Chapter 7: Input and Output

- Input and output are not part of the C language itself.

7.1 Standard Input and Output

- The library implements a simple model of text input and output.
- A text stream consists of sequences of lines.
  - each line ends with a newline character.
- The simplest input mechanism is to read one character at a time from the standard input.
  - use `int getchar(void)`
    - returns the next input character each time it is called or EOF.
    - They symbol EOF is defined in
      - typically has value -1.
      - Tests using EOF should be value independent.
- It is possible to substitute a file for the keyboard using redirection.
  - `prog < infile`
  - causes prog to read characters from the infile instead of the keyboard.
  - the <infile portion is not included in the command line args.
    - switching is done independent of the program.
- Input switching is invisible if the input comes from another program via a pipe or the command line (on some systems).
  - `otherprog | prog`
  - runs the two programs otherprog and prog piping the standard output of otherprog ito the standard input of prog.
- The output of the program can be redirected as well.
  - `prog > file`
- Many programs read only one input stream and write only one output

stream.
    - the functions getchar, putchar, and printf are adequate for these.

7.2 Formatted Output -- Printf

- The output function printf translates internal values to characters.
- printf converts, formats, and prints its args on the standard output under control of the format.
    - returns the number of characters printed.
- Format string contains two types of objects.
    - ordinary characters - copied to the output stream
    - conversion specifications - each causes conversion and printing of the next successive arg to pritnf.
        - each conversion specification starts with a % and ends with a conversion character.
        - There may be specifiers between the % and the character.
            - a minus sign - specifies left adjustment
            - a number - specifies the minimum field width.
            - a period - separates the field width from precision
            - a number - the precision specifying the max number of characters.
            - an h - if the integer is to be printed as a short.
            - an l - if the integer is to be printed as a long.
    - The width may also be specified as * and placed as an argument.

7.3 Variable-length Argument Lists

- We can write our own version of printf to take variable length args.

7.4 Formatted Input -- Scanf

- The function `scanf` is the input version of `printf`
  - provides many of the same features.
- scanf reads chars from the standard input, interprets them using the spefication format, and stores the result.
  - each argument must be a pointer.
  - the arguments show where the input is to be stored.
- scanf stops when it reaches the end of its format string or input doesn't match specifiers.
- returns the number of successfully matched specifiers that are assigned.
  - There is a difference between EOF and O
    - 0 means that a specifier didn't match input.
- sscanf reads from a string instead of standard input.
- scanf ignores blanks and tabs in its format string.
  - skips over white space.
  - If the format isn't fixed it's best to read in line by line and parse each line using sscanf
- The arguments of scanf and sscanf must be pointers.

7.5 File Access

- Before a file can be read or written it has to be opened by the library function fopen.
  - fopen takes an external name and returns a pointer to be used in reading and writing the file.
  - The pointer is called the *file pionter*
    - points to a structure that contains information about the file.
    - location of a buffer, current character position in buffer, if the file is being read or written, if errors or EOF is reached.
  - It also takes a mode of operation for the file.
    - allowable modes are: read, write, and append.

        - some systems distinguish between text and binary files.
- If a file is opened for reading and writing doesn't exist it's created if possible
- Opening a file for writing discards the old contents.
- Trying to read a file that doesn't exist is an error.
    - Also trying to read a file without permissions is an error as well.
- In the event of an error fopen will return NULL

7.6 Error Handling -- Stderr and Exit

- `stderr` is used to output errors to the screen even if stdout is redirected.
- The program signals errors two ways.
    - The diagnostic output produced by `fprintf` goes to `stderr`
    - The program then uses the standard library function `exit`
- A return of 0 is a signal that things went well.
    - Non-zero means there was an abnormal problem.
- Large programs should return useful status values.

7.7 Line Input and Output

- The standard library provides `fgets` similar to `getline`.
- `fgets` reads the next input line from a file fp into the character array line.