CS 135 Winter 2020

Tutorial 03: Lists and Stepping

Goals of this tutorial:

By the end of this tutorial you should be able to...

- **Step** through a given piece of code with mathematical and boolean functions, and conditional statements.
- Navigate flat **lists** and lists with **one** level of nesting.
- Utilize list functions.

Clicker Question: Debugging Conditional Statements

Will the below function always produce the correct result?

```
;; remainder-by-4: Nat → Str
(define (remainder-by-4 n)
  (cond
    [(= (remainder n 4) 1) "n-1 is divisible by 4"]
    [(= (remainder n 4) 2) "n-2 is divisible by 4"]
    [else "n-3 is divisible by 4"]
```

A Yes
B No

Review: Stepping Rules

Always evaluate the **topmost, leftmost** unsimplified expression first.

Application of built-in functions: (f v1 . . . vn) ⇒ v where f is a built-in function and v is the value of $f(v_1, \ldots, v_n)$

Substitution of Constants: id ⇒ val, where (define id val) occurs previously.

Review: Stepping Rules

Application of user-defined functions: The general substitution rule is:

(f v1 . . . vn) ⇒ exp'

where (define (f x1 . . . xn) exp) occurs previously, and exp' is obtained by substituting all occurrences of the formal parameter xi replaced by the value vi (for i from 1 to n) into the expression.

Clicker Question

What are the next two steps for this code? (Do not skip any steps.)

```
(define x 5)
(define (foo a b) (+ a b x (max a (sqr b))))
(foo 1 x)
```

A ⇒ (+ 1 5 5 (max 1 (sqr 5))) ⇒ (+ 1 5 5 (max 1 25))

B ⇒ (+ 1 5 x (max 1 (sqr 5))) ⇒ (+ 1 5 5 (max 1 (sqr 5)))

C ⇒ (foo 1 5) ⇒ (+ 1 5 5 (max 1 (sqr 5)))

D ⇒ (foo 1 5) ⇒ (+ 1 5 x (max 1 (sqr 5)))

Clicker Question

The following definitions have been processed:

```
(define x 10)
(define y (+ x x))
```

what are the next two steps for this code?

```
(+ y y)
```

A ⇒ (+ (+ x x) y) ⇒ (+ (+ 10 x) y)

B ⇒ (+ (+ 10 x) y) ⇒ (+ (+ 10 10) y)

C ⇒ (+ 20 20) ⇒ 40

D ⇒ (+ 20 y) ⇒ (+ 20 20)

Review: Stepping Rules

Substitution in cond expressions

There are three rules: when the first expression is false, when it is true, and when it is else.

(cond [false exp] . . .) ⇒ (cond . . .)

(cond [true exp] . . .) ⇒ exp

(cond [else exp]) ⇒ exp

These suffice to simplify any cond expression, note the error case too:

(cond [false exp]) ⇒ (cond) ⇒ ERROR

Group Problem - Stepping cond

The following have been processed in the Beginning Student language:

```
(define x 1)
(define y 1)
```

Step through the following:

```
(cond [(= x 0) 'one]
      [else (< (/ y x) c)])
```

Review: Stepping Rules

Simplification Rules for and and or

The simplification rules we use for Boolean expressions involving and and or differ from the ones the Stepper in DrRacket uses.

(and false ...) ⇒ false

(and true ...) ⇒ (and ...)

(and) ⇒ true

(or true ...) ⇒ true

(or false ...) ⇒ (or ...)

(or) ⇒ false

Group Problem - Stepping and

The following have been processed in the Beginning Student language:

```
(define x 0)
(define y (+ x 1))
```

Step through the following:

```
(and (not (= x 0)) (<= (/ y x) c))
```


Review: Lists

A list is a recursive structure, meaning it is made by combining a value with another list.

Think of Russian dolls:

- A list of 3 dolls, is a single doll with 2 dolls inside of it.
- A list of 2 dolls, is a single doll with 1 doll inside of it.
- A list of 1 doll, is a single doll with nothing inside of it.
- A list of 0 dolls is special. It is an empty list however, it is still a list.

