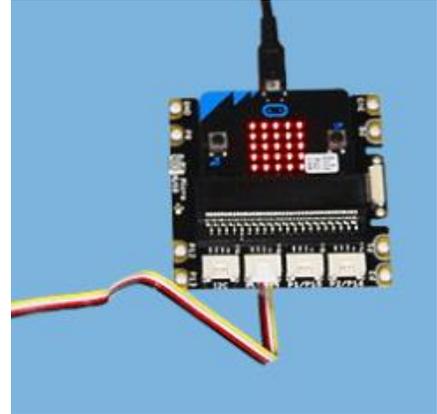


The Ultrasonic Ranger can be used to detect a change in distance. This concept can be used to build an alarm-type system that triggers lights and sounds when the distance is not within an allowed range. This activity develops the basic tools for an alarm system and the idea can be adapted to lots of different applications.

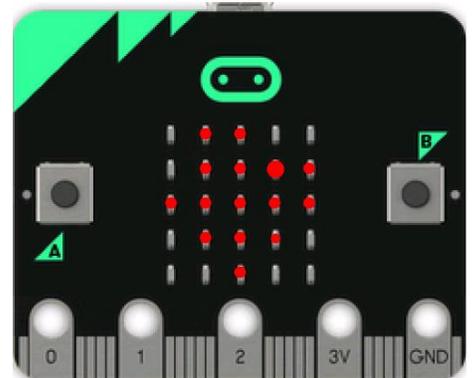
This alarm will trigger when a small box containing the ranger is opened.

1. Before you begin, be sure that:
 - your **micro:bit** is connected to your TI-Nspire™ CX II
 - your **micro:bit** is inserted in the **expansion board**.
(Grove shield shown here)
 - *the Grove accessories you will need are coming up...*



2. Your micro:bit should look like this when it has power from the TI-Nspire™ graphing calculator:

Recall that the display on the **micro:bit** is showing the TI logo, an icon of the state of Texas with a bright spot near Dallas, the home of **Texas Instruments, Inc.**



10 Minutes of Code: Python Modules

MICRO:BIT: INTRUDER ALERT!

TI-NSPIRE™ CX II

- This activity uses three external devices: the ultrasonic ranger, a speaker, and the LED that was used in the Light Switch activity. On the LED board, remember to turn the yellow dial (a potentiometer) all the way to the right to get the maximum brightness. Make the following micro:bit connections:

- pin0: speaker
- pin1: ranger
- pin2: LED

Note micro:bit version 2 has an on-board speaker. An external speaker can also be used. We will also use the micro:bit display board to flash lights in a custom pattern.

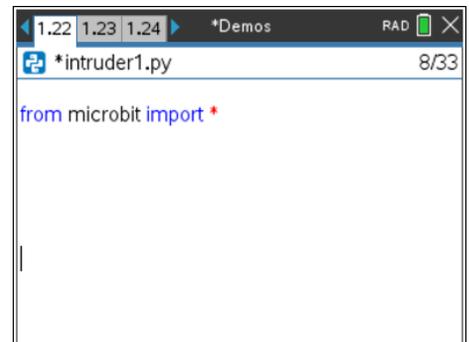


- Start a new TI-Nspire™ document and select **Add Python > New** to begin a new program (blank program), named 'intruder1'. In the Python Editor use **[menu] > More Modules > BBC micro:bit** to select the **import** statement at the top of the menu items:

from microbit import *

Tip: If the message 'micro:bit not connected' ever appears, just unplug the micro:bit and plug it in again (reset).

- The program is divided into two parts: a loop that 'watches' the distance and another loop that produces the alarm effects: lights and sounds. There are two **while** loops in the program...



```
1.22 1.23 1.24 *Demos RAD [X]  
*intruder1.py 8/33  
from microbit import *
```



```
1.22 1.23 1.24 *Demos RAD [X]  
*intruder1.py 6/36  
from microbit import *  
  
while ...  
  
while ...
```

10 Minutes of Code: Python Modules

TI-NSPIRE™ CX II

6. The first loop simply monitors the distance from the ranger that is connected to pin1 on the expansion board:

```
dist = grove.read_ranger_cm(pin1)  
while dist < 10:  
    dist = grove.read_ranger_cm(pin1)  
    sleep(250)
```

Recall that the ranger will be inside a small (<10cm high) closed box with a lid. The ranger (not the micro:bit) is placed on the inside bottom of the box facing upward to detect when the box cover is opened. If the lid is closed this loop will continue. When the lid is opened this loop ends.

