Racket Programming Assignment #4: Lambda and Basic Lisp

What's It All About?

The first task pertains to lambda functions. The remaining three are intended merely to better acquaint you with material presented in Racket Lesson #5 on basic Lisp processing.

Overall Charge

Generate a solution document template that is consistent with the accompanying solution template. Then, working within the DrRacket PDE, please do each of the tasks, adding source code and demos to your template in the appropriate manner.

Task 1 - Lambda

Task 1a - Three ascending integers

Consider the following demo:

```
> ( asc 5 )
'(5 6 7)
> ( asc 0 )
'(0 1 2)
> ( asc 108 )
'(108 109 110)
>
```

Your job is to generate this exact demo, except that you must replace the call to the `asc` function in each of the three applications with a **lambda** function. Note that the Definitions area will not come into play in this exercise, nor will you create any named functions. Your demo will simply feature three anonymous function applications, the first with argument 5, the second with argument 0, and the third with argument 108.

Task 1b - Make list in reverse order

Consider the following demo:

```
> ( mlr 'red 'yellow 'blue )
'(blue yellow red)
> ( mlr 10 20 30 )
'(30 20 10)
> ( mlr "Professor Plum" "Colonel Mustard" "Miss Scarlet" )
'("Miss Scarlet" "Colonel Mustard" "Professor Plum")
>
```

Your job is to generate this exact demo, except that you must replace the call to the `mlr` function in each of the three applications with a **lambda** function. Note that the Definitions area will not come into play in this exercise, nor will you create any named functions. Your demo will simply feature three anonymous function applications, each using the same three arguments that I used in the given demo.

Task 1c - Random number generator

Consider the following demo:

```
> ( rn 3 5 )
5
> ( rn 3 5 )
3
> ( rn 3 5 )
5
> ( rn 3 5 )
5
> ( rn 3 5 )
3
> ( rn 3 5 )
4
> ( rn 3 5 )
3
> ( rn 3 5 )
3
> ( rn 3 5 )
5
> ( rn 3 5 )
3
> ( rn 11 17 )
17
> ( rn 11 17 )
12
> ( rn 11 17 )
14
> ( rn 11 17 )
12
```

```
> ( rn 11 17 )
12
> ( rn 11 17 )
14
> ( rn 11 17 )
16
> ( rn 11 17 )
14
> ( rn 11 17 )
16
> ( rn 11 17 )
13
>
```

Your job is to generate demo like this one, except that you must replace the call to the `rn` function in each of the three applications with a **lambda** function. Note that the Definitions area will not come into play in this exercise, nor will you create any named functions. Your demo will simply feature ten anonymous function applications with arguments 3 and 5, and ten anonymous function applications with arguments 11 and 17.

Contribution to the solution document

For the section of your solution document that corresponds to this part of your assignment, please include:

1. The first demo.
2. The second demo.
3. The third demo.

Task 2 - List Processing Referencers and Constructors

Simply create the demo, for real, that is presented in redacted form in Lesson 5 “Basic Lisp Programming”, in the section titled “Redacted Racket Session Featuring Referencers and Constructors”.

Contribution to the solution document

For the section of your solution document that corresponds to this part of your assignment, please simply include the required demo.

Task 3 - The Sampler Program

This task involves typing the Sampler program from Lesson 5 into a file and generating the accompanying demo. More specifically, please do the following two things:

1. In a file called `sampler.rkt`, establish the "sampler" program of Lesson 5.
2. Generate a demo by mimicking the demo which accompanies the "sampler" program.

Contribution to the solution document

For the section of your solution document that corresponds to this part of your assignment, please include:

1. The "sampler" code from Lesson 5.
2. The accompanying demo.

Task 4 - The Card Playing Example

This task involves typing the card playing code from Lesson 5 into a file and generating the accompanying demo. More specifically, please do the following two things:

1. In a file called `cards.rkt`, establish the card playing code presented in Lesson 5.
2. Generate a demo by mimicking the demo which accompanies the card playing program.

Contribution to the solution document

For the section of your solution document that corresponds to this part of your assignment, please include:

1. The card playing code from Lesson 5.
2. The accompanying demo.

Due Date

Thursday, March 9, 2023

But note: It would be best for you to do the essential part of this assignment (code/demo) prior to the exam.