Declan ONUNKWO 02/04/2023

CSC344 – Programming Languages

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

ABSTRACT

The aim of this first program is to familiarize oneself with Racket/DrRacket. In order to achieve this, a simple “Hello…” program is run first, followed by an LEL sentence generator—LEL which is short for Little English Language. The program generates random words and sentences which the user can define, or the programmer has defined. The program was consciously written and ran to ensure full understanding of the Racket syntax and environment. The code and demo of the LEL sentence generator is placed in this document for visual reference.

CODE FOR THE LEL SENTENCE GENERATOR

lel_generator.rkt ▾ (define ...) ▾

```
#lang racket

;-----------------------------------------------------
; LEL sentence generator, with helper PICK,
; several applications of APPEND, several
; applications of LIST, and one use of MAP
; with a LAMBDA function.

( 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 ) )
'red
> ( pick '( red yellow blue ) )
'red
> ( pick '( red yellow blue ) )
'yellow
> ( pick '( red yellow blue ) )
'red
> ( pick '( Racket Prolog Haskell Rust ) )
'Haskell
> ( pick '( Racket Prolog Haskell Rust ) )
'Rust
> ( pick '( Racket Prolog Haskell Rust ) )
'Rust
> ( pick '( Racket Prolog Haskell Rust ) )
'Prolog
> ( noun )
'(dog)
> ( noun )
'(baby)
> ( noun )
'(todler)
> ( noun )
'(baby)
> ( verb )
'(kissed)
> ( verb )
'(chased)
> ( verb )
'(hornswoggled)
> ( verb )
'(hugged)
```

```
> ( article )
'(the)
> ( article )
'(the)
> ( article )
'(a)
> ( article )
'(a)
> ( qualifier )
'()
> ( qualifier )
'(howling)
> ( qualifier )
'()
> ( qualifier )
'(happy)
> ( qualifier )
'()
> ( qualifier )
'(talking)
> ( qualifier )
'(talking)
> ( qualifier )
'()
> ( qualifier )
'(howling)
> ( qualifier )
'()
> ( qualifier )
'()
> ( qualifier )
'(barking)
> ( qualifier )
'()
> ( qualifier )
'(howling)
> ( qualifier )
'(talking)
> ( qualifier )
'(talking)
```

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