Past-paper walk-through

Floating-point numbers, representation and manipulation ▪ 13.3

eXercise

Questions in this set

- 9618/31 N25 Q2 ▪ 6 marks
- 9618/31 J25 Q2 ▪ 6 marks
- 9618/32 J25 Q2 ▪ 6 marks
- 9618/31 N24 Q1 ▪ 6 marks
- 9618/32 N24 Q4 ▪ 6 marks

Reading the mark scheme

- **B1** a mark that stands on its own – the point is either made or not.
- **M1** a *method* mark: the right approach, even if the number is wrong.
- **A1** an *answer* mark, and it usually needs its M mark first.
- **C1** a working mark on the way to the answer.
- // separates answers the examiner accepts equally.
- **ecf** = error carried forward: a later part is marked on your own earlier answer.
- **owtte** = "or words to that effect" – the wording need not match.

Question 2

9618/31 N25 ▪ Q2 ▪ 6 marks

Numbers are stored in a computer system using binary floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Write the normalised floating-point representation of the following positive binary number using this system.

$$0.00000001110101101$$

Mantissa **Exponent**

Question 2

□□□□□□□□□□□□ □□□□ **[2]**

(b) Calculate the normalised binary floating-point representation of -76.1875 in this system. Show your working.

Mantissa **Exponent**

□□□□□□□□□□□□ □□□□ **[4]**

Solution 2

(a) Worked steps

Normalising means shifting until the mantissa starts `0.1` for a positive number — the sign bit, then the first significant 1 immediately after the point.

- The number is `0.00000001110101101`. Its first 1 sits **8** places after the point.
- Shift the point 8 places right to get `0.1110101101…`, so the mantissa is `0 1110101101…` (sign bit 0, then the digits) padded to its field width.
- Each shift right of the point is one **added** to the exponent, so the exponent is $+8$, written as a two's-complement binary integer in its field: `01000`.

The check: a normalised **positive** mantissa always reads `0.1…`; if it reads `0.0…` you have not shifted far enough, and if it reads `1.…` the sign is wrong. **Mark scheme**

Solution 2

- One mark per mark point (Max 2)
- MP1
- Correct mantissa
- MP2
- Correct exponent
- Mantissa
- Exponent
- 0
- 1
- 1

Solution 2

- 1
- 0
- 1
- 0
- 1
- 1
- 0
- 1
- 0
- 1

Solution 2

- 0
- 0
- 1

Solution 2

(b) Worked steps

Work in three separate steps, and show each — two of the four marks are for the working.

1. **Convert the magnitude to binary.** $76 = 1001100$ and $0.1875 = 0.0011$, so $76.1875 = 1001100.0011$.
2. **Normalise the positive form.** Move the point 7 places left: `0.10011000011`, exponent $+7$.
3. **Negate the mantissa** (the number is negative), using two's complement on the mantissa only: flip every bit and add 1, giving a mantissa that starts `1.0…` — the mark scheme's pattern for a normalised negative number.

Solution 2

The exponent is **not** negated: it is a separate two's-complement integer, and only the mantissa carries the sign of the number. **Mark scheme**

- One mark per mark point (Max 4)
- MP1 correct method to find the binary number
- MP2 additional working towards binary number
- MP3 correct use of exponent
- MP4 correct answer in the space provided
- Two from:
- e.g.

Solution 2

- 76.1875 = 64+8+4+0.125+0.0625
- (0)1001100.0011
- –76.1875 = –128+32+16+2+1+0.5+0.25+0.0625
- 10110011.11 01
- One mark movement of binary point by 7 places // 1.011001111 01 x 27
- One mark
- Mantissa
- Exponent
- 1
- 0

Solution 2

- 1
- 1
- 0
- 0
- 1
- 1
- 1
- 1
- 0
- 1

Solution 2

- 0
- 1
- 1
- 1

Question 2

9618/31 J25 ▪ Q2 ▪ 6 marks

Numbers are stored in a computer using binary floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Write the normalised floating-point representation of the following binary number using this system.

0.00000011010111

Mantissa **Exponent**

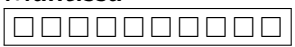
□□□□□□□□□□	□□□□□□

[2]

(b) Calculate the normalised binary floating-point representation of -25.3125 in this system. Show your working.

