4. Approximation & Rounding errors

State why some binary representation can lead to rounding errors

- There’s no exact binary conversion for some numbers
- More bits are needed to store the number

[

...............Overflow.................. can occur in the exponent of a floating-point number, when the exponent has become too large to be represented using the number of bits available.

A calculation results in a number so small that it cannot be represented by the number of bits available. This is called ....Underflow.......... .

]

[

A student enters the following into an interpreter:

```
OUTPUT(0.2 * 0.4)
```

The student is surprised to see that the interpreter outputs the following:

```
0.08000000000000002
```

Explain why the interpreter outputs this value.

]

- 0.2 and 0.4 cannot be represented exactly in binary, there is a rounding error
- 0.2 has been represented by a number just greater than 0.2
- This is similar for 0.4
- Therefore, multiplying these two representations together increases the difference
- Difference after calculation is significant enough to be seen

---

[

A student writes a program to output numbers using the following code:

```
X ← 0.0
FOR i ← 0 TO 1000
   X ← X + 0.1
   OUTPUT X
ENDFOR
```

The student is surprised to see that the program outputs the following sequence:

0.0   0.1   0.2   0.2999999   0.3999999 ……

Explain why this output has occurred.

]

- 0.1 cannot be represented exactly in binary, there is a rounding error
- 0.1 is represented by a value just less than 0.1
- The loop keeps adding this approximate value to the counter
- Until all accumulated small difference become significant enough to be seen