ANSWERS

A7.1 Number sets, bases and units

Data representation · A level · OCR H446 1.4.1, AQA 7517 4.5.1.1, Eduqas A500QS 2.3 · about 25 min

What this lesson is about

Natural, integer, rational, irrational, real and ordinal numbers; any number base; bits, bytes, kilo and kibi.

- Number sets
- Counting and measurement
- Number bases
- Bits, bytes and units

Questions

6 marks in all

1. Which set of numbers is the square root of 2 in, but not in the rational numbers? [1 mark]

- ☐ **A** Irrational numbers
- ☐ **B** Natural numbers
- ☐ **C** Integers
- ☐ **D** Ordinal numbers

> **Answer:** A. The square root of 2 cannot be written as a fraction of two integers, so it is irrational (and real).

2. Which of these are rational numbers? [1 mark]

Tick every answer that is true.

- ☐ **A** 0.75
- ☐ **B** -3
- ☐ **C** 22/7
- ☐ **D** The square root of 2
- ☐ **E** Pi

> **Answer:** A, B, C. A rational number can be written as one integer divided by another: 3/4, -3/1 and 22/7. The square root of 2 and pi cannot.

3. What are ordinal numbers used for? [1 mark]

- ☐ **A** To describe a position in an ordered list, such as 1st, 2nd, 3rd
- ☐ **B** To count how many things there are
- ☐ **C** To measure a quantity such as a length
- ☐ **D** To store negative values

> **Answer:** A. Ordinal numbers give position. Counting uses natural numbers; measuring uses real numbers.

4. Write the denary number 255 in base 8. [1 mark]

> **Answer:** 377. 255 = 3 x 64 + 7 x 8 + 7, so the digits are 3, 7, 7.

5. How many bytes are in 3 KiB? [1 mark]

> **Answer:** 3072. A kibibyte is 2 to the power 10 = 1,024 bytes, so 3 KiB is 3 x 1,024 bytes.

6. What does this program print? [1 mark]

```
n = 47
digits = ""
while n > 0:
    digits = str(n % 5) + digits
    n = n // 5
print(digits)
```

Answer:

```
142
```

Repeated division by 5 gives remainders 2, 4, 1, read from last to first: 47 = 1 x 25 + 4 x 5 + 2.

The task: any base

Write `to_base(n, base)`. The parameter `n` is a whole number, 0 or more; `base` is a whole number from 2 to 16. It returns the digits of `n` in that base as a string, using `0` to `9` then `A` to `F`, and returns `"0"` when `n` is 0. Use repeated division with `//` and `%`: you may not use `bin`, `hex`, `oct` or `format`. The loop at the bottom prints lines such as `173 in base 16 = AD`.

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

DIGITS = "0123456789ABCDEF"

def to_base(n, base):
    # repeated division goes here
    return ""

for n, base in [(173, 2), (173, 8), (173, 16), (0, 2), (2024, 16)]:
    print(n, "in base", base, "=", to_base(n, base))
```

The hint students can ask for: Divide by the base again and again. Each remainder is one digit, and the first remainder you get is the rightmost digit, so build the string from the right. Decide what happens when n starts at 0.

A solution

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

DIGITS = "0123456789ABCDEF"

def to_base(n, base):
    if n == 0:
        return "0"
    digits = ""
    while n > 0:
        digits = DIGITS[n % base] + digits
        n = n // base
    return digits

for n, base in [(173, 2), (173, 8), (173, 16), (0, 2), (2024, 16)]:
    print(n, "in base", base, "=", to_base(n, base))
```

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