WORKSHEET

F8.5 Negative numbers: two's complement

Data representation · GCSE · Edexcel 1CP2 2.1.2 · about 12 min

Name _____ Class _____ Date _____

What this lesson is about

Signed 8-bit integers, and why the same adder works for them.

- The idea
- Making a number negative
- Why it works so well
- Negative speeds

Questions

4 marks in all

1. Write -5 in 8-bit two's complement. [1 mark]

2. What is 11111111 in 8-bit two's complement? [1 mark]

3. What range can an 8-bit two's complement number hold? [1 mark]

- ☐ **A** -128 to 127
- ☐ **B** -127 to 127
- ☐ **C** 0 to 255
- ☐ **D** -255 to 255

4. What does a leftmost bit of 1 mean in two's complement? [1 mark]

- ☐ **A** The number is negative
- ☐ **B** The number is odd
- ☐ **C** The number is over 100
- ☐ **D** Overflow happened

The task: signed readings

Write `to_twos(n)` yourself for -128 to 127 (you may use `format(n, "08b")` for the positive part). Print the 8-bit two's complement of 42, -42 and -128, one per line, in the form `-42 = 11010110`.

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

def to_twos(n):
    return format(n, "08b")
```

Plan your program here, then type it in and press Run.

Do it on the robot

www.bugbotlab.com/learn/f8-5-twos-complement/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. What happens if you try `to_twos(200)`? Add a check that refuses numbers out of range.
2. Add -60 and 40 as 8-bit two's complement using the adder from F8.4. Is the answer right?
3. How many bits would a two's complement number need to hold -1,000?