Review: Making Lists

In beginner language you can make lists using the (cons value list) function. Lists can take in any data type as show in the example below.

```
;; A (listof Any) is one of:
;; * empty
;; * (cons Any (listof Any))
(define doll-set (cons "blue" (cons 'green (cons true empty))))
```

However, lists can also be restricted to take in very specific data.

```
;; A (listof Int) is one of:
;; * empty
;; * (cons Int (listof Int))
(define no-scope (cons 3 (cons 6 (cons 0 empty))))
```


Problem 0: second, third, and fourth

Using only first and rest, define functions my-second, my-third and my-fourth that do the same thing as the built-in functions second, third, and fourth. You don't have to write the entire resign recipe, but include contracts.

Clicker: Accesing an element in a list

Which of the following options will produce 'kittens from lst?

```
(define lst (cons 'raindrops (cons 'roses (cons (cons 'whiskers (cons 'kittens empty))
        (cons 'kettles (cons 'mittens (cons (cons 'paper-packages (cons 'strings empty))
        empty)))))))
```

A (first (rest (rest lst)))

B (first (rest (first (rest (rest lst)))))

C (first (rest (rest (rest (rest lst)))))

D (first (fourth lst))

E None of the above

Problem 1: 3D → 2D

You have been provided a set of 3-dimensional vectors in the form (cons x (cons z (cons y empty))).

Define a function that consumes a vector and produces a plane of the X and Y Values.

Problem 1: 3D → 2D - Design Recipe

```
;; (change-dimension vector) Consumes a vector and produces a plane.
;; change-dimension: (listof Num) → (listof Num)
;; requires: vector must be a 3 element list
;; Example:
(check-expect (change-dimension (cons 1 (cons 2 (cons −3 empty)))
                                (cons 1 (cons −3 empty))))

(define (change-dimension vector). . . )
;; Tests:
(check-expect (change-dimension (cons 0 (cons 2 (cons .5 empty)))
                                (cons 0 (cons .5 empty))))
(check-expect (change-dimension (cons 1 (cons 2 (cons −3 empty)))
                                (cons 1 (cons −3 empty))))
```


Extra Practice: Stepping

We will not cover these questions in the tutorial, unless we have some time left over in the end. Solutions will not be posted to these questions - you may discuss these problems with classmates and work together to come up with a solution.

Complete the **practice questions** on the Stepping webpage of the course practice. Try Modules 2a, 2b, 2c, 4a, and 4b.

Extra Practice: Triangle Touch

We will not cover these questions in the tutorial, unless we have some time left over in the end. Solutions will not be posted to these questions - you may discuss these problems with classmates and work together to come up with a solution.

For this question you will be working with a Cartesian graph ranging from 0 to 10 in both the X and Y direction. A triangle will be formed of the three points (cons 0 (cons 0 empty)), (cons 10 (cons 0 empty)), and a third coordinate entered by the use as the first parameter of the function. Write a function called “in-triangle” that consumes one point to complete a triangle with the points (0, 0) and (10, 0). Then consumes a second target point. if the the second point is on or within the area of the triangle produce true, otherwise produce false.

Triangle Touch: Hint 1

The first challenge we face when approaching this problem is how do we make a triangle? It might help up to think of this triangle as two slopes that meet at the X coordinate of peak, since regardless of where we put this point it will always result in two right-angle triangles. With this in mind, we can tackle a smaller portion of the large question, how do we form a slope from (make-posn 0 0) to (make-posn peak-x peak-y)?

Triangle Touch: Hint 2

Now the second problem we face is how to determine if a point falls within the area of the slope? Since we know exactly what Y value will be produced given your new found slope, we can just compare the point-y value to the y-value produced from running the slope with the point-x value. if the point-y value is greater than the produced value, then you know it is not in or on the slope’s area, otherwise you know it **is** within the slope’s area.