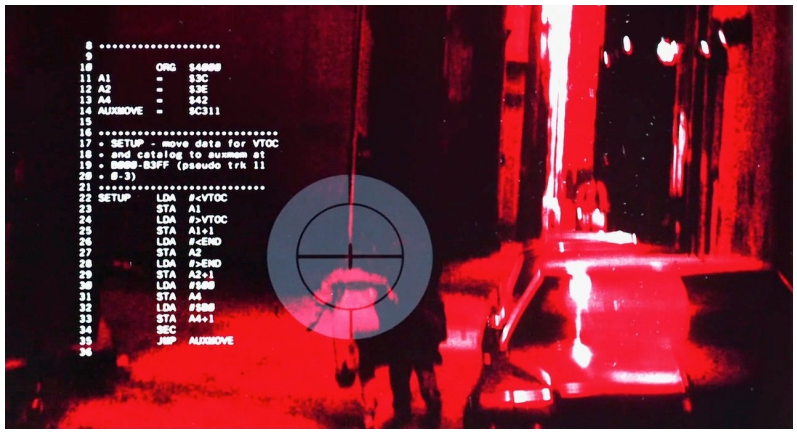
The Terminator

One option is to write a function like `die()`, which you saw in earlier lessons, and which prints an error message and then terminates. This is not really a good solution for runtime errors unless they are very severe, since it **doesn't give the user a chance to recover**. Imagine if your Web browser **shut down** every time you typed a URL incorrectly or clicked on a dead link.



Using a function like `die()` does prevent the program from continuing with garbage values, (which is good!), but it is simply too drastic and **too inflexible** to be a good universal approach.

However, there is one time when a "terminator" **is the correct way** to handle errors:

- When you are developing your code and ...
- When the error is a programming problem that you can fix.

In fact, the C++ library has a **built-in macro** which does this, called `assert()`.