7. **Sounding the alarm:**

The second loop creates the alarm effects. If you have an external speaker and LED you can use those to produce lights and sound. You can also use the on-board display and speaker (*speaker is on micro:bit version 2 only*).

This second loop ends when a 'special key' is pressed. We will use the **[esc]** key as usual but in practice you should use a special 'password' key to stop the alarm:

```
while get_key() != "esc":
```

8. To make the LED turn on and an alarm to sound, use the functions

```
grove.set_power(pin2, 100)  
music.pitch(440, 100)
```

grove.set_power() is found on > **micro:bit** > **Grove devices** > **Output**

pin2 is the connection for the LED

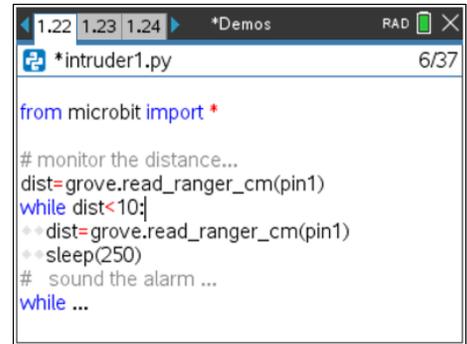
100 is full intensity (brightness)

music.pitch() is found on > **micro:bit** > **music**

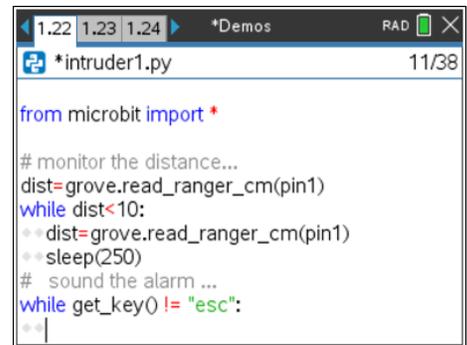
440 is 440Hz which is middle A on the music scale

100 is 0.1 seconds (100 milliseconds)

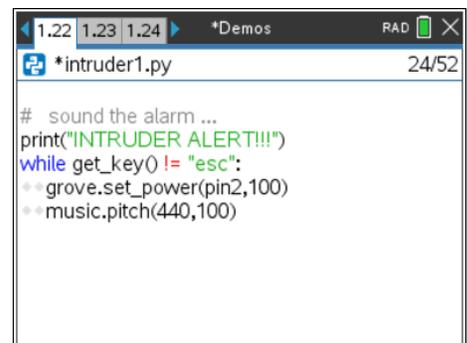
MICRO:BIT: INTRUDER ALERT!



```
1.22 1.23 1.24 *Demos RAD 6/37  
*intruder1.py  
from microbit import *  
  
# monitor the distance...  
dist=grove.read_ranger_cm(pin1)  
while dist<10:  
    dist=grove.read_ranger_cm(pin1)  
    sleep(250)  
# sound the alarm ...  
while ...
```



```
1.22 1.23 1.24 *Demos RAD 11/38  
*intruder1.py  
from microbit import *  
  
# monitor the distance...  
dist=grove.read_ranger_cm(pin1)  
while dist<10:  
    dist=grove.read_ranger_cm(pin1)  
    sleep(250)  
# sound the alarm ...  
while get_key() != "esc":  
    |
```



```
1.22 1.23 1.24 *Demos RAD 24/52  
*intruder1.py  
  
# sound the alarm ...  
print("INTRUDER ALERT!!!")  
while get_key() != "esc":  
    grove.set_power(pin2,100)  
    music.pitch(440,100)
```

10 Minutes of Code: Python Modules

TI-NSPIRE™ CX II

9. To make the LED blink and the alarm to sound 'alarming' add two more statements to turn the LED off and change the musical note:

```
grove.set_power(pin2, 0)  
music.pitch(220, 100)
```

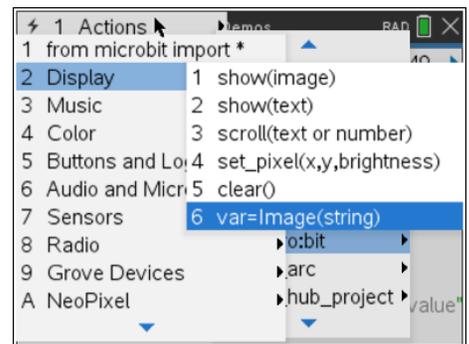
You can choose other frequencies for the musical tones.

