Racket Assignment #1: Getting Acquainted with Racket/Dr. Racket + LEL Sentence Generation

Abstract:

In this assignment, we will use Racket to create a sentence generator to understand and introduce myself to the Programming Language. The code below will give a full understanding of how the Racket syntax and environment work.

CODE FOR THE ASSIGNMENT

```
#lang racket

( define (pick list)
   ( list-ref list ( random ( length list) ) )
   )

( define ( noun)
   ( list ( pick '(robot baby todler hat dog) ) )
   )

(define (verb)
  ( list (pick '( kissed hugged protected chased hornswoggled) ) )
  )

(define (article)
  ( list (pick ' (a the) ) )
  )

(define ( qualifier )
  (pick ' ( (howling) (talking) (dancing)
                      (barking) (happy) (laughing)
                      () () () () () ()
                      )
        )
  )

(define (noun-phrase)
  (append (article)(qualifier) (noun))
  )
(define (sentence)
  (append (noun-phrase) (verb) (noun-phrase) )
  )
(define (ds) ; display a sentence
  (map
   (lambda (w) (display w) (display " "))
   (sentence)
   )
  (display "");an artificial something
  )

```

DEMO FOR THE LEL SENTENCE GENERATOR

```
Welcome to DrRacket, version 8.7 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (pick '( red yellow blue) )
'blue
> (pick ' (red yellow blue))
'red
> (pick ' (red yellow blue))
'yellow
> (pick' (red yellow blue))

'yellow
> (pick' (Racket ProLog Haskell Rust) )
'Racket
> (pick' (Racket ProLog Haskell Rust) )
'Haskell
> (pick' (Racket ProLog Haskell Rust) )
'Racket
> (pick' (Racket ProLog Haskell Rust))
'ProLog
> (noun)
'(todler)
> (noun)
'(baby)
> (noun)
'(todler)
> (noun)

'(baby)
> (verb)
'(chased)
> (verb)
'(kissed)
> (verb)
'(protected)
> (verb)
'(protected)
>
(article)
'(the)
> (article)
'(a)
```

```
> (article)
'(a)
> (article)
'(the)
>
(qualifier)
'(talking)
> (qualifier)
'()
> (qualifier)
'()
> (qualifier)
'()
> (qualifier)
'()
> (qualifier)
'(laughing)
> (qualifier)
'(barking)
> (qualifier)
'(talking)
> (qualifier)
'(talking)
> (qualifier)
'(laughing)
> (qualifier)
'(barking)
> (qualifier)
'()
> (qualifier)
'(talking)
> (qualifier)
'()
> (qualifier)
'()
> (qualifier)
'(talking)
> (noun-phrase)
'(the talking baby)
> (noun-phrase)
'(a baby)
```

```
> ( sentence )
'(the talking baby protected the howling todler)
> ( sentence )
'(the happy baby kissed a todler)
> ( sentence )
'(the laughing todler kissed the robot)
> ( sentence )
'(the happy robot hornswoggled the laughing todler)
> (ds)
the hat hugged the robot
> (ds)
the robot hugged a howling dog
> (ds)
a dog hornswoggled a dog
> (ds)
a baby protected a todler
> (ds)
a baby hornswoggled the dog
> (ds)
a happy baby hugged a robot
> (ds)
the robot chased the baby
> (ds)
a howling baby hugged a todler
> (ds)
a talking hat protected the happy baby
> (ds)
the howling dog protected the robot
> (ds)
a howling todler hugged a happy hat
> (ds)
the barking robot hornswoggled the dog
>
```