Subject: Task 10
Posted by jzarob on Sat, 20 Feb 2016 04:06:45 GMT
View Forum Message <> Reply to Message

I'm having issues with the test script after redefining + to be non-variadic in task 10. Specifically, I'm having issues when test script when evaluating Task2 because addition, multiplication, and subtraction have all been redefined to be nonvariadic. The test script calls the curry function with std + and multiple operands, which is resulting in an exception. I just want to be absolutely clear on how to proceed.

Subject: Re: Task 10
Posted by lush on Sat, 20 Feb 2016 16:47:44 GMT
View Forum Message <> Reply to Message

Use old+ if you want plus to be variadic.

Subject: Re: Task 10
Posted by jzarob on Sat, 20 Feb 2016 21:27:38 GMT
View Forum Message <> Reply to Message

The test script that you provided calls Task 2 with the + with 3 arguments, but it get redefined to be nonvariadic with 2 arguments in task 10 and calling apply throws an exception.

I know to use old+ if I want the old functionality, I'm just unsure how to approach the test script with the redefinition of plus.

Subject: Re: Task 10
Posted by amsainju on Sat, 20 Feb 2016 22:22:34 GMT
View Forum Message <> Reply to Message

I think there is a problem in test2.scm
In question 2,
(define (f a b c d e)
  (old+ a b c d e))

(define (run2)
  (exprTest (curry f 2 3 4 5 6) (f 2 3 4 5 6)) gives 20
  (exprTest ((curry f 2) 3 4 5 6) (f 2 3 4 5 6)) gives 20
  (exprTest (((curry f 2 3) 4) 5 6) (f 2 3 4 5 6)) gives 20
  (exprTest (((curry f 2) 3) 4) 5 6) (f 2 3 4 5 6)) gives 20
  (exprTest ((((curry f 2 3) 4) 3) 4) 5 6) (f 2 3 4 5 6)) gives 20
)

my test case is working fine.
but for the other testcase provided in test2.scm it says too many arguments to function +
(inspect (define (f a b c) (+ a b c))) here + should be replaced with old+.

Subject: Re: Task 10
Posted by lusth on Sat, 20 Feb 2016 23:01:23 GMT
View Forum Message <> Reply to Message

jzarob wrote on Sat, 20 February 2016 15:27 The test script that you provided calls Task 2 with the+
+ with 3 arguments, but it get redefined to be nonvariadic with 2 arguments in task 10 and calling
apply throws an exception.

I know to use old+ if I want the old functionality, I'm just unsure how to approach the test script
with the redefinition of plus.

Edit the test script to call old+.

When we do the real testing, will will use the original value of +, which we can get a hold of with
the expression (context'+).

Subject: Re: Task 10
Posted by jmhossler on Tue, 23 Feb 2016 20:39:03 GMT
View Forum Message <> Reply to Message

Would it be okay to convert all integers passed in to bigInts, and then run the corresponding big
operation on those? Or are we suppose to optimize the computations by only running the big
operations when overflow is inevitable?

Subject: Re: Task 10
Posted by jmhossler on Wed, 24 Feb 2016 22:19:33 GMT
View Forum Message <> Reply to Message

Also, can we make our +,-, and * variadic?

Subject: Re: Task 10
Posted by ffoo on Sat, 27 Feb 2016 16:52:51 GMT
View Forum Message <> Reply to Message

Do we have to modify Q 1-9 replacing + with old+ etc?
When using (__context'+) I am getting an about to many arguments. Is it actually calling our redefined + after, which does not need to be variadic, and am worrying over nothing?