Question 2

Mantissa **Exponent**



[4]

Solution 2

(a) Worked steps

- The first 1 in `0.00000011010111` is **7** places after the point.
- Shift the point 7 places right: the mantissa becomes `0.11010111`…, written as sign bit 0 followed by those digits and padded to the field width.
- Shifting right adds to the exponent, so the exponent is $+7$ as a two's-complement integer.

Solution 2

A normalised positive number always reads `0.1...` — that pattern is the check, and it is one of the two marks. **Mark scheme**

One mark per mark point
MP1 Correct mantissa
MP2 Correct exponent

Mantissa

0	1	1	0	1	0	1	1	1	0

Solution 2

Exponent

1	1	1	0	1	0

Solution 2

(b) Worked steps

Show the working; two of the four marks are for it.

1. **Magnitude in binary:** $25 = 11001$ and $0.3125 = 0.0101$, so $25.3125 = 11001.0101$.
2. **Two's complement:** flip every bit of the positive form and add 1, which gives the negative representation.
3. **Normalise:** shift the point until the mantissa reads `1.0…` — the pattern a normalised **negative** number takes — and set the exponent to the number of places shifted.

Solution 2

Order matters: negate first, then normalise. Normalising the positive value and negating afterwards can leave the mantissa un-normalised, and that costs the last mark. **Mark scheme**

One mark per mark point for working (Max 2)

- number converted to binary e.g., positive binary version of $25.3125 = (0)11001.0101$
- two's complement version bits flipped and 1 added $= 100110.1011$
- $-32 + 4 + 2 + 0.5 + 0.125 + 0.0625$ // $-32 + 4 + 2 + 1/2 + 1/8 + 1/16$
- movement of binary point seen (5 places)

Solution 2

One mark per mark point

- correct mantissa
- correct exponent

Mantissa

1	0	0	1	1	0	1	0	1	1

Solution 2

Exponent

0	0	0	1	0	1

Question 2

9618/32 J25 ▪ Q2 ▪ 6 marks

Numbers are stored in a computer using binary floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised binary floating-point representation of $+124.4375$ in this system. Show your working.

Mantissa **Exponent**

□□□□□□□□□□□□	□□□□

[3]

Question 2

9618/32 J25 ▪ Q2 ▪ 6 marks

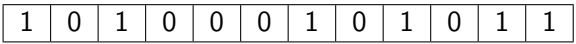
Numbers are stored in a computer using binary floating-point representation with:

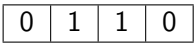
- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(b) Calculate the denary value of the following normalised binary floating-point number. Show your working.

Mantissa **Exponent**

Question 2





Question 2

[3]

Solution 2

(a) Worked steps

Three steps, always the same three, and the marks follow them exactly: two for the working, one for the answer.

Step 1 — convert to binary, whole part and fraction separately.

- $124 = 64 + 32 + 16 + 8 + 4$, so `1111100`.
- $0.4375 = 0.25 + 0.125 + 0.0625$, so `.0111`.
- Together: `1111100.0111`

Step 2 — normalise. A *positive* two's complement mantissa is normalised when it reads `0.1...`, so slide the binary point until exactly that pattern appears. Here it moves **7 places to the left**, so the exponent is **7**.

- $124.4375 = 0.11111000111 \times 2^{7}$

Solution 2

Step 3 — pack the fields.

- Mantissa, 12 bits: `0111 1100 0111` — the sign bit 0, then the eleven fraction bits.
- Exponent, 4 bits in two's complement: $7 = 0111$.

The normalisation rule is the whole question. `0.1…` for a positive number and `1.0…` for a negative one — anything else is unnormalised and loses the mark even when the value is right. Count the places you move the point and write the number down as you go; recounting at the end is where the exponent goes wrong. **Mark scheme**

- Two marks for working
- ▪ number converted to binary (0)1111100.0111

Solution 2

- ▪ use of exponent = 7 // Moving binary point the correct number (7) of places
- One mark for correct answer
- Mantissa
- Exponent
- 0
- 1
- 1
- 1
- 1
- 1

Solution 2

- 0
- 0
- 0
- 1
- 1
- 1
- 0
- 1
- 1
- 1

Solution 2

(b) Worked steps

Reverse the same three steps: unpack, denormalise, then evaluate.
Step 1 — apply the exponent. The exponent is 6, so move the binary point **6 places to the right**, giving `1010001.01011`.
Step 2 — evaluate as two's complement. The leading bit is 1, so the number is negative and the top place value is **negative**:

