Racket Assignment #2: Interactions, Definitions, Applications

Cameron Francois
Feburary 21, 2023
CSC 344

Abstract:
This assignment affords you an opportunity to do some relatively simple Racket programming. You will perform various interactions, write a number of function definitions, and engage in computational problem solving, bits of which feature the reuse of code, imaginative constructions, and the reconfiguration of existing code.

Task 1: Interactions - Scrap of Tin:

Arithmetic Expressions:

```
1 #lang racket
2
```

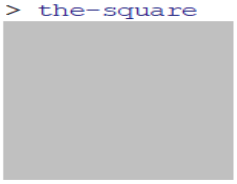
```
Welcome to DrRacket, version 8.3 [cs].
Language: racket, with debugging; memory limit: 128 MB.
> 5
5
> 5.3
5.3
> ( * 3 10 )
30
> ( + ( * 3 10 ) 4 )
34
> ( * 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 )
12157665459056928801
>
```

Solve a Simple Problem (Area of 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
>
```

Rendering an Image of the Problem Situation:

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

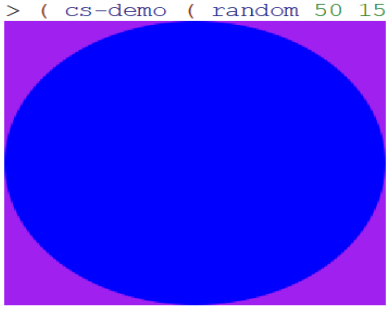


```
> ( define radius ( / side 2 ) )
> ( define the-circle ( circle radius "solid" "white" ) )
> ( define the-image ( overlay the-circle the-square ) )
> the-image
```

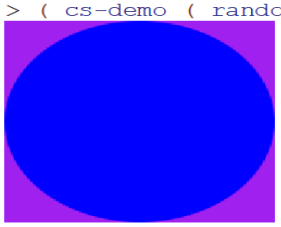
Task 2: Definitions - Inscribing/Circumscribing Circles/Squares:

cs-demo:

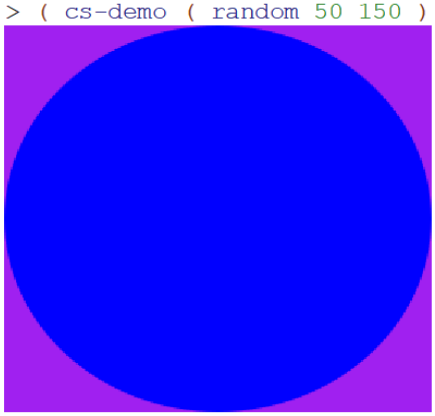
```
> ( cs-demo ( random 50 150 ) )
```



```
> ( cs-demo ( random 50 150 ) )
```

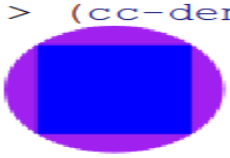


```
> ( cs-demo ( random 50 150 ) )
```

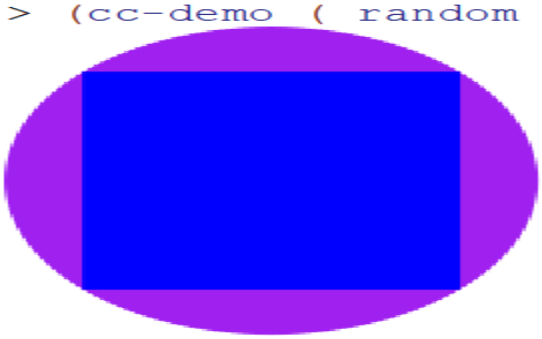


cc-demo:

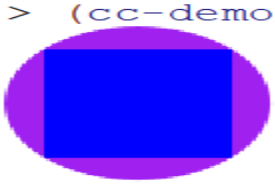
```
> (cc-demo ( random 50 150 ) )
```



