Excercise 1.4

Josh H

January 3, 2024

We describe the behavior of the following procedure:

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

The body of the procedure is a combination, and we evaluate combinations by first evaluating their subexpressions. The leftmost subexpression `(if (> b 0) + -)` is a special form and the operands `a b` are evaluated simply. We evaluate the predicate `(> b 0)`, then based on that result we evaluate either the consequent or the alternative.

Suppose the predicate returns true, then the interpreter evaluates the consequent `+`. The value of `+` is the value of the expression `(if (> b 0) + -)` when evaluated. The body of the procedure is reduced to `(+ a b)`, where the operator is already evaluated, and the interpreter handles this primitive procedure.

Suppose the predicate return false, then the interpreter evalutes the alternative `-`. The body of the procedure reduces to `(- a b)`.