Theorem: For any language $L$ over a finite alphabet $\Sigma$, the following statements are equivalent:

1. There is a DFA $M = (Q, \Sigma, \delta, s, F)$ such that $L = L(M)$
2. There is an NFA $N$ s.t. $L = L(N)$
3. There is a regular expression $r$ s.t. $L = L(r)$

Proof:

Base case: $\emptyset$ is a regex, $L(\emptyset) = \emptyset$. $N_{\emptyset} = \emptyset$.

For any char $c \in \Sigma$:

- $c$ is a regex, $L(c) = \{c\}$. $N_c = \rightarrow c \rightarrow$.

For any regex $r$, $L(r) = L(\epsilon \cup r) = L(\epsilon) \cup L(r)$. By induction hypothesis, there are NFAs $N_a$ and $N_b$ such that $L(N_a) = L(a)$ and $L(N_b) = L(b)$. Build $N_r = L(a \cdot b)$ is a regex with $L(N_r) = L(a) \cdot L(b)$.

For any regex $r$, $L(r) = L(\epsilon \cup r)$ is a regex with $L(N_r) = L(\epsilon) \cup L(N_r)$.

Parallel circuit: $L(N_r) = L(N_a) \cup L(N_b)$.

Series circuit: $L(N_r) = L(N_a) \cdot L(N_b)$.
The lecture included a long demo of the "Turing Kit" software (a still-working JAR file from 1997). Here is a drawing of the DFA that was shown:

\[ \Sigma = \{ 0, $, D \} \] meaning "empty room," "spear," and "dragon."

- Start
- No Spear
- No Spear
- Dead
- Spear
- Spear
- Spear
- Spear

\[ L(M) = \{ x \in \Sigma^* : x \text{ represents a "dungeon" in which the player exists alive} \} \]

The regular expression "means" that to come back to the start state without being killed, either we see an empty room (0) or pick up a spear ($). We can't pick up any more spears, and the only way to be back in the no-spear state is if — after zero or more rooms with 0 or $ — we get and kill a dragon (D). This last sentence represents the $ (0+$) * D part. Since we can come back to start any number of times and stay alive, what we will soon call \[ L = \{ (0+$) (0+$) * D \} * \] left the dungeon ends so we exit there (E), fine. Or if we pick up a spear and act any number of dragon-free rooms, then we exit with a spear in hand (\$ (0+D) *).