Hannah Zhang

Checkup Problems #1

1. The function, factorial, as written below, is a recursive process.

```
(define (factorial n)
  (cond ((or (= n 1) (= n 0)) 1)
        (else (* n (factorial (- n 1))))))
```

It is a recursive process because of the space it takes up when invoked. Consider (factorial 4):

```
(factorial 4)
(* 4 (factorial 3))
(* 4 (* 3 (factorial 2)))
(* 4 (* 3 (* 2 (factorial 1))))
(* 4 (* 3 (* 2 1)))
(* 4 (* 3 2))
(* 4 6)
24
```

Write a different version of factorial that uses an iterative process. factorial should still be called the same way (eg., (factorial 4)), so you will need to write a helper function that has one more variable.

Nested function

```
(define (factorial1 n)
  (define (iter i)
    (if (= n 0)
        i
        (factorial2 (- n 1) (* i n))))
  (iter 1))
```

2. Here is a definition of accumulate:

```
(define (accumulate combiner null-value term a next b)
  (cond ((> a b) null-value)
        (else (combiner (term a)
                        (accumulate combiner null-value term
(next a) next b)))))
```

Write factorial in terms of accumulate. Here's a skeleton of part of what your answer *must* look like:

```
(define (factorial n) (accumulate <??> <??> <??> <??> <??>
<??>))
```

You have to fill in the <??>s. Any attempt to write factorial as a recursive procedure will earn a score of zero.

```
(define (same x) x)

(define (plus-one x)
  (+ x 1))
```

Combiner is multiplication because factorial takes products
Null-value is 1 in multiplication/division
Does not apply anything special to each term
Starts at 1
Each recursive call adds 1
Stops at n which is given

```
(define (factorial n)
  (accumulate * 1 same 1 plus-one n))
```