32 bit IEEE 754 format

- **Sign Bit:**
  - 0 means positive, 1 means negative
- Value of a number is:
  \[ (-1)^s \times F \times 2^E \]

Normalized Numbers and the significand

- Normalized binary numbers *always* start with a 1 (the leftmost bit of the significand value is a 1).
- Why store the 1 (it’s always there)?
- IEEE 754 uses this, so the fraction is 24 bits but only 23 need to be stored.
- All numbers must be normalized!

Exponent Representation

- We need negative and positive exponents.
- Could use 2s complement notation
  - this would make comparison of floating point numbers a bit tricky.
    - exponent value 11111111 is smaller than 00000000.
  - Instead they chose a *biased* (excess-K) representation.
    - exponent values are offset by a fixed bias.
32 bit IEEE 754 exponent

- The exponent uses 8 bits.
- The bias is 127.
  - treat the 8 bit exponent as an unsigned integer and subtract 127 from it.

00000001 is the representation for \(-126\)
10000000 is the representation for +1
11111110 is the representation for +127

Special Exponents

- 00000000 is a special case exponent
  - used for the representation of the floating point number 0 (and other things, depending on the sign and significand).

- 11111111 is also a special case
  - used in the representation of infinity (and other things, depending on the sign and significand).

32 bit IEEE 754 Range

- Smallest (positive) normalized number is:
  \( 1.00000000000000000000000 \times 2^{-126} \)

- Largest normalized number is:
  \( 1.11111111111111111111111 \times 2^{127} \)
Expression for value of 32 bit IEEE 754

\[ (-1)^s \times (1 + \text{significand}) \times 2^{(\text{exponent} - 127)} \]

- **Sign Bit**
- 23 bit significand as a fraction
- 8 bit exponent as unsigned int

---

Double Precision

<table>
<thead>
<tr>
<th>11 bits</th>
<th>20 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>exponent</td>
<td>significand</td>
</tr>
<tr>
<td>11 bits</td>
<td>55 bits</td>
</tr>
</tbody>
</table>

- 32 bits

---

64 bit IEEE 754

- exponent is 11 bits
  - bias is 1023
  - range is a little larger than the 32 bit format.

- Significand is 55 bits
  - plus the leading 1.
  - accuracy is much better than 32 bit format.
Example Representations

\[ 0.75_{10} \rightarrow \frac{1}{2} + \frac{1}{4} \rightarrow 0.11 \times 2^1 \rightarrow 1.1 \times 2^{-1} \]

\[
\begin{array}{c|c|c}
\text{s} & \text{exponent} & \text{significand} \\
\hline
0 & 01111110 & 100000000000000000000000 \\
\end{array}
\]

As unsigned int is 126. Leading 1 is not stored!

What number is this?

\[
\begin{array}{c|c|c}
\text{s} & \text{exponent} & \text{significand} \\
\hline
0 & 10000001 & 110000000000000000000000 \\
\end{array}
\]

Exercises

- What is the double precision (64 bit format) representation for the number 128?
- Same question for single precision

- What is the single precision format for the number –8.125?
- Same question for double precision

Comparing Numbers

- Comparison of normalized floating point numbers:
  - check sign bits
  - check exponents.
    - unsigned integer comparison works.
    - Larger exponents are represented by larger unsigned ints.
  - check significand.