Can we assume that the function definition will be of the form

\begin{verbatim}
(define (id ...) ...)
\end{verbatim}

and not

\begin{verbatim}
(define id (lambda (...) ...)
\end{verbatim}

?

Also, just to clarify, any internal function definition will be of the form

\begin{verbatim}
(define square (lambda (x) (* x x)))
\end{verbatim}

and not

\begin{verbatim}
(define (square x) (* x x))
\end{verbatim}

right?

For your first my understanding is that you will always have a function of the form

\begin{verbatim}
(define (name args)
 ; defines
\end{verbatim}
Also the result should be

(define (name args)
  ((lambda ( ... ) execution) a1 a2 a3 ...)
)

Subject: Re: Task 4
Posted by padiel on Sun, 16 Oct 2016 00:02:36 GMT
View Forum Message <> Reply to Message

I don't think that's correct. It says that "Note: Scam's local defines may refer to previous local definitions and are processed sequentially."

So that means that we have to have nested lambdas right?

Subject: Re: Task 4
Posted by luseth on Sun, 16 Oct 2016 19:20:31 GMT
View Forum Message <> Reply to Message

Yes, if you do not nest your lambdas, you will fail some test cases (not a majority, though).

Subject: Re: Task 4
Posted by padiel on Sun, 16 Oct 2016 19:33:43 GMT
View Forum Message <> Reply to Message

So Dr J, if one of the inner defines is a function definition can we assume that it's of the form

(define a (lambda ...

And not

(define (a ...) (...
That's what the spec says.

New question for problem 4: If a defined value is never used, do we need to create a lambda for it? For example, the function (define (test x) (define y (+ 1 x)) (+ x x)) just returns (+ x x) and never uses y. I wrote my code assuming that we could eliminate that local define by just excluding it, but the test case seems to want a lambda over y that would return (+ x x) for all inputs.

Never mind, I just changed how I was implementing it to match the test cases.