SICP Chapter 1

ProducerMatt

Testing

```
(define (foo a b)
  (+ a (* 2 b)))

(foo 5 3)
```

```
11
```

^ Dynamically evaluated when you press "enter" on the `BEGIN_SRC` block!

Also consider:

- `:results output` for what the code prints
- `:exports code` or `results` to just get one or the other

$a + (\pi \times b)$ <~ inline Latex btw :)

Exercise 1.1

Q

Below is a sequence of expressions. What is the result printed by the interpreter in response to each expression? Assume that the sequence is to be evaluated in the order in which it is presented.

A

```
10 ;; 10
(+ 5 3 4) ;; 12
(- 9 1) ;; 8
(/ 6 2) ;; 3
(+ (* 2 4) (- 4 6)) ;; 6
(define a 3) ;; a=3
(define b (+ a 1)) ;; b=4
(+ a b (* a b)) ;; 19
(= a b) ;; false
```

```
(if (and (> b a) (< b (* a b)))
    b
    a) ;; 4
(cond ((= a 4) 6)
      ((= b 4) (+ 6 7 a))
      (else 25)) ;; 16
(+ 2 (if (> b a) b a)) ;; 6
(* (cond ((> a b) a)
         ((< a b) b)
         (else -1))
   (+ a 1)) ;; 16
```

Exercise 1.2

Q

Translate the following expression into prefix form:

$$\frac{5+2+(2-3-(6+\frac{4}{5}))}{3(6-2)(2-7)} \tag{1}$$

A

```
(/ (+ 5 2 (- 2 3 (+ 6 (/ 4 5))))
   (* 3 (- 6 2) (- 2 7)))
```

```
1/75
```

Exercise 1.3

Q

Define a procedure that takes three numbers as arguments and returns the sum of the squares of the two larger numbers.

```
(define (square x)
  (* x x))
```

A

```
<<square>>
(define (sum-square x y)
  (+ (square x) (square y)))
(define (square-2of3 a b c)
  (cond ((and (>= a b) (>= b c)) (sum-square a b))
        ((and (>= a b) (> c b)) (sum-square a c))
```

```
          ((and (> b a) (>= c a)) (sum-square b c))
          (else "This shouldn't happen")))

(list (square-2of3 7 5 3)
      (square-2of3 7 3 5)
      (square-2of3 3 5 7))

```

```
(74 74 74)
```