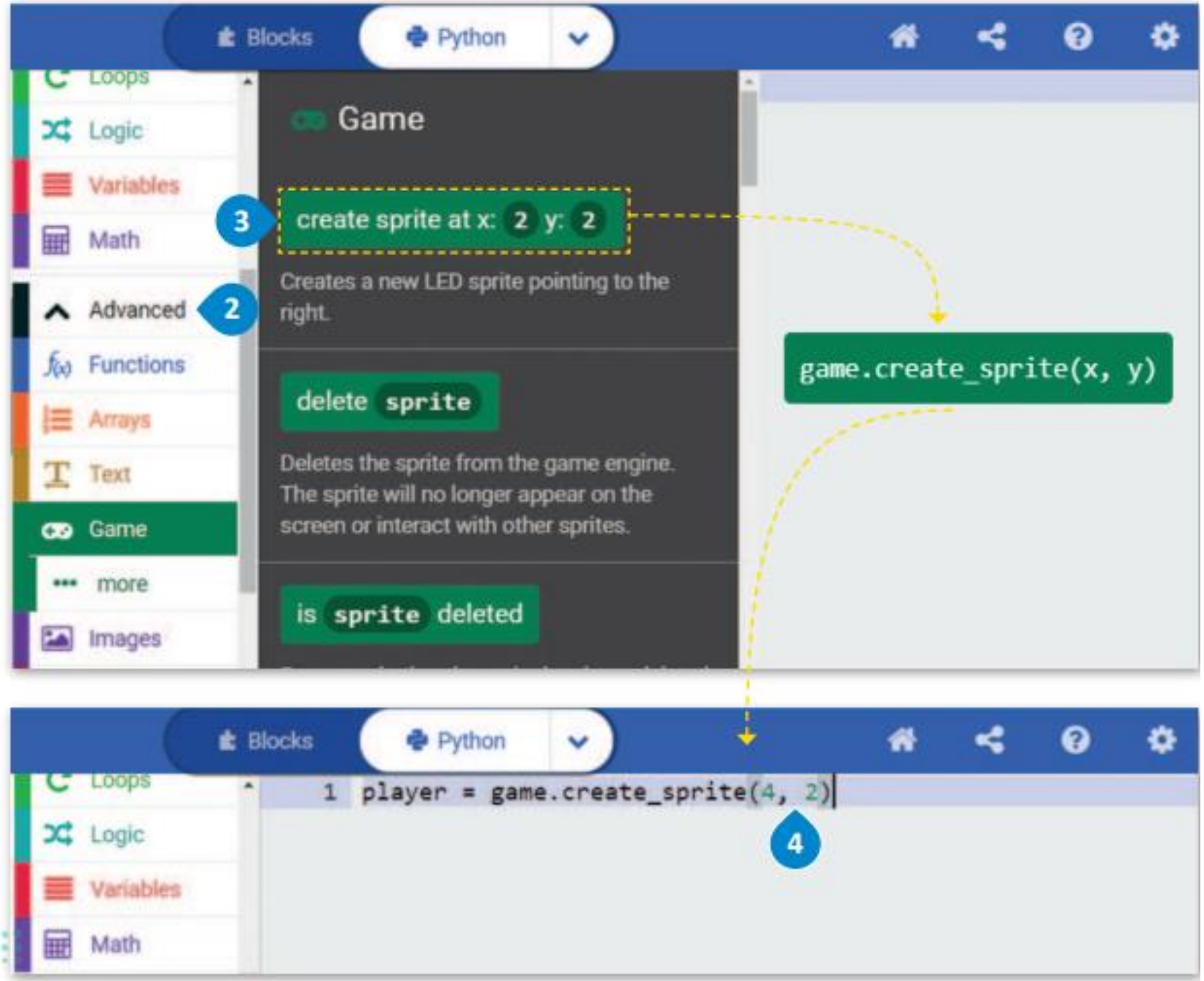
> من فئة **Variables** (متغيرات)، اسحب وأفلت أمر **item = 0** (العنصر = 0)، واكتب **player** (لاعب) على الجانب الأيسر. 1

