Built up from characters and the empty string \( (\varepsilon \text{ or } \lambda) \) via the operations \(+\) (also written \( \cup \text{ or } |\)), \( \cdot \), and \(*\):

- this + that: this or that
- this \( \cdot \) that: this followed by that
- \((\text{this})^*\): zero or more occurrences of this.

**Examples:**

\[(a + b) \cdot (a + c) = aa + ac + ba + bc \quad (0 + 01) \cdot (10 + 0) = 010 + 00 + 0110\]

\[(a + bc)^* = \{\varepsilon, a, bc, aa, abc, bcab, bca, aaa, \ldots\}\]

But not \(bac\) for instance.

\[(00)^* = \{\varepsilon, 00, 0000, \ldots\} = \{0^n : n \text{ is even}\}\]

\[(11)^*1 = \{1, 111, 11111, \ldots\} = \{1^n : n \text{ is odd}\} \quad 1(11)^* \text{ is equivalent.}\]

Now how about strings over \(\{0, 1\}\) containing an odd number of 1s?

Try even 1s first: \((0^*10^*1^*)^*\). Fixing the even case, use \((0^*10^*1^*)^*0^*\).

Then add a 1 to make it odd: \((0^*10^*1^*)^*1\). Is that good? **Sound? Comprehensive?**

(needs to allow ending in 0s) Note incidentaly that \(0^*0^* = 0^*\).

Economical is: \((0^*10^*1^*)^*0^*10^*\).

This was not comprehensive, did not match 0, 1.

How about \(\{x \in \{0, 1\}^* : \text{every 5th char of } x \text{ from the first is a 1}\}\)? We can try \((1(0 + 1)^4)^*\).

But this forces the string to have length a multiple of 5. To allow other lengths, try:

\[\left(1(0 + 1)^4\right)^*(\varepsilon + 1(e + 0 + 1)(e + 0 + 1)(e + 0 + 1))\] [Will pause for why it works.]
Now how do we apply these ideas to make a regular expression for Wed.'s language \( L(M_{5,2}) = \{ x : x \text{ has an odd # of 1s in positions } \equiv 2 \mod 5 \} \)?

First, we need at least 3 chars, to get at least one 1 in such a position. The first two such chars are arbitrary: \((0 + 1)^2\). Then we see the equation:

\[
L(M_{5,2}) = (0 + 1)^2 \cdot L(M_{5,0})
\]

Thus we can focus on "blocks" of the form \( Z = 0(0 + 1)^4 \) or \( I = 1(0 + 1)^4 \). Take our previous "template" for an odd number of 1's and sub. 0 by \( Z \), 1 by \( I \):

\[
L(M_{5,0}) = (ZIZ^*)I^*ZIZ^*
\]

But this has another "overkill" problem: The last 1 in a multiple-of-5 position need not be followed by 4 chars. So instead define \( Y = (0 + 1)^40 \). Then:

\[
L(M_{5,2}) = (0 + 1)^2 \cdot (ZIZ^*)^*Z^*1Y^*(\epsilon + 0 + 1)^4.
\]

\[
= (0 + 1)^2 \cdot \left( (0(0 + 1)^4)^*1(0 + 1)^4 \left( (0(0 + 1)^4)^*1(0 + 1)^4 \right)^* \left( (0(0 + 1)^40)^* (\epsilon + 0 + 1)^4 \right) \right).
\]

Yuck---?---! We got it by top-down reasoning---but maybe there's a better way...

Here is the DFA \( M_{5,0} \) that was referred to, from the first-day lecture in 2021:
If we start this machine up in state $q_3$ then we get $M_{5,2}$: the machine either gets just zero or one char and accepts, or it gets two chars corresponding to the initial $(0 + 1)^2$ and then goes into the same machinations as $M_{5,0}$. 