```
> (cc-demo ( random 50 150 ) )
```



```
> (cc-demo ( random 50 150 ) )
```



ic-demo:

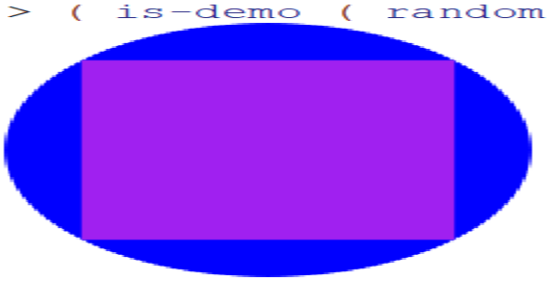
```
> ( ic-demo ( random 50 150 ) )
```

```
> ( ic-demo ( random 50 150 ) )
```

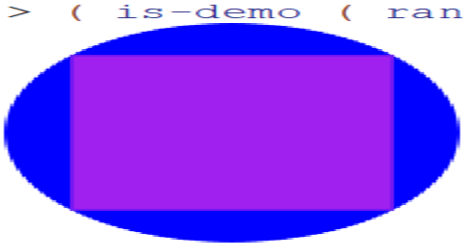
```
> ( ic-demo ( random 50 150 ) )
```

is-demo:

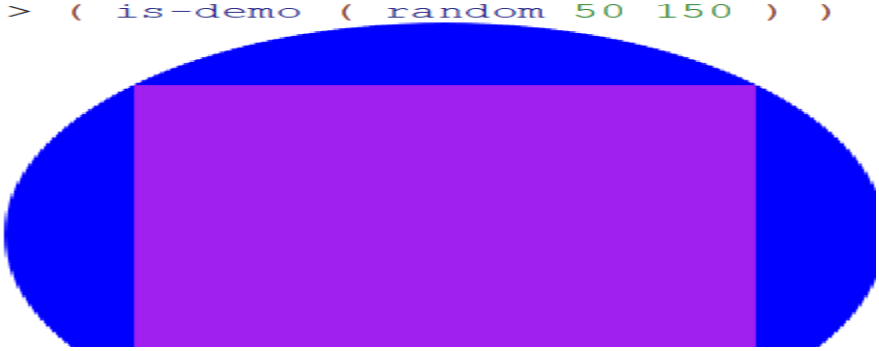
```
> ( is-demo ( random 50 150 ) )
```



```
> ( is-demo ( random 50 150 ) )
```



```
> ( is-demo ( random 50 150 ) )
```



The Code:

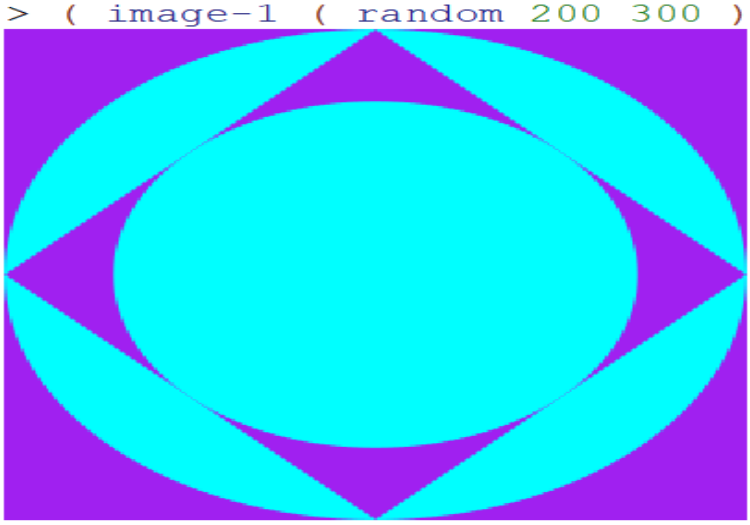
```
#lang racket
( require 2htdp/image )

( define ( cs radius )
   ( * 2 radius )
)

( define ( cc side-length )
   ( / side-length ( sqrt 2 ) )
)