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

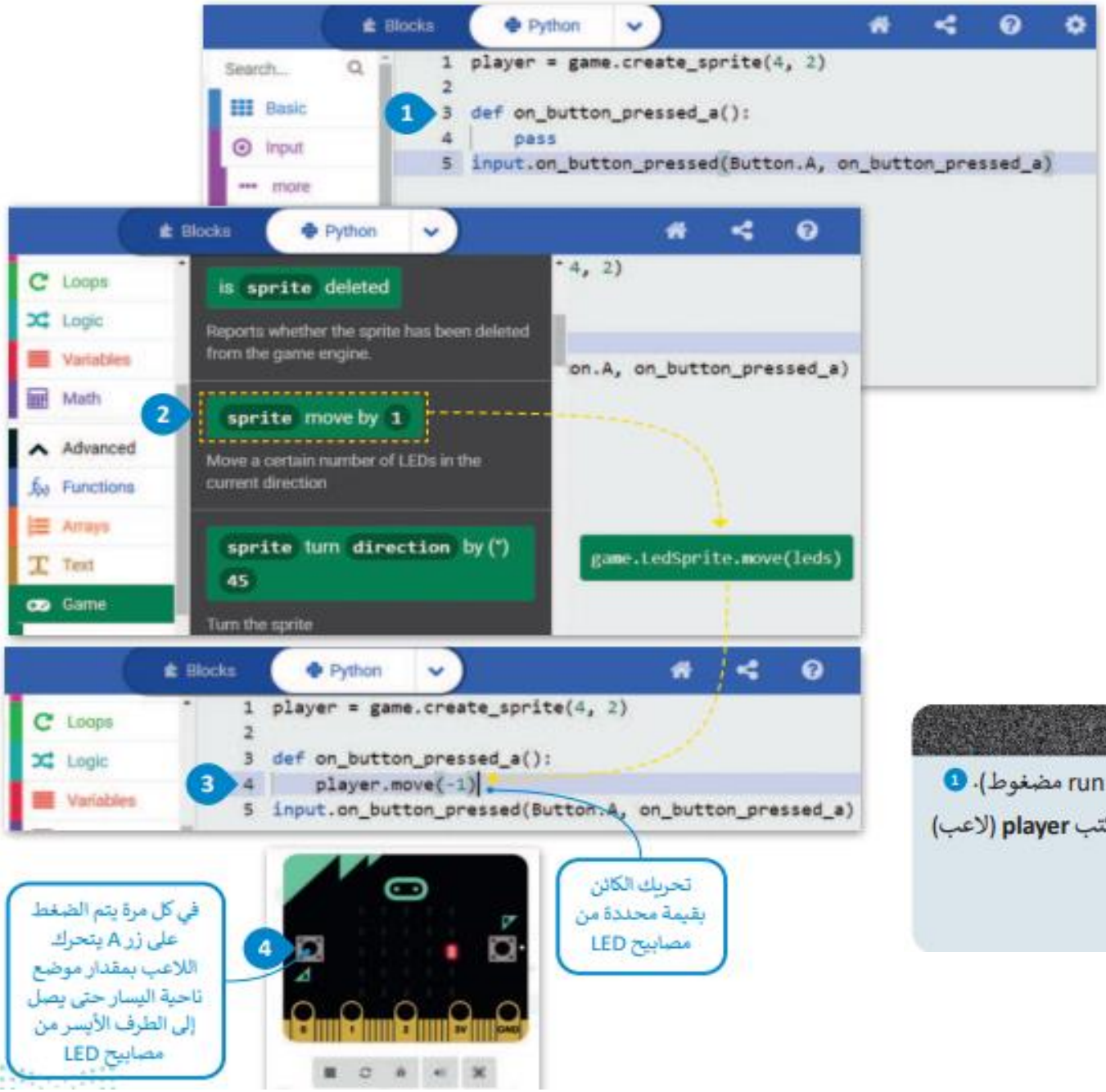
> من فئة **Game** (اللعبة)، اسحب وأفلت الأمر **create sprite at x:2 y:2** (إنشاء كائن رسومي في x:2 و y:2). 3

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




Blocks
Python
Loops
Logic
Variables
Math
Advanced
Functions
Arrays
Text
Game
more
Images
Game
3
create sprite at x: 2 y: 2
Creates a new LED sprite pointing to the right.
2
delete sprite
Deletes the sprite from the game engine. The sprite will no longer appear on the screen or interact with other sprites.
is sprite deleted
game.create_sprite(x, y)
Blocks
Python
Loops
Logic
Variables
Math
1 player = game.create_sprite(4, 2)
4


