# Micro:bit Programming - melodies

You can use a micro:bit to play your own (or other people’s) melodies

# Micro:bit Programming - melodies

Each note has a name (like C# or F), an octave (how high or low it is) and a duration (how long it lasts).

Octaves are indicated by a number - 0 is the lowest, 4 contains middle C and 8 is about as high as you'll ever need unless you're making music for dogs.

Durations are written as numbers - the higher the number, the longer it will last.

If you use the note name R then MicroPython will play a rest (i.e. silence) for the specified duration.

# Micro:bit Programming - melodies

Each note is written as a string of characters like this:

`NOTE[octave]:[duration]`

For example:

- `"C4:1"` is a single middle C note
- `"A1:4"` refers to the note named A in octave number 1 to be played for a duration of 4.

# Micro:bit Programming - melodies

Here's how you'd write a program to play a melody:

```
import music

tune = ["C4:4", "D4:4", "E4:4", "C4:4",
"C4:4", "D4:4", "E4:4", "C4:4", "E4:4",
"F4:4", "G4:8", "E4:4", "F4:4", "G4:8"]

music.play(tune)
```

Remember: `NOTE[octave]:[duration]`

# Micro:bit Programming - melodies

Python will remember the octave and duration you use to start with - so you can simplify the code like this:

```
import music

tune = ["C4:4", "D", "E", "C"]

music.play(tune)
```

Then you only need to type the octave or duration if it changes

Remember: `NOTE[octave]:[duration]`

# Micro:bit Programming - melodies

Using the **Python Editor**…

Write a program to play a melody

***Challenges:***

1. Start with something simple like the one I've given you
2. The try something short but interesting, like the opening bars of Beethoven's Fifth symphony (you'll know it)
3. Then try something more complicated...