( define ( ic side-length )
   ( / side-length 2 )
)

( define ( is radius )
   ( * radius ( sqrt 2 ) )
)

( define ( cs-demo radius )
   ( define sqr ( square ( cs radius ) "solid" "purple" ) )
   ( define cir ( circle radius "solid" "blue" ) )
   ( overlay cir sqr )
)
```

```
25
26  ( define ( cc-demo side-length )
27     ( define sqr ( square side-length "solid" "blue" ) )
28     ( define cir ( circle ( cc side-length ) "solid" "purple" ) )
29     ( overlay sqr cir )
30  )
31
32  ( define ( ic-demo side-length )
33     ( define sqr ( square side-length "solid" "blue" ) )
34     ( define cir ( circle ( ic side-length ) "solid" "purple" ) )
35     ( overlay cir sqr )
36  )
37
38  ( define ( is-demo radius )
39     ( define sqr (square ( is radius ) "solid" "purple" ) )
40     ( define cir ( circle radius "solid" "blue" ) )
41     ( overlay sqr cir )
42  )
```

Task 3: Inscribing/Circumscribing Images:

Image 1 Demo:

```
> ( image-1 ( random 200 300 ) )
```



```
> ( image-1 ( random 200 300 ) )
```

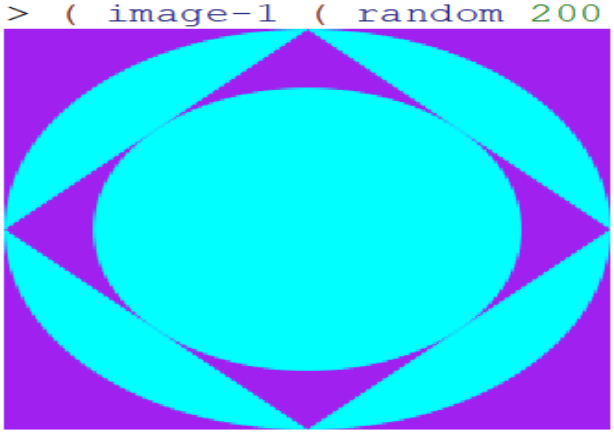
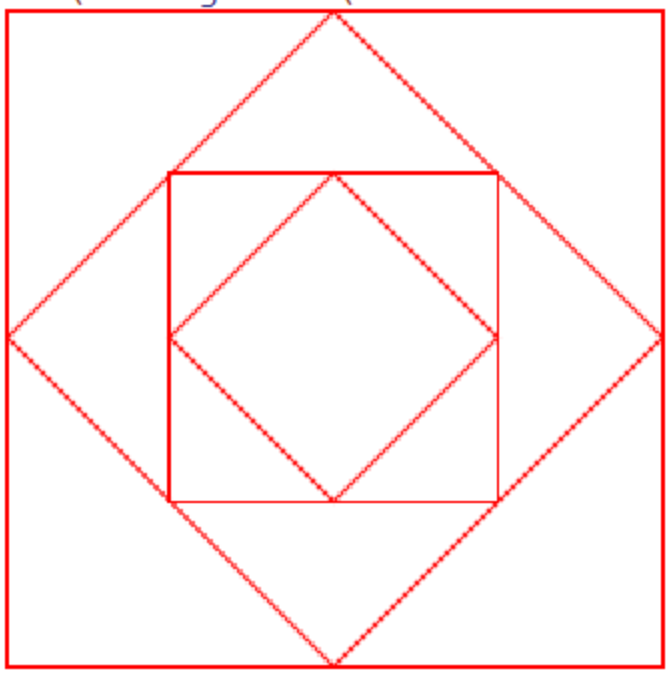
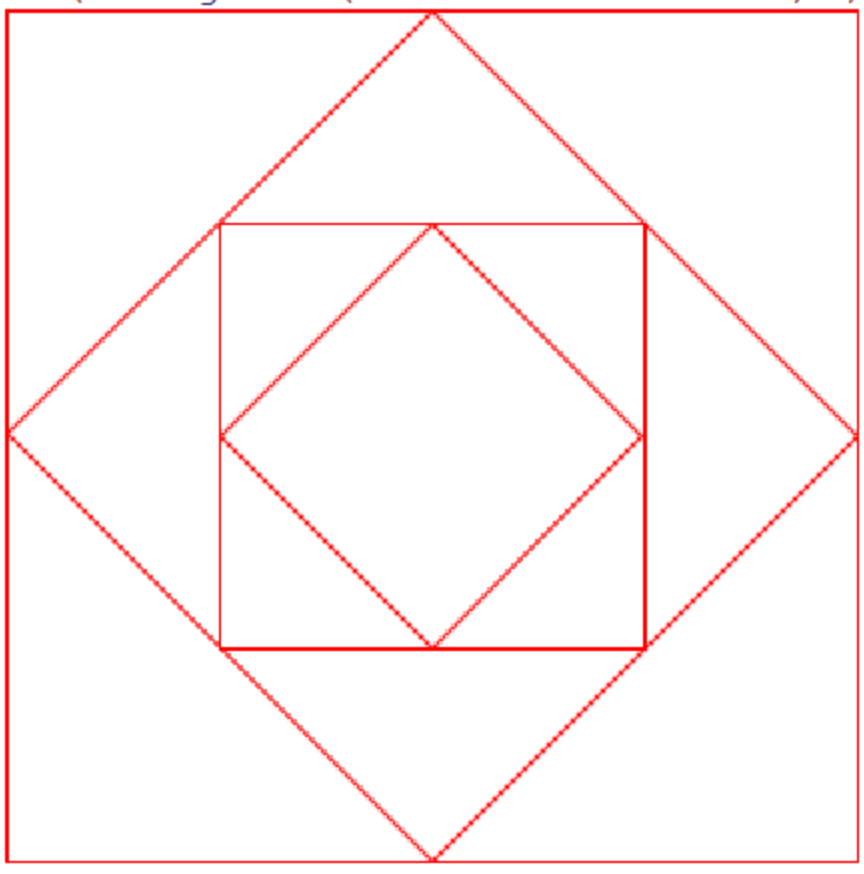


Image 2 Demo:

```
> ( image-2 ( random 200 300 ) )
```

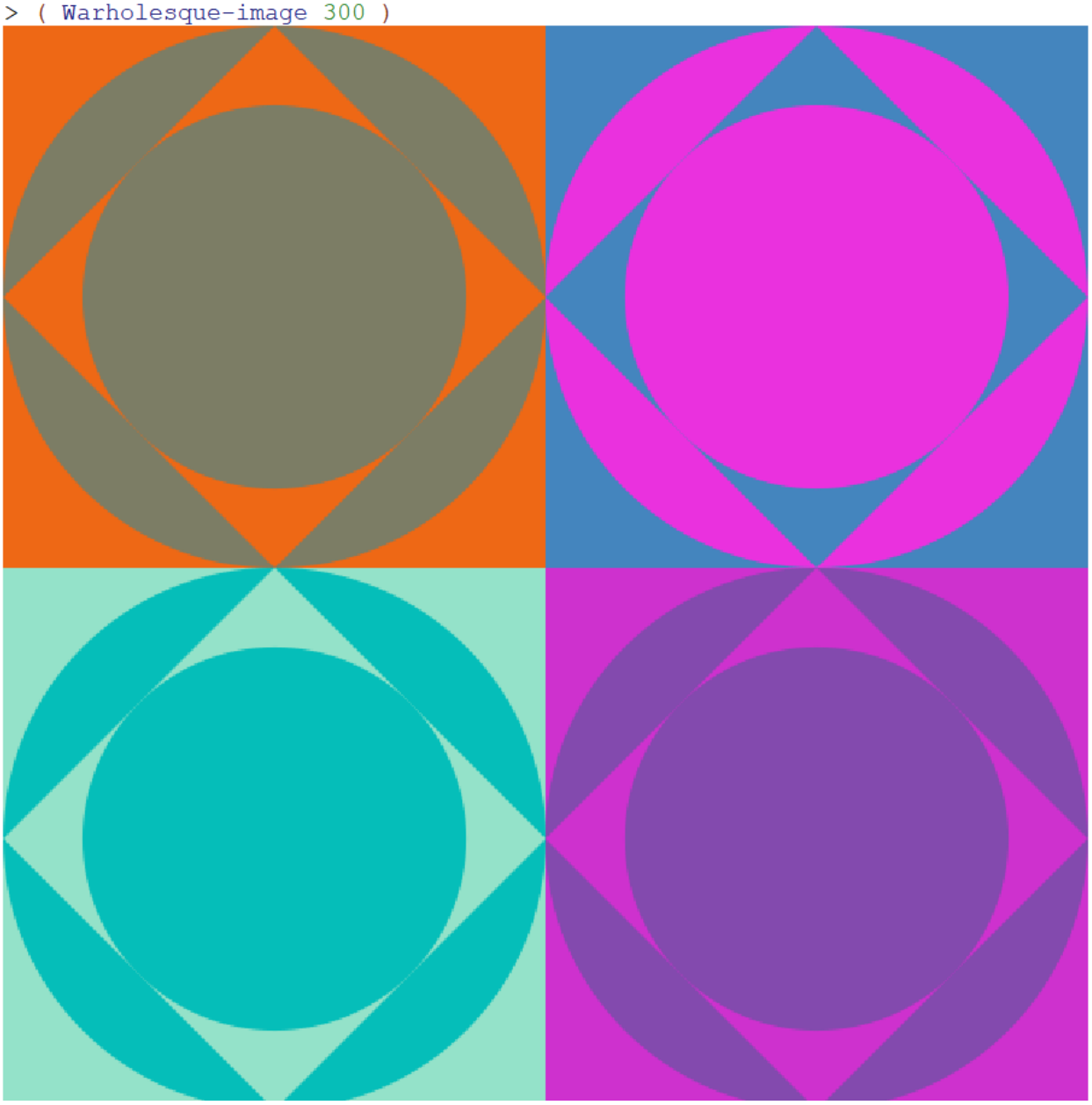