جعل الكائن الرسومي
يتحرك في شاشة Led



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

التقويم الختامي

تقويم ختامي

١	يتم تنفيذ عملية الضرب والقسمة قبل الجمع والطرح في البايثون	☑
٢	تكتب العمليات الحسابية في البرمجة بطريقة مختلفة عن التي تكتب بها في الرياضيات	☑
٣	يتم تعريف المتغيرات المحلية داخل دالة ولذا تنتمي فقط إلي هذه الدالة المحددة	☒

انتهى الجزء الأول
من الدرس☺

أهداف الدرس الجزء الثاني

ستتعلم في هذا الدرس

- ❑الحسابات والأرقام.
- ❑الإحداثيات في بايثون.
- ❑أوامر اللعب.
- ❑التكرارات.

التكرارات

تحتاج احياناً إلى تكرار جزء من البرنامج عدة مرات في البرمجة ، لهذا السبب فإن معظم لغات البرمجة توفر دوال مختلفة خاصة بالتكرارات البرمجية.

تسمح التكرارات بتنفيذ سطر واحد أو مجموعة من التعليمات البرمجية لعدة مرات. توفر بايثون عددا من أوامر التكرار التي تساعد على تجنب إعادة كتابة أوامر التعليمات البرمجية،

وتدعم بايثون نوعين من التكرارات:

- تكرار For
- تكرار While

وزارة التعليم
Ministry of Education

التكرارات في مايكروبت بلغة بايثون

لاحظ أنه يجب كتابة : بعد التعبير التكراري

```
for loop_variable in range(x):
    statements
```

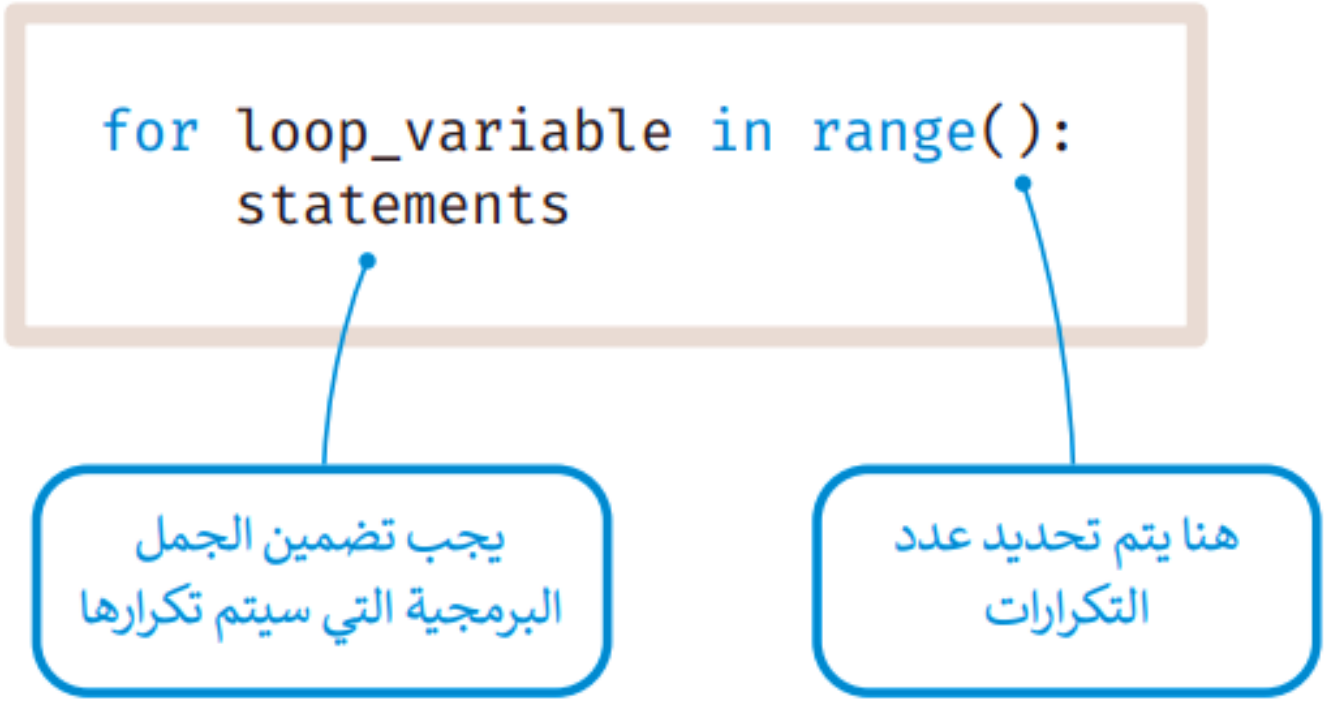
تكرار for

```
while condition:
      statements
```