MICRO:BIT: INTRUDER ALERT!

```
*intruder1.py 34/49  
  
# sound the alarm ...  
print("INTRUDER ALERT!!!")  
while get_key() != "esc":  
    grove.set_power(pin2,100)  
    music.pitch(440,100)  
    ..  
    ..  
    grove.set_power(pin2, 0)  
    music.pitch(220,100)
```

10. Let's add a flashing **display** effect on the micro:bit using **display.show()**.

The display can show any of the images in the built-in collection or you can make your own custom images using the **Image()** function found on the **> micro:bit > Display** menu shown here.



11. Here is a sample custom image:

The variable name is **img_alarm1**

The argument is a "string" of 25 digits (each from 0 to 9 indicating brightness) for each of the 25 pixels on the display. Each row of 5 values is separate by a colon (:).

It is convenient to write the string on five separate lines to arrange the digits as they will appear on the display (in a 5x5 grid):

```
Image("90909:"  
      "09090:"  
      "90909:"  
      "09090:"  
      "90909")
```

But it is also possible to write the string in one long line:

```
Image("90909:09090:90909:09090:90909")
```

or with extra quotes:

```
Image("90909:""09090:""90909:""09090:""90909")
```

Python will concatenate the separate strings for you!

```
*intruder1.py 4/53  
  
from microbit import *  
  
img_alarm1= Image("90909:"  
                  "09090:"  
                  "90909:"  
                  "09090:"  
                  "90909")
```

10 Minutes of Code: Python Modules

TI-NSPIRE™ CX II

12. `img_alarm1` lights up every other pixel on the display in a 'checkerboard' pattern.

Make another image which does the opposite pattern:

13. Finally, to get the display to flash when the alarm goes on, add two statements to your alarm code to display the two images (at the right moments!)

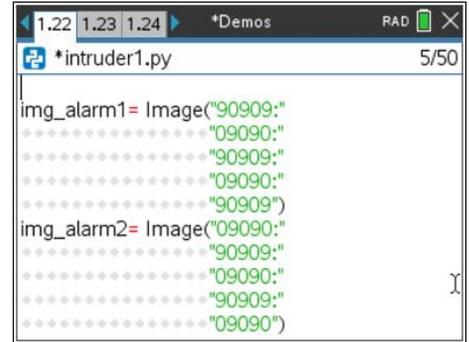
`display.show()` is on **>micro:bit > Display**

Type the names of your custom image variables as the arguments.

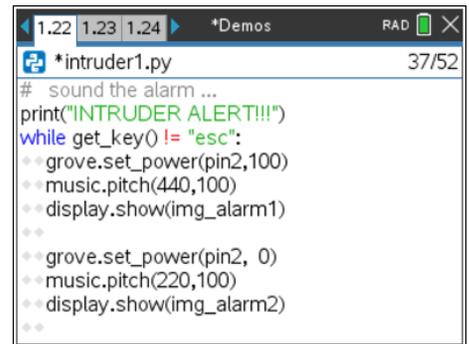
14. The effect is alarming!

Note: the video shows a portable amplified speaker connected to the micro:bit using a custom cable. (plug-n-play!).

MICRO:BIT: INTRUDER ALERT!



```
*intruder1.py 5/50
img_alarm1= Image("90909:"
....."09090:"
....."90909:"
....."09090:"
....."90909")
img_alarm2= Image("09090:"
....."90909:"
....."09090:"
....."90909")
```



```
*intruder1.py 37/52
# sound the alarm ...
print("INTRUDER ALERT!!!")
while get_key() != "esc":
    grove.set_power(pin2,100)
    music.pitch(440,100)
    display.show(img_alarm1)
    ..
    grove.set_power(pin2, 0)
    music.pitch(220,100)
    display.show(img_alarm2)
    ..
```

<see CX intruder.mp4>