```
> ( image-2 ( random 200 300 ) )
```

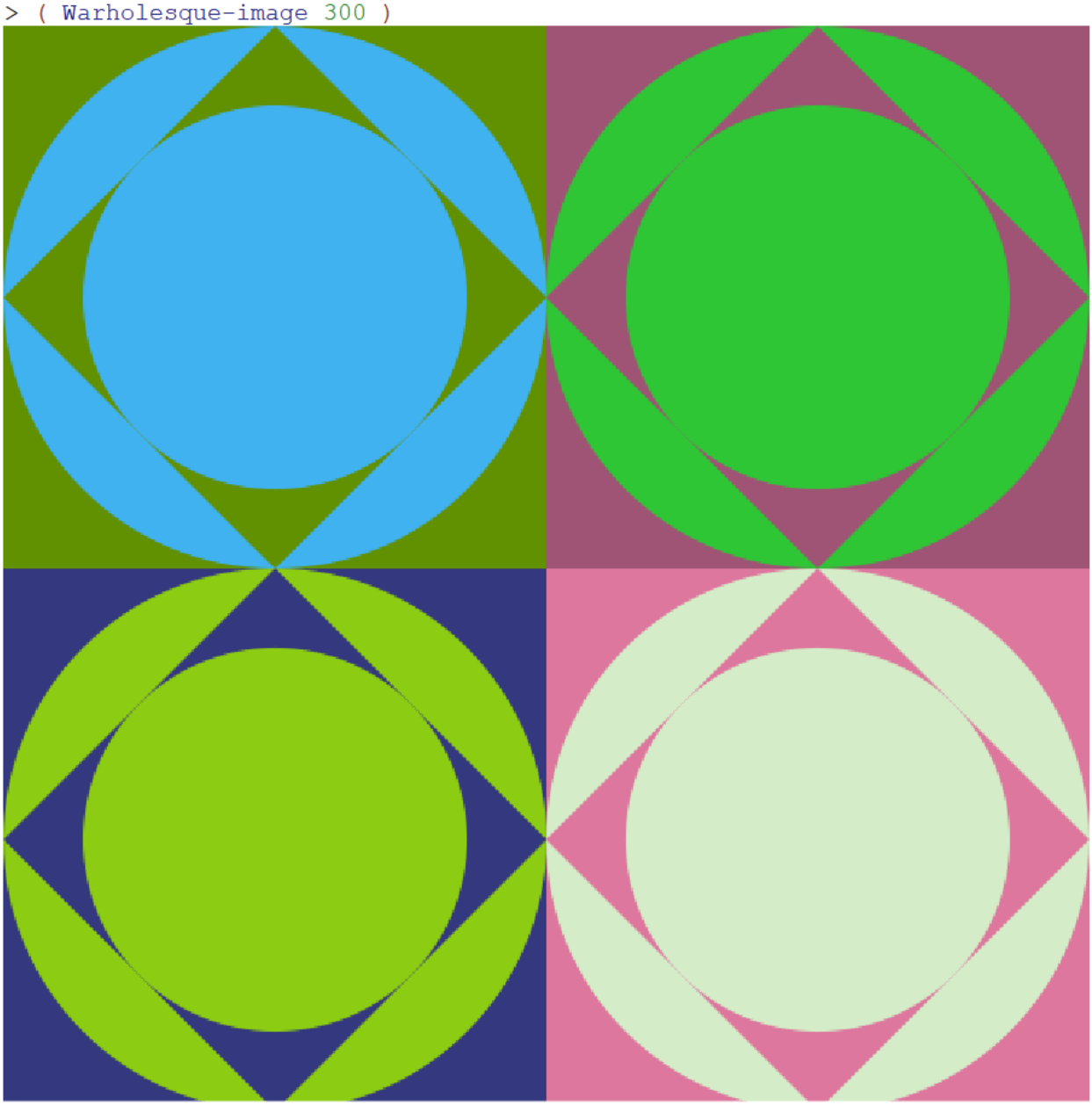


Warholesque Image:

```
> ( Warholesque-image 300 )
```



```
> ( Warholesque-image 300 )
```



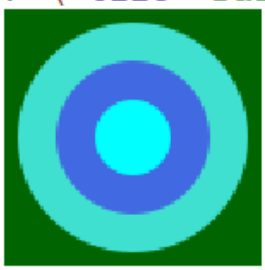
The Code:

```
( define ( image-1 side )
  ( overlay
    ( circle ( ic ( is ( ic side ) ) ) "solid" "cyan" )
    ( rotate 45 ( square ( is ( ic side ) ) "solid" "purple" ) )
    ( circle ( ic side ) "solid" "cyan" )
    ( square side "solid" "purple" )
  )
)

( define ( image-2 side )
  ( overlay
    ( square side "outline" "red" )
    ( rotate 45 ( square ( cc side ) "outline" "red" ) )
    ( square ( cc ( cc side ) ) "outline" "red" )
    ( rotate 45 ( square ( cc ( cc ( cc side ) ) ) "outline" "red" ) )
  )
)

( define ( rgb-val ) ( random 256 ) )
  ( define ( rdm-color )
    ( color ( rgb-val ) ( rgb-val ) ( rgb-val )
    )
  )

( define ( Warholesque-image-func side )

    ( define rdm-one ( rdm-color ) )
    ( define rdm-two ( rdm-color ) )
    ( overlay
      ( circle ( ic ( is ( ic side ) ) ) "solid" rdm-one )
      ( rotate 45 ( square ( is ( ic side ) ) "solid" rdm-two ) )
      ( circle ( ic side ) "solid" rdm-one )
      ( square side "solid" rdm-two )
    )
  )