تكرار while

تكرار For

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



الدَّرس الثَّاني: المتغيِّراتُ والتَّكراراتُ


إنشاء كائن رسومي للمطر

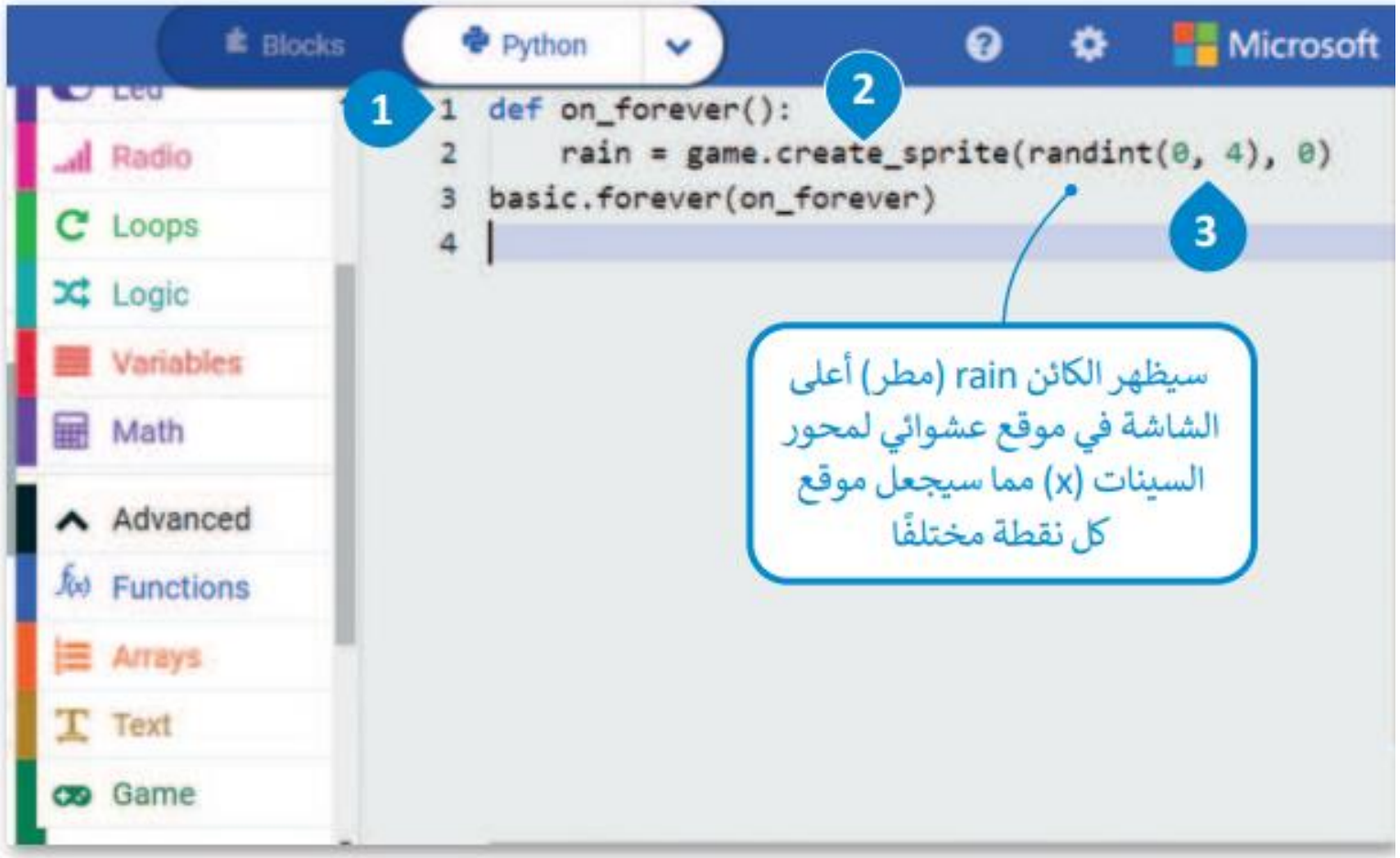
مثال برمجي : سقوط المطر

تعرف في الدرس السابق على مثال يحرك به اللاعب كائناً يساراً بالضغط على الزر A سترى في هذا المثال كيف يمكن تطبيق تكرار for لجعل الكائن يبدو كأنه يسقط من الأعلى .

ستنشئ مقطعاً برمجياً يمثل 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 المقاطع البرمجية الأخرى في نفس الوقت، حيث أن المقطع البرمجي الموجود داخل تكرار "للأبد" (forever) سينتج عن المقطع الآخر الموجود في برنامجك.


إنشاء الكائن الرسومي
باستخدام التكرارات

إنشاء الكائن الرسومي باستخدام التكرارات

لإنشاء الكائن الرسومي باستخدام التكرارات:

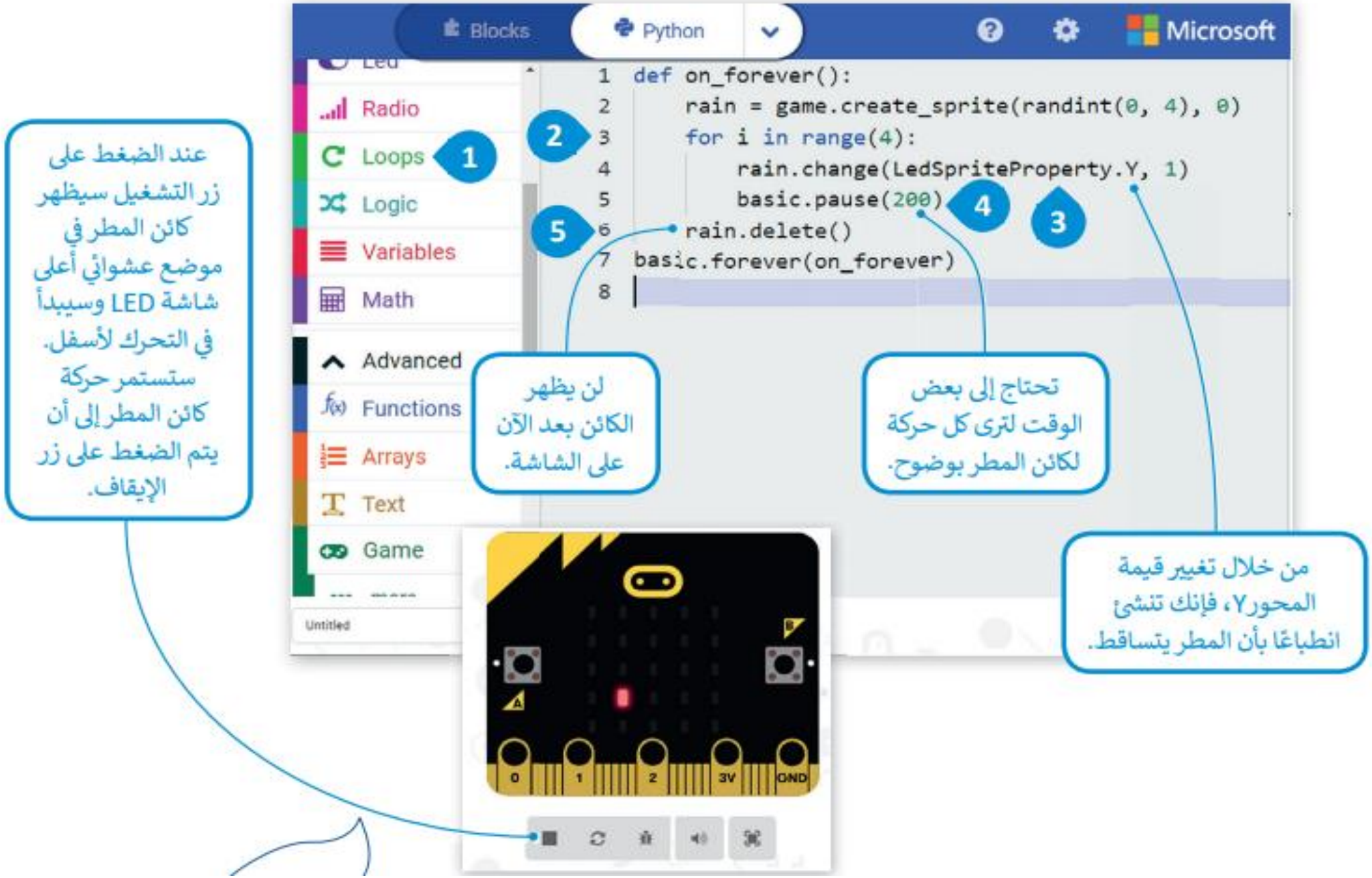
> اضغط على فئة **Loops** (حلقات). 1

