Tutorial 3

- Side effects
- More Recursion Practice
- Reading input
- Global variables & mutation
Side effects

(there are 3 types of side effects in functions for now)

- print output
- read input
- mutate a global variable
// MANY SIDE EFFECTS

int y = 2;

int main(void) {
    int x = read_int();
    printf("%d", printf("%d", y));
    if ((x = y - 1)) {
        printf("%d", x = x + 2);
    }
    if (x) {
        y *= y;
        printf("%d", y);
    }
    printf("\n");
}
Side Effects: printing vs. returning

```c
int pure_functional(int n) {
    return n * n;
}

// effects: displays a message
void just_a_side_effect(int n) {
    printf("n squared is %d\n", n * n);
    return; // (optional)
}

// effects: displays a message
int has_both_side_effect_and_return_value(int n) {
    printf("n squared is %d\n", n * n);
    return n * n;
}
```
Recursion with side effects

Define the following C function: (use recursion)

```c
// fizzbuzz(n, fizz, buzz): produces a sequence
// from 1 ... n with the following replacements:
// Replace numbers with divisible by fizz with fizz
// Replace numbers divisible by buzz with buzz
// Replace numbers divisible by both with fizzbuzz
// Requires: 1 <= fizz, buzz, n

For example:

fizzbuzz(16, 3, 5) produces the output:

1 2 fizz 4 buzz fizz 7 8 fizz buzz 11 fizz 13 14 fizzbuzz 16.

(note the spacing and the final period)
Read Input: Exercise

Implement the following function using recursion:

// print_average(): reads in input from the user until
// READ_INT_FAIL then prints the average of
// all inputs.
// Effects:
// * reads input
// * prints an output message
Global variables & mutation

- Global variables are defined *outside* of functions (at the “top level”).
- A function that mutates a global variable **does** have a side effect.
- Even if a function does not have a side effect, its behaviour may depend on other mutable global variables.