WORKSHEET

A7.3 Fixed point and floating point

Data representation · A level · OCR H446 1.4.1, AQA 7517 4.5.4.4, Eduqas A500QS 2.3 · about 35 min

BugBotLab

Name _____ Class _____ Date _____

What this lesson is about

Binary fractions, mantissa and exponent, normalisation, and adding floating point numbers.

- Binary fractions
- Fixed point
- Floating point: mantissa and exponent
- Normalisation
- Adding and subtracting floating point
- The robot's view

Questions

6 marks in all

1. An unsigned fixed point byte has 4 bits before the binary point and 4 after. What is the value of 10110110? [1 mark]

2. A floating point number has an 8-bit two's complement mantissa 01010000 and a 4-bit two's complement exponent 0011. What is its denary value? [1 mark]

3. A floating point number has an 8-bit two's complement mantissa 10100000 and a 4-bit two's complement exponent 0001. What is its denary value? [1 mark]

4. Which of these 8-bit two's complement mantissas are normalised? [1 mark]

Tick every answer that is true.

- ☐ **A** 01101000
- ☐ **B** 00110000
- ☐ **C** 10110000
- ☐ **D** 11010000

5. Normalise the floating point number with 8-bit mantissa 00011000 and 4-bit exponent 0011. Give the mantissa, a space, then the exponent. [1 mark]

6. Why are floating point numbers normalised? [1 mark]

Tick every answer that is true.

- ☐ **A** It gives the greatest precision for the number of mantissa bits
- ☐ **B** Each value has only one representation
- ☐ **C** It increases the range of the exponent
- ☐ **D** It stops rounding errors completely

The task: decode a float

Write two functions. - `float_value(mantissa, exponent)` : `mantissa` is a string of 8 characters `0` or `1` , a two's complement fixed point number with the binary point after the first bit (place values -1, ½, ¼ down to 1/128). `exponent` is a string of 4 characters `0` or `1` , a two's complement integer from -8 to 7. It returns the value, mantissa × 2 to the power of exponent, as a `float` . - `is_normalised(mantissa)` : returns `True` when the first two bits of the mantissa are different, otherwise `False` . The loop prints each line as `01101000 0011 = 6.5 normalised` or `00011000 0101 = 6.0 not normalised` , with the value exactly as Python prints the float.

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

def float_value(mantissa, exponent):
    return 0.0

def is_normalised(mantissa):
    return False

for mantissa, exponent in [("01101000", "0011"), ("10110000", "0010"), ("00011000",
"0101"), ("11100000", "1111")]:
    word = "normalised" if is_normalised(mantissa) else "not normalised"
    print(mantissa, exponent, "=", float_value(mantissa, exponent), word)
```

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

Do it on the robot

www.bugbotlab.com/learn/a7-3-fixed-and-floating-point/

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

Challenges

1. Write `normalise(mantissa, exponent)` that returns the normalised pair of strings.
2. Store -0.375 in normalised form with an 8-bit mantissa and 4-bit exponent.
3. What is the largest positive value this 12-bit format can hold, and what is the smallest positive normalised value?