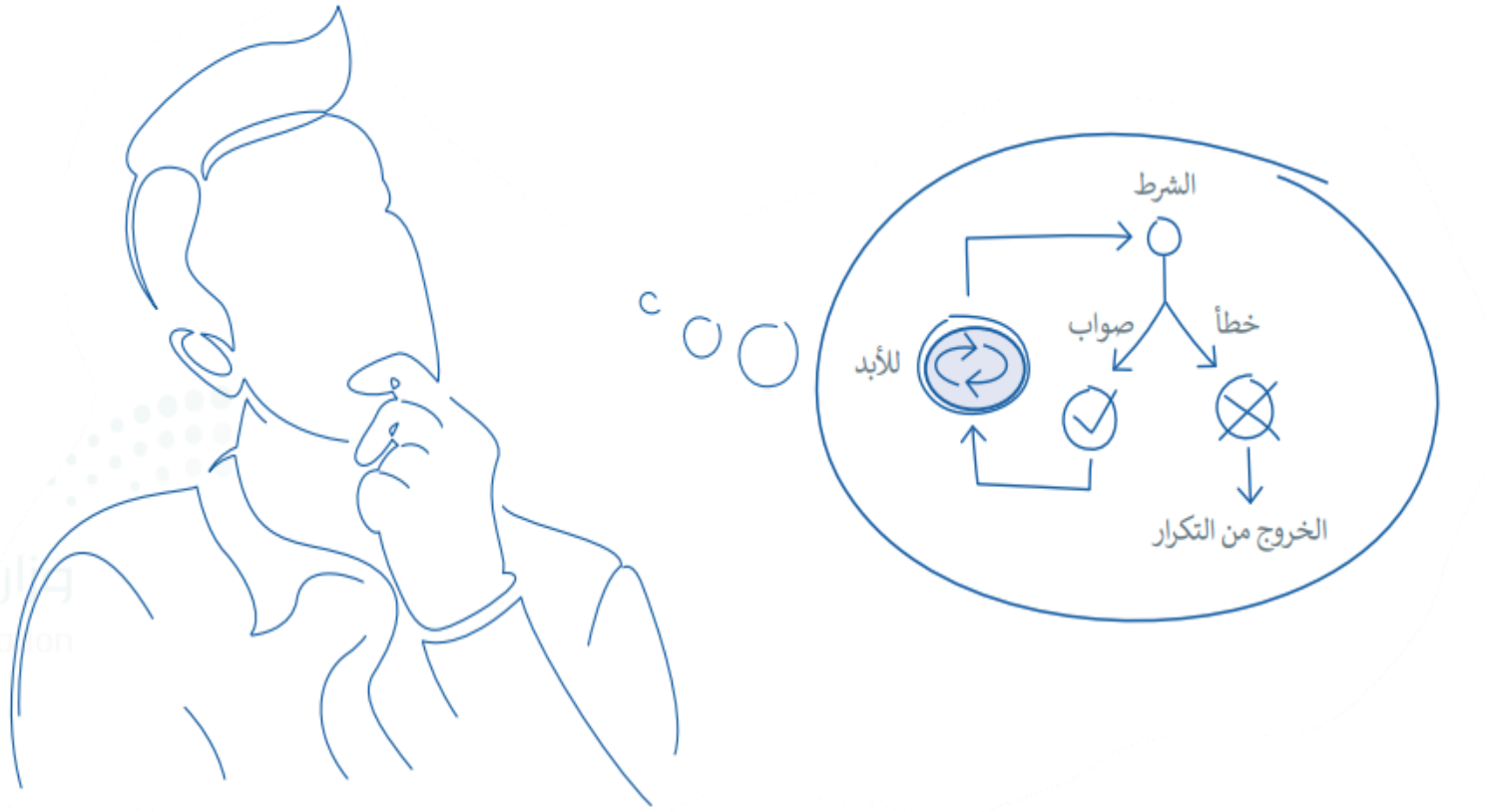
> حدد دالة **for** وضعها داخل دالة **run code forever** (للأبد run code). 2

