Core Java – Chapter 3 Section 3.3.2 & 3.3.3

12/1/22

exponential form

E or e represents decimal exponent e.g. $1.23E3 = 1230 = 1.23 \times 10^3$

floating point hexadecimal literals

floating point literals can be written as hexadecimals:

$2^{-3} = 0x1.0p^{-3}$

- "p" because "e" is a hex digit
- written in decimal
- base is 2, not 10

3 special values:

- Positive infinity
- Negative infinity
- Not a number (NaN)

$\forall x, x > 0 \Rightarrow \frac{x}{0} =$ Positive infinity

$\frac{0}{0} =$ NaN $\quad \sqrt{-x}$ where $x < 0 =$ NaN

Double.POSITIVE_INFINITY, Double.NEGATIVE_INFINITY, and Double.NaN are never used in practice

Trap!
NaN != NaN

All NaN values are distinct, so $x == $ Double.NaN won't ever be true.
Use Double.isNaN().

BigDecimal for financial calculations

Use BigDecimal for exact computations w/o roundoff errors that floating point has.
→ e.g. financial calculations

3.3.3 char type

\u0000 to \uffff

char != full character (not always) anyway

some characters use 1 char, but many need 2 chars to be represented.
char = code unit. code point can require either 1 or 2 chars. (described later in chapter in 3.3.4)

single quotes 'A', not "A"
- 'A' → char
- "A" → string

escapes

special escapes aside from \u, can be used in either strings or char literals

\b	\t	\n	\r	\f	\"	\'	\\	\s	\<newline>
backspace	tab	newline	carriage return	form feed	double quote	single quote	backslash	space	in text blocks only, join this line with the next

\u escape can be used outside strings or chars, directly in the Java syntax ⇒ public static void main(String\u005B\u005D args)

Hacky, retain trailing whitespace in text blocks

Question: can you check for infinity like this: $x == $ Double.POSITIVE_INFINITY
How do the \s and \<newline> escapes work?