CS 344 Notes
1.1 → 1.3

"<" and ">" wrt stdin, out
- ">" "specifies" the output location so that when program writes to stdout, it goes there.
- "<" specifies what to be loaded into the stdin file to be then read by the program.

"?" specifies a single unknown / wildcard character.

"|" (Pipe) operator
- redirects output (stdout) of one command to the stdin of another

">>" Operator
- appends data to a file.

Shebang == "#!"
- used to give path to shell, i.e.:
- #! /bin/bash

Quotes
- Single Quotes: NO variable expansion
- double Quotes: Variables are expanded

"\" Backslash
- means evaluate literally, don't interpret

Variables
- Show forbidden variables with 'compgen -k'
- environment variables - consistent across shells ~

Special Script Variables
- $: PID of process of script
- ?: return of previous command / script
- #: num of arguments passed
- 1 or 2 ... : Parameters

Errors & Bash Scripts
- use #! bin/bash -e so that next command only runs if the previous one exited with 0

For loops
- "for i in a b c d
  do
    printf "<%s>" $i
  done"

While loops
- "while test ...
  do
  ...
  done"

Tests
- 'test' can be used, interchangeably with [cond.]

Subshells & Expr
- `` (backticks) can surround a command to be run as a subshell
- $(command) also does that, and is POSIX.
- $((expres)) is the same as expr(...)
- ((i++)) standalone.

CS344 Notes
1.4

'read' command

- reads input line by line, echoing it out or placing it into a variable
- while read myLine

    do

        echo "$ MyLine"

    done $\underbrace{< \$1}_{\text{stdin file name}}$

Arrays

- defined: array=(1␣2␣3)
- gotten array [n] or [@] for all

for loop from a "line"

- you can submit a tab or space delineated string of values to for so that it iterates over those, i.e.

    for i in "1   2   3"

    ...

Cut Command

- cut -c choses column n of file
- cut -f respects white space