Exercise 1.4. Observe that our model of evaluation allows for combinations whose operators are compound expressions. Use this observation to describe the behavior of the following procedure:

```
(define (a-plus-abs-b a b)
  ((if (> b 0) + -) a b))
```

Answer:

In the body of `a-plus-abs-b,` the operator of the combination `((operator) a b)` is determined by `(if (> b 0) + -)`, which makes the whole expression look more elegant and compact since you won't need to write two expressions specifically for `(> b 0)` is true and `(> b 0)` is false. Example:

```
(define (a-plus-abs-b a b)
  (if (> b 0)
      (+ b a)
      (+ (- b) a)))
```

Here you can see that we wrote 2 expressions each specific to `(> b 0)` is true and `(> b 0)` is false. `(+ b a)` for when `(> b 0)` is true, and `(+ (- b) a)` for when `(> b 0)` is false. Whereas in the first procedure, we only wrote `(if (> b 0) + -)` which handles both cases.