SICP Exercise 1-5

Samuel Lair

January 19, 2020

1 Introduction

Ben Bitdiddle executes the following program:

```
(define (p) (p))
(define (test x y)
    (if (= x 0)
       0
      y))
(test 0 (p))
```

Describe what happens under applicative-order and normal-order evaluation.

2 Applicative-Order Evaluation

Under applicative-order evaluation, operators and operands are evaluated before applying the resulting procedures. This poses a problem when evaluating (test 0 (p)). The definition of (p) is recursive with no terminating base case. Therefore, an applicative-order interpreter will get stuck in an infinite loop when it tries to evaluate the (p) in (test 0 (p)).

The Scheme interpreter uses applicative-order evaluation so this is what actually happens when this program is evaluated.

3 Normal-Order Evaluation

Under normal-order evaluation, operands are not evaluated until their values are needed. Therefore, (test 0 (p)) is expanded to:

```
(if (= 0 0)
    0
    (p)))
```

Since 0 equals 0, a normal-order interpreter will return 0 without evaluating (p).