> 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


Blocks
Python
Microsoft
Radio
Loops
Logic
Variables
Math
Advanced
Functions
Arrays
Text
Game
Untitled
1 def on_forever():
2 rain = game.create_sprite(randint(0, 4), 0)
3 for i in range(4):
4 rain.change(LedSpriteProperty.Y, 1)
5 basic.pause(200)
6 rain.delete()
7 basic.forever(on_forever)
8
1
2
3
4
5
0
1
2
3V
GND
عند الضغط على زر التشغيل سيظهر كائن المطر في موضع عشوائي أعلى شاشة LED وسيبدأ في التحرك لأسفل. ستستمر حركة كائن المطر إلى أن يتم الضغط على زر الإيقاف.
لن يظهر الكائن بعد الآن على الشاشة.
تحتاج إلى بعض الوقت لترى كل حركة لكائن المطر بوضوح.
من خلال تغيير قيمة المحور Y، فإنك تنشئ انطباعًا بأن المطر يتساقط.



يتم استخدام تكرار for عندما يكون عدد التكرارات محددا منذ البداية.

ماذا نفعل عندما يكون هذا الرقم غير معروف ويعتمد التكرار على شرط؟

مثل هذه الحالات تقدم بايثون لنا تكرار while.

تكرار While

تكرار While

يتم استخدامه تكرار عندما يكون عدد التكرارات غير معروف أو محدد مسبقا.

عندما يتحقق الشرط ويكون الشرط صحيحاً فأن التكرار سوف يتسمر الي ما لا نهاية

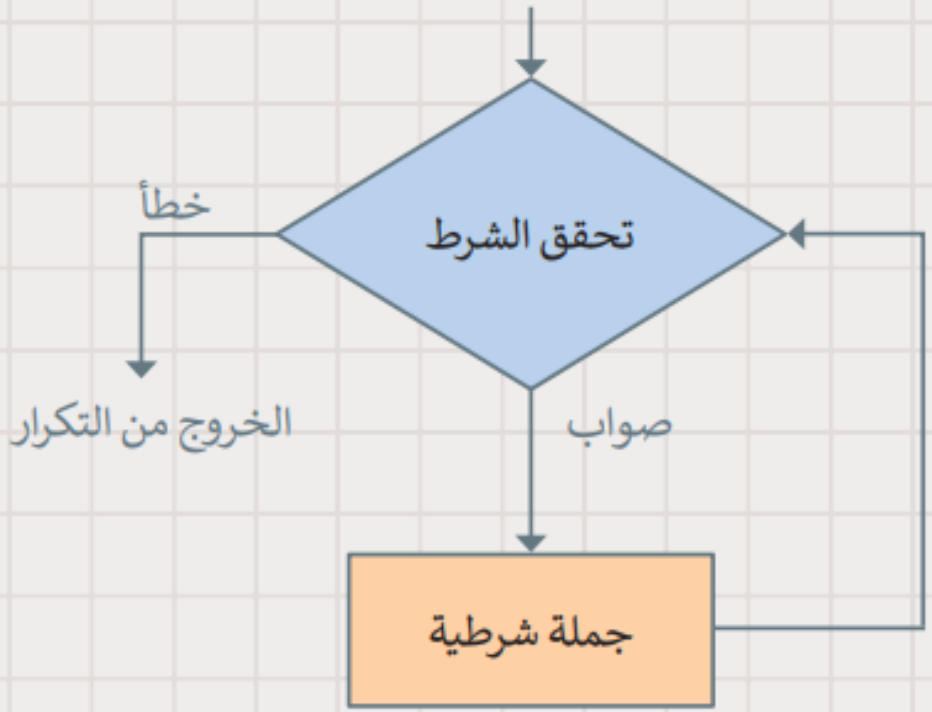
عندما لا يتحقق الشرط او يكون الشرط غير صحيحاً فأن التكرار سوف يتوقف

