CSE 341 Section Handout #8
Cheat Sheet

Mutation

```
; mutation is discouraged except where necessary
(set variable expr)
(set-car! list expr)
(set-cdr! list expr)
(mcons expr mutableList)
(mcar mutableList)
(mcdr mutableList)
(set-mcar! mutableList expr)
(set-mcdr! mutableList expr)
```

Associative Lists (maps)

```
; represented as a list of pairs that represent keys/values
(list '(expr . expr) '(expr . expr) ... '(expr . expr))
(assoc key associationList)          ; returns the key/value pair for the given key
```

Delayed Evaluation

```
(delay expr)         ; returns a 'promise'
(force expr)         ; forces evaluation of a promise
```

Streams

```
; A stream is a thunk (parameterless function) that returns a pair:
; (next-value-in-list . next-thunk-to-call)

; A stream (infinite list) containing an endless list of 1s: 1, 1, 1, ...
(define ones (lambda () (cons 1 ones)))

; A stream of all natural numbers: 1, 2, 3, 4, ...
(define (nat-nums)
  (define (helper x)
    (cons x (lambda () (helper (+ x 1)))))
  (helper 1))

; A stream of all powers of two: 1, 2, 4, 8, 16, ...
(define (powers-of-two)
  (define (helper x)
    (cons x (lambda () (helper (* x 2)))))
  (helper 1))

; A stream for the harmonic series: 1, 1/2, 1/3, 1/4, ...
(define (harmonic)
  (define (helper x)
    (cons (/ 1 x) (lambda () (helper (+ x 1)))))
  (helper 1))

; Returns the first k elements of the given stream.
(define (stream-slice stream k)
  (if (<= k 0) '()
      (let ((strpair (stream)))
        (cons (car strpair) (stream-slice (cdr strpair) (- k 1))))))
```

CSE 341 Section Handout #8
Cheat Sheet

BASIC Syntax

```
; REM (comments)
REM text

; END (end of program)
END

; LET (variable declaration)
LET variable = expression

; INPUT (read a value from the console and store it into a variable)
INPUT variable

; PRINT (output to console)
PRINT (expression | string) {"," (expression | string)}

; GOTO (jump from one line to another)
GOTO lineNumber

; IF/THEN (conditionally jump from one line to another)
IF test THEN lineNumber

; GOSUB (jump from one line to another, but remember this location to RETURN back to)
GOSUB lineNumber

; RETURN (go back to the location of the most recent GOSUB call)
RETURN

; FOR (looping over a range of values)
FOR variable = expression TO expression
```

CSE 341 Section Handout #8
BASIC Examples

```
; Program #1
10 LET x = 34.8
20 LET y = x * 2 + 1
30 PRINT "x =", x, ", y =", y
40 END
```

```
; Program #2
10 for i = 1 to 10
20 print i, sqr(i)
30 next i
40 end
```

```
; Program #3
10 for i = 1 to 5
20 for j = 1 to i
30 print j, "foo"
40 next j
50 print "bar"
60 next i
70 end
```

```
; Program #4
10 print "integer?"
20 input n
30 if int(n/2)*2 = n then 60
40 print "odd"
50 goto 70
60 print "even"
70 print "done"
80 end
```

```
; Program #5
10 let x = 15.4
20 let y = 78.3
30 gosub 90
40 let x = 19.8
50 let y = 82.3
60 gosub 90
70 print "all done now"
80 goto 140
90 print "x =", x
100 print "y =", y
110 let d = sqr(x*x + y*y)
120 print "distance from origin =", d
130 return
140 end
```