- $-64 + 16 + 1 + 0.25 + 0.0625 + 0.03125 = \mathbf{-46.65625}$

Solution 2

Two marks for the working — correct use of the exponent, and a correct conversion method — and one for the answer.
The alternative route earns the same marks: negate the mantissa first (invert and add one), evaluate the resulting positive number, then put the minus sign back. That gives $32 + 8 + 4 + 2 + 0.5 + 0.125 + 0.03125 = 46.65625$, so -46.65625.
The error to avoid is treating the leading 1 as a positive $+64$. In two's complement only the **most significant** place value is negative; every other bit contributes its usual positive amount, which is why the two routes agree. **Mark scheme**

- Two marks for working
- ▪ correct use of exponent seen

Solution 2

- ▪ correct conversion method from binary to denary
- One mark for correct answer
- Working:
- 1010001.01011 // moving bp 6 places to right
- Evaluation of two's complement $-64 + 16 + 1 + 0.25 + 0.0625 + 0.03125$ //
- $-64 + 16 + 1 + 1/4 + 1/16 + 1/32$ // Converting two's complement back and evaluating positive binary number $32 + 8 + 4 + 2 + 0.5 + 0.125 + 0.03125$ // 32
- $+ 8 + 4 + 2 + 1/2 + 1/8 + 1/32$
- Fractions methods - award both working marks for either
- $(- 2048 + 512 + 32 + 8 + 2 + 1)$ / 2048 x 26 = -1493 / 32

Solution 2

- OR
- $-1 + 1/4 + 1/64 + 1/256 + 1/1024 + 1/2048 \times 26 = -1493 / 32$
- Answer:
- -46.65625 // $-4621/32$

Question 1

9618/31 N24 ▪ Q1 ▪ 6 marks

Numbers are stored in a computer using binary floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Calculate the normalised binary floating-point representation of $+201.125$ in this system.

Show your working.

Mantissa **Exponent**

□□□□□□□□□□ □□□□□□

[3]

Question 1

9618/31 N24 ▪ Q1 ▪ 6 marks

Numbers are stored in a computer using binary floating-point representation with:

- 10 bits for the mantissa
- 6 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(b) Calculate the denary value of the given normalised binary floating-point number. Show your working.

Question 1

Mantissa

1	0	1	0	1	1	0	0	1	1

Exponent

0	0	0	1	0	1

Question 1

[3]

Solution 1

(a) Worked steps

Step 1 — convert to binary.

- $201 = 128 + 64 + 8 + 1$, so `11001001`.
- $0.125 = \frac{1}{8}$, so `.001`.
- Together: `11001001.001`

Step 2 — normalise. Move the binary point 8 places left to reach the `0.1…` form a positive mantissa needs, so the **exponent is 8**:

- $201.125 = 0.11001001001 \times 2^{8}$

Step 3 — pack. The exponent is `001000` in 6-bit two's complement. But the mantissa needs **eleven** fraction bits and only **nine** are available, so the last two bits are lost:

Solution 1

- Mantissa: 0110010010

△ **Say so.** A mark here is specifically for stating that the number **loses precision** — a rounding error — because it cannot be represented exactly in this format. The stored value is 201, not 201.125, and a question that asks you to "show your working" on a number that does not fit is asking you to notice that.
Count the bits of your normalised mantissa against the field width **before** you write the answer. That comparison is the whole point of the question, and it is invisible if you simply truncate without comment. **Mark scheme**

- One mark per mark point (Max 1)
- ▪ correct answer

Solution 1

- ▪ statement regarding number losing precision/rounding error
- One mark per mark point for working (Max 2)
- ▪ number converted to binary $201.125 = 11001001.001$
- // $128 + 64 + 8 + 1 + 0.125$ / $1/8$ seen
- ▪ use of the exponent e.g. moving the binary point 8 places / $\times$ 28.
- Mantissa
- Exponent
- 0
- 1
- 1

Solution 1

- 0
- 0
- 1
- 0
- 0
- 1
- 0
- 0
- 0
- 1

Solution 1

- 0
- 0
- 0

Solution 1

(b) Worked steps

Step 1 — apply the exponent. It is 5, so move the binary point 5 places right:

- `1.010110011` becomes `101011.0011`

Step 2 — evaluate as two's complement, with the top place value negative:

- $-32 + 8 + 2 + 1 + 0.125 + 0.0625 = \mathbf{-20.8125}$

Two marks for the working — the movement of the point, and the evaluation — and one for the answer, which may also be written $-20\frac{13}{16}$.

