Lab 6

- streams usage
  - stream parameter types should match what the method wants
    - avoid attaching nozzles to an existing stream within a method
  - flush after writing

Always remember to `flush()` an output stream or writer after you write to it — this writes out anything that might have been buffered. You don't need to `flush()` after every individual thing that is written, but `flush()` after writing a batch of things if you aren't going to write again for a while.

    - some stream operations flush on their own, and closing the stream typically flushes – but flushing after writes is a good habit because it leads to hard-to-find problems if you forget when it is needed
  - close streams where they were opened
    - the hoses created in `main` should be closed in `main`
    - streams are generally closed automatically when the program ends, but closing when they are no longer needed is a good habit – some systems may limit the number of open streams

Lab 6

- exception handling
  - streams operations can generate errors – `IOException`, `FileNotFoundException`, etc
  - exceptions should be caught where they can be handled
    - the `grep`, `copy` functions are not responsible for their parameters so aren't equipped to properly handle problems with the streams – pass the exception up to the caller with a `throws` declaration in the method header

```
public void grep ( … ) throws IOException {
  …
}
```

    - `main` created the hoses and set up the streams (for most of the streams) – it should catch and handle the exceptions
      - `main` should not declare `throws` – then exceptions go unhandled and crash the program
      - print an information user-friendly error message, not a blank or a stack trace