Perhaps there are some design patters that you've been noticing since we started talking about Unix tools, and now we're going to discuss them explicitly. Unix tools were designed along a set of guidelines which are best summarized by Ken Thompson's idea that each Unix program should **do one thing well**. Following this rule when writing functions and programs accomplished several goals:

- Limiting a program to only doing one thing reduces the length of the program, and the shorter a program is the easier it is to fix if it contains bugs or if it needs to be revised.
- Writing short programs also helps the users of your code understand what's going on in your code in the event that they need to read your code. Reading a poem induces a different cognitive load compared to reading a novel.
- Folks who don't read the source code of your program (most users won't - they shouldn't have to) will be able to understand the inputs, outputs, and side effects of your program more easily.
- Using small programs to write a new program will increase the likelihood that the new program will also be small. **Composability** is the concept of stringing small programs together to create a new program.

The concept of composability in Unix is best illustrated by the use of the pipe operator (|) for creating pipelines of programs. When you're considering what inputs your program is going to have and what your program is going to print to the console you should consider whether or not your program might be used in a pipeline, and you should organize your program accordingly.

In the previous section we discussed the difference between functions that compute values and functions that produce side effects. You should notice that the side effect functions like mv and cp do not print any text to the console if they are successful. The concept of *quietness* is another important part of the Unix philosophy. Quietness in this case means that a function should not print to the console unless it is necessary, either to inform the user of a value (pwd), to display the result of a computation (bc), or to warn the user that an error has occurred.

Mark as completed