( define ( Warholesque-image side )
    ( above
      ( beside
        ( Warholesque-image-func side )
        ( Warholesque-image-func side )
      )
     ( beside
       ( Warholesque-image-func side )
       ( Warholesque-image-func side )
     )
    )
  )
```

Task 4: Permutations of Randomly Colored Stacked Dots:

Demo:

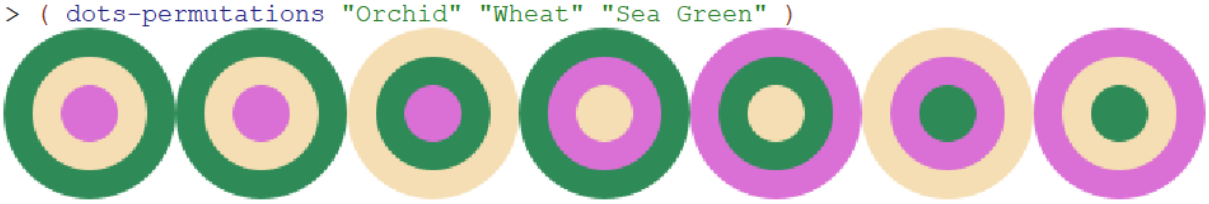
```
> ( tile "Dark Green" "Turquoise" "Royal Blue" "Aqua" )
```



```
> ( tile "Medium Slate Blue" "Navy" "Ghost White" "Tomato" )
```

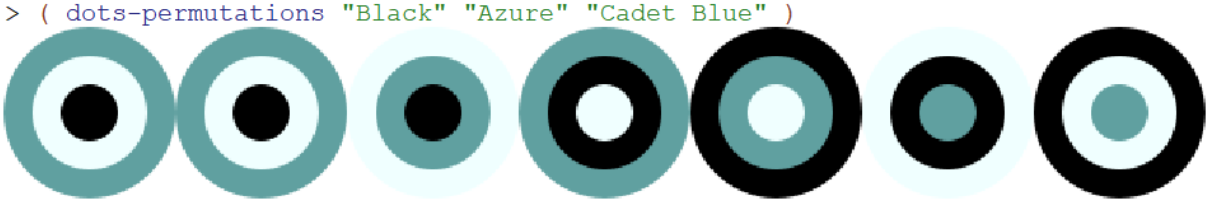
```
> ( dots-permutations "red" "Ghost White" "Red" )
```

```
> ( dots-permutations "Orchid" "Wheat" "Sea Green" )
```



```
> ( dots-permutations "Light Steel Blue" "Indigo" "Plum" )
```

```
> ( dots-permutations "Black" "Azure" "Cadet Blue" )
```



Code:

```
#lang racket
( require 2htdp/image )

( define ( tile c1 c2 c3 c4 )
   ( overlay ( circle 15 "solid" c4 )
   ( overlay ( circle 30 "solid" c3 )
   ( overlay ( circle 45 "solid" c2 )
             ( square 100 "solid" c1 )
           )
         )
     )
  )

( define ( dot-perm c1 c2 c3 )
   ( overlay ( circle 15 "solid" c1 )
   ( overlay ( circle 30 "solid" c2 )
             ( circle 45 "solid" c3 )
           )
      )
 )

( define ( dots-permutations c1 c2 c3 )
   ( beside ( dot-perm c1 c2 c3 )
    ( beside ( dot-perm c1 c2 c3 )
     ( beside ( dot-perm c1 c3 c2 )
      ( beside ( dot-perm c2 c1 c3 )
       ( beside ( dot-perm c2 c3 c1 )
        ( beside ( dot-perm c3 c1 c2 )
                 ( dot-perm c3 c2 c1 )
          )
         )
        )
       )
      )
     )
```