Racket Programming Assignment# 1: First Interactions(by Aaradhya Acharya)

Learning Abstract

This assignment features relatively simple interactions in the Racket programming language. In fact, all the computations take place within the interactions pane of the DrRacket PDE. In the first part of this assignment, I learned a little bit about numeric computations in Lisp. The next two parts of the assignment featured a square tile which was blue except for a centered red dot. In the second part of the assignment, I mimicked the solution of the problem of finding the area of the tile which was blue. In the third part I mimicked the computational rendering of the tile. The last two parts of the assignment featured an image consisting of 5 concentric circles. In the fourth part of this assignment, I rendered the image. In the fifth part, I computed a percentage based on the concentric circles image. Throughout the problem-solving parts of this assignment the concept of binding values to variables was a predominant theme.

Interaction : Simple Numeric Processing

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> x
x: undefined;
 cannot reference an identifier before its definition
> 55
55
> 55.2
55.2
> pi
3.141592653589793
> (* 3 8)
24
> ( + (* 3 8) 6)
30
> (expt 2 8)
256
> (* pi (expt 7 2))
153.93804002589985
> (expt 9 50)
515377520732011331036461129765621272702107522001
>
```

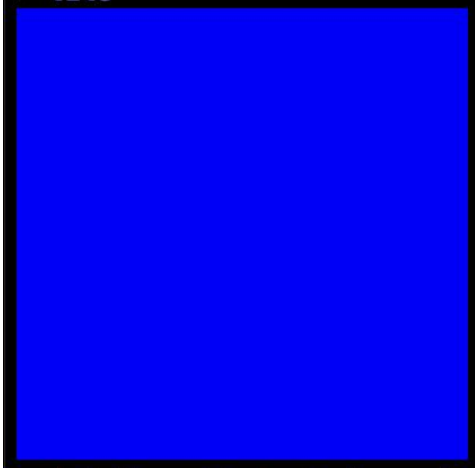
Interaction : Solution to the blue and red area interaction problem

The blue and red tile area problem: A tile of side 200 is blue, except for a centered red disk of radius one-third the side of the tile. What is the area of the tile which is blue?

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> ( define side-of-tile 200 )
> ( define diameter-of-dot ( / side-of-tile 3 ) )
> ( define radius-of-dot ( / diameter-of-dot 2 ) )
> ( define total-tile-area ( expt side-of-tile 2 ) )
> ( define red-dot-area ( * pi ( expt radius-of-dot 2 ) ) )
> ( define blue-tile-area ( - total-tile-area red-dot-area)
> side-of-tile
200
> diameter-of-dot
66 2/3
> radius-of-dot
33 1/3
> total-tile-area
40000
> red-dot-area
3490.658503988659
> blue-tile-area
36509.341496011344
>
```

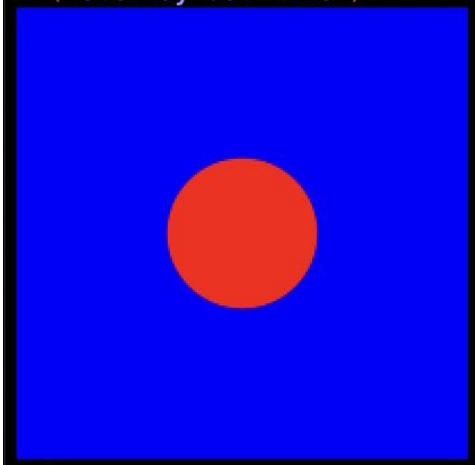
Interaction : Painting the blue and red tile

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> ( require 2htdp/image )
> ( define side-of-tile 200 )
> ( define diameter-of-dot ( / side-of-tile 3 ) )
> ( define radius-of-dot ( / diameter-of-dot 2 ) )
> ( define tile ( square side-of-tile "solid" "blue" ) )
> tile
```



```
> ( define dot ( circle radius-of-dot "solid" "red" ) )
> dot
```

```
> ( overlay dot tile )
```



```
> |
```

Interaction : Painting the blue and red concentric disks image

A concentric circles image was created by overlaying a blue disk of radius 20 on top of a red disk of radius 40 on top of a blue disk of radius 60 on top of a red disk of radius 80 on top of a blue disk of radius 100.

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (require 2htdp/image)
> (define radius-one 20)
> (define radius-two (+ radius-one radius-one))
> (define radius-three (+ radius-two radius-one))
> (define radius-four (+ radius-three radius-one))
> (define radius-five (+ radius-four radius-one))
> (define dot-one (circle radius-one 'solid 'blue))
> (define dot-two (circle radius-two 'solid 'red))
> (define dot-three (circle radius-three 'solid 'blue))
> (define dot-four (circle radius-four 'solid 'red))
> (define dot-five (circle radius-five 'solid 'blue))
> (overlay dot-one dot-two dot-three dot-four dot-five)
```

```
>
```

Interaction : Computing the area of concentric disks image which is blue

```
Welcome to DrRacket, version 8.6 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> (define radius-one 20)
> (define radius-two (+ radius-one radius-one))
> (define radius-three (+ radius-two radius-one))
> (define radius-four (+ radius-three radius-one))
> (define radius-five (+ radius-four radius-one))
> (define area-one (* pi (expt radius-one 2)))
> (define area-two (* pi (expt radius-two 2)))
> (define area-three (* pi (expt radius-three 2)))
> (define area-four (* pi (expt radius-four 2)))
> (define area-five (* pi (expt radius-five 2)))
> (define blue-area (- (+ area-five area-three area-one) (+ area-four area-two)))
> area-one
1256.6370614359173
> area-two
5026.548245743669
> area-three
11309.733552923255
> area-four
20106.192982974677
> area-five
31415.926535897932
> blue-area
18849.555921538755
> (define blue-percentage ( * 100 (/ blue-area area-five)))
> blue-percentage
59.999999999999986
>
```