عندما يكون الشرط خطأ من البداية فإن عبارات التكرار لن يتم تنفيذها إطلاقاً.

يجب إضافة مسافة بادئة
لجمل التكرار

```
while condition:
    statements
```

مخطط المقطع البرمجي



مثال التكرارات 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.

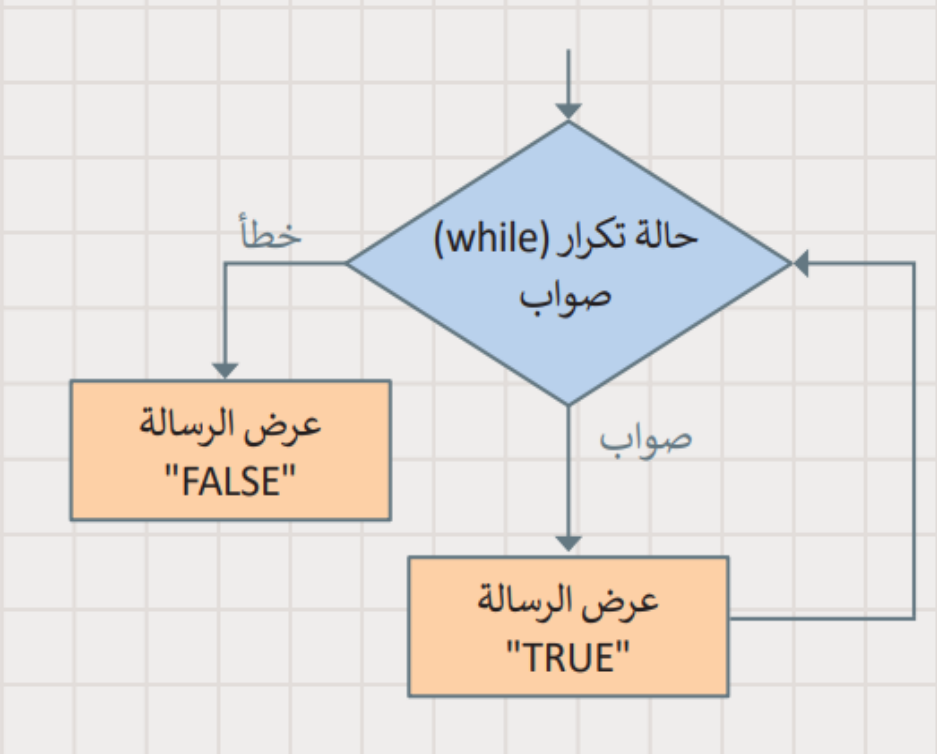
إذا لم تصبح حالة تكرار while خطأ، يصبح لديك تكرار لانهائي، وهو التكرار الذي لا يتوقف أبداً.

عند استخدام تكرار while، يجب عليك تضمين أمر أو مجموعة من الأوامر التي تغير حالة الشرط من الصواب إلى الخطأ.

لتطبق الجملة البرمجية التالية، ما الذي تلاحظه؟

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

ستعرض الشاشة ما يلي:
TRUE



سيتم في المثال السابق عرض الرسالة **TRUE** بشكل مستمر (إلى الأبد)، بينما لن يتم عرض رسالة **FALSE** على الشاشة نهائيا.

التقويم الختامي

تقويم ختامي

١	نستخدم تكرار (while) عندما يكون عدد التكرارات محدد ومعروف	☒
٢	لا تعد المسافة البادئة شيء مهماً في البايثون ويمكن أن ينفذ المقطع البرمجي بدونها	☒
٣	يجب تحدد عدد مرات التكرار عند استخدام تكرار (for)	☑

انتهت الحصة ☺