First Racket Programming Assignment Solution

Learning Abstract:

In this Racket assignment I learned about structuring numeric sequences and defining constants within the language, as well as practicing using these numeric sequences to compute values. I also learned how to define and then display shapes within the Interactions pane of DrRacket.

Interaction: Simple Numeric Processing

```
> 5
5
> 5.3
5.3
> ( * 3 10 )
30
> ( + ( * 3 10 ) 4 )
34
> ( * 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 )
12157665459056928801
>
```

Interaction: Solution to the Scrap Problem

The Scrap Problem: A circular disk of maximal size is cut from a square piece of tin of side 100 units. What is the area of the scrap?

```
> pi
3.141592653589793
> side
side: undefined;
 cannot reference an identifier before its definition
> ( define side 100 )
> side
100
> ( define square-area ( * side side ))
> square-area
10000
> ( define radius ( / side 2 ))
> radius
50
> ( define circle-area ( * pi radius radius ))
> circle-area
7853.981633974483
> ( define scrap-area ( - square-area circle-area ))
> scrap-area
2146.018366025517
>
```

Interaction: Illustration of the Scrap Problem Situation

```
> ( require 2htdp/image )
> ( define side 100)
> ( define the-square ( square side "solid" "silver" ))
> the-square
```

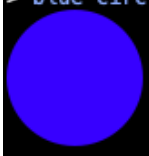
```
> ( define radius ( / side 2 ))
> ( define the-circle ( circle radius "solid" "white" ))
> ( define the-image ( overlay the-circle the-square ))
> the-image
```

```
> |
```

Interaction: Illustration of the Target Problem Situation

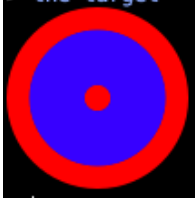
```
> ( require 2htdp/image )
> ( define big-red-radius 50 )
> ( define big-red-circle ( circle big-red-radius "solid" "red" ))
> big-red-circle
```

```
> ( define blue-radius ( * big-red-radius 3/4 ))
> ( define blue-circle ( circle blue-radius "solid" "blue" ))
> blue-circle
```



```
> ( define small-red-radius ( * big-red-radius 1/7 ))
> (define small-red-circle ( circle small-red-radius "solid" "red" ))
> small-red-circle
```

```
> ( define the-target ( overlay small-red-circle blue-circle big-red-circle ))
> the-target
```



```
> |
```

Interaction: Solution to Target Problem

```
> ( define whole-radius 50 )
> ( define whole-area ( * pi whole-radius whole-radius ))
> whole-area
7853.981633974483
> ( define blue-radius ( * whole-radius 3/4 ))
> ( define blue-area ( * pi blue-radius blue-radius ))
> blue-area
4417.864669110647
> ( define small-red-radius ( * whole-radius 1/7 ))
> ( define small-red-area ( * pi small-red-radius small-red-radius ))
> small-red-area
160.285339468867
> ( define outer-ring-area ( - whole-area blue-area ))
> outer-ring-area
3436.116964863836
> ( define total-red-area ( + outer-ring-area small-red-area ))
> total-red-area
3596.402304332703
> ( define red-percentage ( * ( / total-red-area whole-area ) 100 ))
> red-percentage
45.79081632653061
>
```