ANSWERS

# F8.8 Sound

Data representation · GCSE · OCR J277 1.2.4, AQA 8525 3.3.7, Edexcel 1CP2 2.2.3 · about 15 min

## What this lesson is about

Sampling, sample rate, bit depth and file size, and finding a note from its samples.

- Sampling a wave
- Bit depth
- File size
- From samples back to a note
- Quality and size

## Questions

**5 marks in all**

**1.** A sound is sampled at 8,000 Hz with an 8-bit depth for 2 seconds. How many bits is it? [1 mark]

> **Answer:** 128000. 8,000 × 8 × 2.

**2.** What is the sample rate? [1 mark]

- ☐ **A** The number of samples taken each second
- ☐ **B** The number of bits in each sample
- ☐ **C** How loud the sound is
- ☐ **D** The length of the recording

> **Answer:** A. Measured in hertz: samples per second.

**3.** What does increasing the bit depth do? [1 mark]

- ☐ **A** Each sample is stored more accurately, and the file gets bigger
- ☐ **B** More samples are taken each second
- ☐ **C** The sound gets louder
- ☐ **D** The file gets smaller

> **Answer:** A. More bits give more levels for each measurement.

**4.** A 440 Hz wave crosses the middle going upwards how many times in half a second? [1 mark]

> **Answer:** 220. Once per cycle: 440 × 0.5.

**5.** How many levels can a 2-bit sample take? [1 mark]

> **Answer:** 4. 2 to the power 2.

## The task: find the note

Sample a note of 659 Hz for half a second at a sample rate of 8,000 Hz. Print `samples: <n>` and `8-bit size: <n> bits`, worked out from the rate and duration. Then count the upward crossings, print `note: <freq> Hz`, and play that frequency on the buzzer for half a second.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

import math

rate = 8000
seconds = 0.5
```

**The hint students can ask for:** Build the samples with a sine wave, then find the frequency by counting how many times the wave crosses zero on the way up. Divide those crossings by how long the recording lasted.

## A solution

```
from bugbot import *
connect()
import math

rate = 8000
seconds = 0.5
samples = [math.sin(2 * math.pi * 659 * i / rate) for i in range(int(rate * seconds))]
print("samples:", len(samples))
print("8-bit size:", len(samples) * 8, "bits")
rises = 0
for i in range(1, len(samples)):
    if samples[i - 1] < 0 <= samples[i]:
        rises = rises + 1
freq = round(rises / seconds)
print("note:", freq, "Hz")
tone(freq, 0.5)
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.