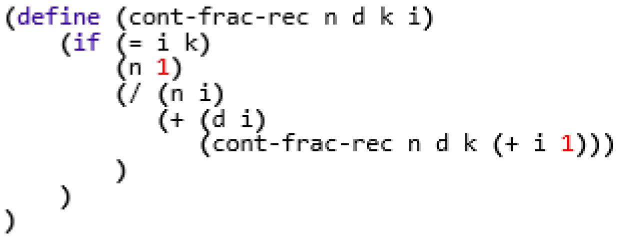
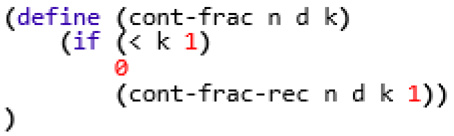
Exercise 1.38. In 1737, the Swiss mathematician Leonhard Euler published a memoir *De Fractionibus Continuis*, which included a continued fraction expansion for *e* - 2, where *e* is the base of the natural logarithms. In this fraction, the N_i are all 1, and the D_i are successively 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, ... Write a program that uses your `cont-frac` procedure from exercise 1.37 to approximate *e*, based on Euler's expansion.

Solution

First, we need to define the function cont-frac from exercise 1.37. The function may look like:



```
(define (cont-frac-rec n d k i)
    (if (= i k)
        (n 1)
        (/ (n i)
           (+ (d i)
              (cont-frac-rec n d k (+ i 1)))
        )
    )
)
```



```
(define (cont-frac n d k)
    (if (< k 1)
        0
        (cont-frac-rec n d k 1))
)
```

This could still use some refactoring (for example define cont-frac-rec in lexical scope of cont-frac - we would not have to pass parameters n d k to cont-frac-rec, because it would be closed over these. Also assigning a name to some of the constants would help). The solution above however looks readable and is almost direct translation of the formula of the continued fraction from the assignment. Function cont-frac is a wrapper that calls cont-frac-rec with correct parameters and also takes care of incorrect input parameters.

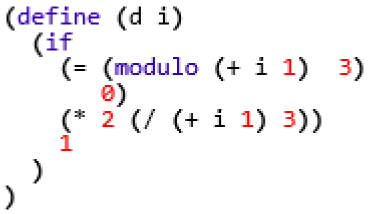
Now we need to define a function d that would return the following results for input i.

i	1	2	3	4	5	6	7	8	9	10	11
value	1	2	1	1	4	1	1	6	1	1	8

All values are 1 except for values at indexes 2,5,8,11,… These are indexes where (i +1) % 3 == 0 where % is modulo operation.

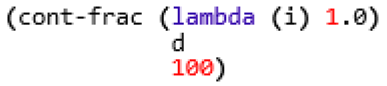
To calculate values at these indexes, I have used similar thinking. At index 2 the value is 2, at 5 => 4, at 8 => 6, at 11 => 8 etc. Therefore the formula would be $\frac{i+1}{3}$ * 2.

This can be written in Scheme as:



```
(define (d i)
  (if
    (= (modulo (+ i 1)  3)
       0)
    (* 2 (/ (+ i 1) 3))
    1
  )
)
```

And then we can test the code and compare the result to e-2



```
(cont-frac (lambda (i) 1.0)
           d
           100)
```