Objectives - 1/20/2022 (Thu)

- Test-driven development (TDD)
- VIM intermediate tips

Test-driven development

A method of development that helps you break the work into small increments that are tested as you go.

print_integer.c	~~source~~ implementation function code
print_integer.h	header function headers
expected.txt	expected output

test_print_integer.c	test code
	int main() {
	}

Vim - tabs

```
$ vim -p *.c *.h expected.txt
```

from bash

open files in tabs

-OR-

```
vim -p *.*
```

```
:tabe expected.txt
```

(OR tabedit)

AQ shortcut:

F2

VIM splits

```
$ vim -O print_integer.*
```

~~print_it~~

↑ from bash

(capital letter O)

— OR —

```
:vs print_integer.h
```

↑ from VIM

Use mouse to adjust split

vertical

```
$ vim -o print_integer.*
```

(lowercase letter o)

— OR —

```
:sp print_integer.h
```

horizontal

Vim - replace

Ugly way

```
:%s/old/new/gc
```

% — Do this to entire file.

s — substitute

g — "global" (replace every occurrence in each line)

c — "confirm" each replacement

Multiple cursors

Ctrl-N (for each occurrence)

S to substitute, i.e., start typing replacement

VIM - autocomplete

Ctrl-N

Ctrl-P

AQ shortcuts

F3

cycle through choices

F4

Example:

Pr

Then, press Ctrl-N to autocomplete to print_integer

VIM Tips

Mouse is okay (except HW01)

~~Chan~~

Replace:

```
:%s/old/new/gc
```

% entire document

s substitute

g "global" (all in each line)

c "confirm" each match

OR

Ctrl N (press for each occurrence)

S (substitute → start typing to replace)