The mantissa begins `1.0`…, which is the **normalised form for a negative number**. That is the immediate signal that the answer must come out negative, and it is worth checking against your arithmetic before you write it down: a positive result from a

Solution 1

mantissa starting with 1 means the top bit was given a positive place value. **Mark scheme**

- One mark per mark point (Max 2)
- ▪ application of exponent to go from 1.010110011 to 101011.0011 // x 25 // movement of binary point 5 places seen
- ▪ −32 + 8 + 2 + 1 + .125 + .0625 // −32 + 8 + 2 + 1 + 1/8 + 1/16 seen
- // −1 + ¼ + 1/16 + 1/32 + 1/256 + 1/512 // −1 + 179/512 // −333/512
- One mark for correct answer (Max 1)
- ▪ −20.8125 //

Solution 1

- –2013/16

Question 4

9618/32 N24 ▪ Q4 ▪ 6 marks

4 Numbers are stored in a computer using binary floating-point representation with:

- 12 bits for the mantissa
- 4 bits for the exponent
- two's complement form for both the mantissa and the exponent.

(a) Calculate the denary value of the given normalised binary floating-point number.

Show your working.

Mantissa

0	1	0	0	0	1	1	1	0	1	1	1

Exponent

0	1	1	1

Question 4

[2]

(b) Calculate the normalised binary floating-point representation of –49.1875 in this system.

Show your working.

Mantissa **Exponent**

Question 4

[4]

Solution 4

(a) Worked steps

Unpack, apply the exponent, then evaluate.

- The mantissa is `0.10001110111` and the exponent is 7, so move the binary point **7 places right**: `01000111.0111`.
- The leading bit is 0, so the number is positive and every place value is positive: $64 + 4 + 2 + 1 + 0.25 + 0.125 + 0.0625 = \mathbf{71.4375}$

Solution 4

One mark for the working, one for the answer, which is also $71\frac{7}{16}$.
Either method is accepted: shift the point and add place values as above, or treat the mantissa as a fraction and multiply by $2^7 = 128$. They must agree — doing both is a free check that costs one line.
Moving the point the wrong way is the standard error. A **positive** exponent means the number is larger than the mantissa, so the point moves **right**; a negative exponent moves it left. **Mark scheme**

- One mark for working
- ▪

Solution 4

- application of exponent to mantissa to go from 0.10001110111 to 01000111.0111 //
- moving the binary point 7 places //
- multiplying by 27/128 in the fractions method //
- 64 + 4 + 2 +1 + .25+ .125 + .0625 seen
- One mark for correct answer
- ▪ 71.4375
- // 717/16
- 2

Solution 4

(b) Worked steps

Two marks and both are all-or-nothing — **exact answers only**, one for the mantissa and one for the exponent. So work in the order that makes each field certain.

1. **Convert to binary**, whole part and fraction separately, and write out every bit.
2. **Normalise**: slide the binary point until the mantissa reads `0.1…` for a positive number or `1.0…` for a negative one. Count the places as you move, and note the direction — left gives a positive exponent, right a negative one.
3. **Pack the mantissa** into its 12 bits: sign bit, then the fraction bits, padding with **zeros on the right** if there are too few.
4. **Pack the exponent** into 4 bits of two's complement. Four bits hold -8 to $+7$, so check the exponent is in range before writing it.

Solution 4

For a **negative** number, do the negation last: convert the magnitude, normalise it, then take the two's complement of the whole mantissa — invert every bit and add one. Negating before normalising is what produces a mantissa that is correct in value but the wrong shape, and with exact-answer marking that scores nothing. **Mark scheme**

- One mark per mark point (Max 2)
- ▪ correct mantissa – exact answer only
- ▪ correct exponent – exact answer only
- Mantissa
- Exponent
- 1

Solution 4

- 0
- 0
- 1
- 1
- 1
- 0
- 1
- 1
- 0
- 1

Solution 4

- 0
- 0
- 1
- 1
- 0
- One mark per mark point for working (Max 2)
- ▪
- number converted to binary e.g., positive binary version of 49.1875 = 0110001.0011 //
- two's complement version bits flipped and 1 added = 1001110.1101 //

Solution 4

- $-64 + 8 + 4 + 2 + .5 + .25 + .0625$ //
- $-64 + 14.8125$
- use of the exponent e.g. moving the binary point 6 places / $\times$ 26.
- 4
- 9618/32
- October/November 2024