Design recipe with \texttt{cond}

... no big change, but a little extra advice
Examples

When the problem statement divides the input into several categories, test each one

Example:

Write the function **line-part** that determines whether a number is on zero, to the left, or to the right on a number line

\[
\begin{align*}
\text{(check-expect } (\text{line-part } 0) & \text{ "zero")}
\text{(check-expect } (\text{line-part } -3) & \text{ "left")}
\text{(check-expect } (\text{line-part } 3) & \text{ "right")}
\end{align*}
\]
Body

When the problem statement divides the input into $N$ categories:

- Start the body with a `cond` expression and $N$ lines
- Formulate a question to recognize each category

Example:

Write the function `line-part` that determines whether a number is on zero, to the left, or to the right on a number line

Three cases, so three lines:  

```
(define (line-part n)
  (cond
    [(= n 0) ...]
    [(< n 0) ...]
    [(> n 0) ...]))
```