ANSWERS

A7.2 Binary arithmetic and signed integers

Data representation · A level · OCR H446 1.4.1, AQA 7517 4.5.2.1, Eduqas A500QS 2.3 · about 30 min

What this lesson is about

Unsigned addition and multiplication, sign and magnitude, two's complement, subtraction and overflow.

- Unsigned binary
- Sign and magnitude
- Two's complement
- Carry is not overflow
- Hexadecimal and signed values

Questions

6 marks in all

1. What is the denary value of the 8-bit two's complement number 11101100? [1 mark]

> **Answer:** -20. The place values are -128, 64, 32, 16, 8, 4, 2, 1: -128 + 64 + 32 + 8 + 4 = -20.

2. Write -37 as an 8-bit two's complement binary number. [1 mark]

> **Answer:** 11011011. 37 is 00100101. Flip every bit to get 11011010, then add 1 to get 11011011.

3. What is the range of an 8-bit two's complement integer? [1 mark]

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

> **Answer:** A. The most negative value is 10000000 = -128 and the largest is 01111111 = 127.

4. Write -5 in 8-bit sign and magnitude. [1 mark]

> **Answer:** 10000101. The first bit is the sign, 1 for negative, and the other seven bits are 5 in binary, 0000101.

5. 01100100 + 01000110 is calculated in 8-bit two's complement. What happens? [1 mark]

- ☐ **A** Overflow: two positive numbers give a result with the sign bit set
- ☐ **B** A carry out of the top bit, which is an error
- ☐ **C** The correct answer, 170
- ☐ **D** Nothing unusual: the result is positive

> **Answer:** A. 100 + 70 = 170 is more than 127. The result 10101010 reads as negative, which is overflow; there is no carry out of the top bit.

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

```
a = 0b11111011
b = 0b00000101
total = a + b
print(format(total & 0xFF, "08b"), total >> 8)
```

Answer:

```
00000000 1
```

The sum is 256, which needs 9 bits: the low 8 bits are all 0 and the carry out is 1. In two's complement this is -5 + 5 = 0, with no overflow.

The task: a signed adder

Write `add8(a, b)`. Each parameter is a string of exactly 8 characters, each `0` or `1`, holding an 8-bit two's complement number. It returns a tuple `(total, carry, overflow)`: - `total`: the 8-character bit string of the sum, with any carry out of the leftmost column dropped - `carry`: the carry out of the leftmost column, the integer 0 or 1 - `overflow`: `True` when both inputs have the same sign bit and `total` has the other one, otherwise `False` Add column by column from the right; do not convert to denary with `int(..., 2)`, `bin` or `format`. The loop prints each line as `01110000 + 00110000 = 10100000 carry 0 overflow True`.

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

def add8(a, b):
    total = ""
    carry = 0
    # add the columns from right to left here
    return total, carry, False

tests = [("01110000", "00110000"), ("11111011", "00000101"),
         ("10000000", "11111111"), ("00101101", "11110110")]
for a, b in tests:
    total, carry, overflow = add8(a, b)
    print(a, "+", b, "=", total, "carry", carry, "overflow", overflow)
```

The hint students can ask for: Work from the rightmost column to the left, adding the two bits and the carry in. The column's bit is the total mod 2 and the new carry is the total divided by 2. Overflow depends only on the sign bits: of the two inputs, and of the answer.

A solution

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

def add8(a, b):
    total = ""
    carry = 0
    for i in range(7, -1, -1):
        s = int(a[i]) + int(b[i]) + carry
        total = str(s % 2) + total
        carry = s // 2
    overflow = a[0] == b[0] and total[0] != a[0]
    return total, carry, overflow

tests = [("01110000", "00110000"), ("11111011", "00000101"),
         ("10000000", "11111111"), ("00101101", "11110110")]
for a, b in tests:
    total, carry, overflow = add8(a, b)
    print(a, "+", b, "=", total, "carry", carry, "overflow", overflow)
```

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