Subject: Exercise 1  
Posted by jarobinson3 on Thu, 29 Sep 2016 05:58:52 GMT  
View Forum Message <> Reply to Message

Are we required to construct a lambda? For instance I could construct some code using cons and symbols and call eval on it. Would that be acceptable?

Subject: Re: Exercise 1  
Posted by lusth on Thu, 29 Sep 2016 11:13:15 GMT  
View Forum Message <> Reply to Message

Eval is an expensive operation, so you should only call eval a constant number of times.

Subject: Re: Exercise 1  
Posted by apluth on Tue, 04 Oct 2016 21:17:42 GMT  
View Forum Message <> Reply to Message

for the function call to iterator should there be a function passed in as a third parameter such as (iterator i (1 2 3 4) (lambda (x) (some code here))) or is there a specific function that we should be building inside the body of iterator

Subject: Re: Exercise 1  
Posted by lusth on Tue, 04 Oct 2016 21:25:38 GMT  
View Forum Message <> Reply to Message

The iterator is variadic, with all arguments after the list of elements to iterate over comprising the body of the iterator. Construct a list that looks like a lambda wrapping the body expressions, then turn that list into a closure using eval. Then map over the list elements using the closure as the mapping function.

Subject: Re: Exercise 1  
Posted by apluth on Tue, 04 Oct 2016 21:32:16 GMT  
View Forum Message <> Reply to Message

OK that makes sense a was a little confused after today's lecture but I think I got it we would use one of the latter call-by method in the parameter parsing section

Subject: Re: Exercise 1  
Posted by apluth on Tue, 04 Oct 2016 22:09:00 GMT  
View Forum Message <> Reply to Message
how do we delay the evaluation of "i" upon calling iterate function when I run the test script I get "undefined variable i"

Subject: Re: Exercise 1
Posted by lusth on Tue, 04 Oct 2016 22:39:50 GMT
View Forum Message <> Reply to Message

The signature of your iterator function should be something like:

(define (iterator # $x items $)

The # captures the calling environment, the $x captures the loop variable, and the $ captures the loop body.

Subject: Re: Exercise 1
Posted by apluth on Tue, 04 Oct 2016 22:58:10 GMT
View Forum Message <> Reply to Message

ok i was doing this (define (iterate # $i lst @)

Subject: Re: Exercise 1
Posted by lusth on Wed, 05 Oct 2016 01:11:41 GMT
View Forum Message <> Reply to Message

The $i is fine. The @ needs to be $ since you want to delay those arguments.

Subject: Re: Exercise 1
Posted by jrmelton on Thu, 06 Oct 2016 00:06:36 GMT
View Forum Message <> Reply to Message

I believe my code is correct (for the most part) but when I run it in the terminal, I get this error and I'm not sure why - I thought cons was a built-in function? I'm using it exactly the same way we were shown in class on Tuesday.

EXCEPTION: nonFunction, attempted to call CONS as a function

Subject: Re: Exercise 1
Posted by tmurphy2 on Thu, 06 Oct 2016 02:21:27 GMT
View Forum Message <> Reply to Message
You might have an extra set of parentheses somewhere?

(cons 'x 'y) works

while

*)((cons 'x 'y)) gives the error

---

Subject: Re: Exercise 1
Posted by jrmelton on Thu, 06 Oct 2016 20:13:25 GMT
View Forum Message <> Reply to Message

What should we read to better understand assignment 2 as a whole?

---

Subject: Re: Exercise 1
Posted by Iusth on Thu, 06 Oct 2016 20:23:41 GMT
View Forum Message <> Reply to Message

The tasks are based upon the chapter 2 material (including exercises), plus the ability of Scam to define your own special forms (via delayed evaluation).

---

Subject: Re: Exercise 1
Posted by jrmelton on Thu, 06 Oct 2016 23:01:20 GMT
View Forum Message <> Reply to Message

Task 1 states: "You are to define an iterator loop that binds each item in a list, in turn, to a variable. With a binding accomplished, the loop executes its body." Are we given the body or should we just hard code it?

It also states: "For each item in the list, construct an appropriate lambda and evaluate it in the calling environment." Are we supposed to create a new lambda for each item in the list? Wouldn't we just want to call that lambda (which we've already evaluated to be an anonymous function) with each item in the list? Also, aren't we only supposed to call eval once? This says to do it for each item in the list.

---

Subject: Re: Exercise 1
Posted by Iusth on Thu, 06 Oct 2016 23:15:55 GMT
View Forum Message <> Reply to Message

The body consists of all the expressions beyond the first two arguments.

You are correct about the lambda, so I've updated the task description to read:
Construct an appropriate lambda-like list and then evaluate the lambda-like list in the calling environment. Use the resulting closure to process each item in the second argument.

Subject: Re: Exercise 1
Posted by jrmelton on Fri, 07 Oct 2016 00:53:56 GMT
View Forum Message <> Reply to Message

Thank you! That makes perfect sense now! One last question, are we required to use the eval function you wrote in class or is fine to use the built-in eval function?

Subject: Re: Exercise 1
Posted by lusth on Fri, 07 Oct 2016 11:40:51 GMT
View Forum Message <> Reply to Message

The eval function I wrote in class is for your DPL. It does mimic the Scam eval function pretty much exactly, though.

Subject: Re: Exercise 1
Posted by jrmelton on Fri, 07 Oct 2016 13:54:57 GMT
View Forum Message <> Reply to Message

oh ok thank you!