

# Second Racket Programming Assignment

## Learning Abstract

There are many different things I learned with this assignment. First and foremost I was able to learn how to create pictures and shapes. I was able to learn how to create circles, squares, and create different random colors. I was also able to create number sequences which was pretty neat. Overall, I was able to further my knowledge using dr racket and because of that I feel stronger in the language.

## Task 1 - Permutations of Randomly Colored Stacked Dots

```
#lang racket
```

Welcome to [DrRacket](#), version 8.2 [cs].

Language: racket, with debugging; memory limit: 128 MB.

```
> (require 2htdp/image)
```

```
> (define s 100)
```

```
> (define first_radius 45)
```

```
> (define second_radius 30)
```

```
> (define third_radius 15)
```

```
> (define (rectangle a b c d)
```

```
  (define ab(square s "solid" a))
```

```
  (define bc (circle first_radius "solid" b))
```

```
  (define cd (circle second_radius "solid" c))
```

```
  (define de (circle third_radius "solid" d))
```

```
  (overlay de cd bc ab))
```

```
> (define (dots a b c)
```

```
  (define first_ab (circle first_radius "solid" a))
```

```
  (define first_bc (circle second_radius "solid" b))
```

```
  (define first_cd (circle third_radius "solid" c))
```

```
  (define second_ab (circle first_radius "solid" b))
```

```
  (define second_bc (circle second_radius "solid" c))
```

```
  (define second_cd (circle third_radius "solid" a))
```

```
  (define third_ab (circle first_radius "solid" c))
```

```
  (define third_bc (circle second_radius "solid" a))
```

```
  (define third_cd (circle third_radius "solid" b))
```

```
  (define fourth_ab (circle first_radius "solid" a))
```

```
  (define fourth_bc (circle second_radius "solid" c))
```

```
  (define fourth_cd (circle third_radius1 "solid" b))
```

```
  (define fifth_ab (circle first_radius "solid" b))
```

```
  (define fifth_bc (circle second_radius "solid" a))
```

```
  (define fifth_cd (circle third_radius "solid" c))
```

```
  (define sixth_ab (circle first_radius "solid" c))
```

```
  (define sixth_bc (circle second_radius "solid" b))
```

```
  (define sixth_cd (circle third_radius "solid" a))
```

```
  (beside
```

```
    (overlay first_cd first_bc first_ab)
```

```
    (overlay second_cd second_bc second_ab)
```

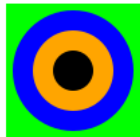
```
    (overlay third_cd third_bc third_ab)
```

```
    (overlay fourth_cd fourth_bc fourth_ab)
```

```
    (overlay fifth_cd fifth_bc fifth_ab)
```

```
    (overlay sixth_cd sixth_bc sixth_ab)))
```

```
> (rectangle "green" "blue" "orange" "black")
```



```
> (dots "red" "black" "white")
```

```
⚠️⚠️ third_radius1: undefined;  
cannot reference an identifier before its definition
```

```
> (define fourth_cd (circle third_radius "solid" b))
```

```
⚠️⚠️ b: undefined;  
cannot reference an identifier before its definition
```

```

> (dots "red" "black" "white")
❗❗ third_radius1: undefined;
cannot reference an identifier before its definition
> (define fourth_cd (circle third_radius "solid" b))
❗❗ b: undefined;
cannot reference an identifier before its definition
> (define (dots a b c)
(define first_ab (circle first_radius "solid" a))
(define first_bc (circle second_radius "solid" b))
(define first_cd (circle third_radius "solid" c))
(define second_ab (circle first_radius "solid" b))
(define second_bc (circle second_radius "solid" c))
(define second_cd (circle third_radius "solid" a))
(define third_ab (circle first_radius "solid" c))
(define third_bc (circle second_radius "solid" a))
(define third_cd (circle third_radius "solid" b))
(define fourth_ab (circle first_radius "solid" a))
(define fourth_bc (circle second_radius "solid" c))
(define fourth_cd (circle third_radius "solid" b))
(define fifth_ab (circle first_radius "solid" b))
(define fifth_bc (circle second_radius "solid" a))
(define fifth_cd (circle third_radius "solid" c))
(define sixth_ab (circle first_radius "solid" c))
(define sixth_bc (circle second_radius "solid" b))
(define sixth_cd (circle third_radius "solid" a))
(overlay first_cd first_bc first_ab)
(overlay second_cd second_bc second_ab)
(overlay third_cd third_bc third_ab)
(overlay fourth_cd fourth_bc fourth_ab)
(overlay fifth_cd fifth_bc fifth_ab)
(overlay sixth_cd sixth_bc sixth_ab))
> (dots "red" "black" "white")

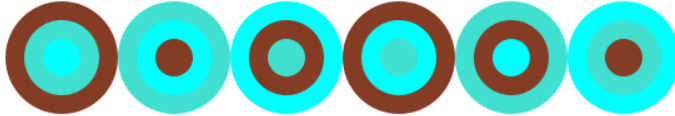
```



```

> (dots "brown" "turquoise" "cyan")

```



```

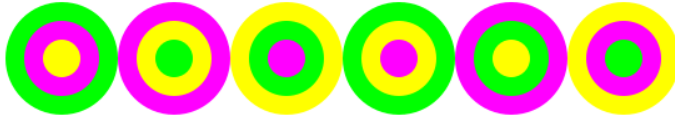
> (dots "green" "fuchsia" "yellow")

```

```

❗❗ circle: expects an image-color-or-pen as third argument, given "fuchsia"
> (dots "green" "fuchsia" "yellow")

```



## Task 2 – Number Sequences

---

#lang racket

Welcome to [DrRacket](#), version 8.2 [cs].

Language: racket, with debugging; memory limit: 128 MB.





```
> (define (natural-number n)
  (cond
    ((= n 1)
     1)
    ((> n 1)
     (+ (- n 1) 1))
    )))
> (define (natural-sequence n)
  (cond
    ((> n 0)
     (natural-sequence (- n 1))
     (display (natural-number n)) (display " "))))

> (define (copies a n)
  (cond
    ((= n 1)
     (display a)(display " "))
    (( > n 1)
     (display a )(display " ") (copies a (- n 1)))))


> (define (special-natural-sequence n)
  (cond
    ((= n 1)
     (display 1)(display " "))
    ((> n 1)
     (special-natural-sequence (- n 1))
     (copies n n))))

> (natural-sequence 7)

1 2 3 4 5 6 7
> (natural-sequence 14)

1 2 3 4 5 6 7 8 9 10 11 12 13 14
> (copies 19)
  copies: arity mismatch;
the expected number of arguments does not match the given number
expected: 2
given: 1
> (copies 77)
  copies: arity mismatch;
the expected number of arguments does not match the given number
expected: 2
given: 1
> (copies 7 6)
7 7 7 7 7 7
> (copies 4 9)
4 4 4 4 4 4 4 4 4
```

```

> (special-natural-sequence 7)

special-natural-sequence: undefined;
cannot reference an identifier before its definition
> (special-natural-sequence 7)
1 2 2 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6 6 7 7 7 7 7 7 7 7
> (special-natural-sequence 21)
1 2 2 3 3 3 4 4 4 4 5 5 5 5 6 6 6 6 6 7 7 7 7 7 7 7 8 8 8 8 8 8 9 9 9 9 9 9 9 10 10 10 10 10 10 10 10 10 11 11 11 11 11 11 11 11 12 12 12 12 12 12 12
14 14 14 14 14 14 14 14 14 14 15 15 15 15 15 15 15 15 15 15 15 15 16 16 16 16 16 16 16 16 16 16 16 16 16 16 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
>|

```

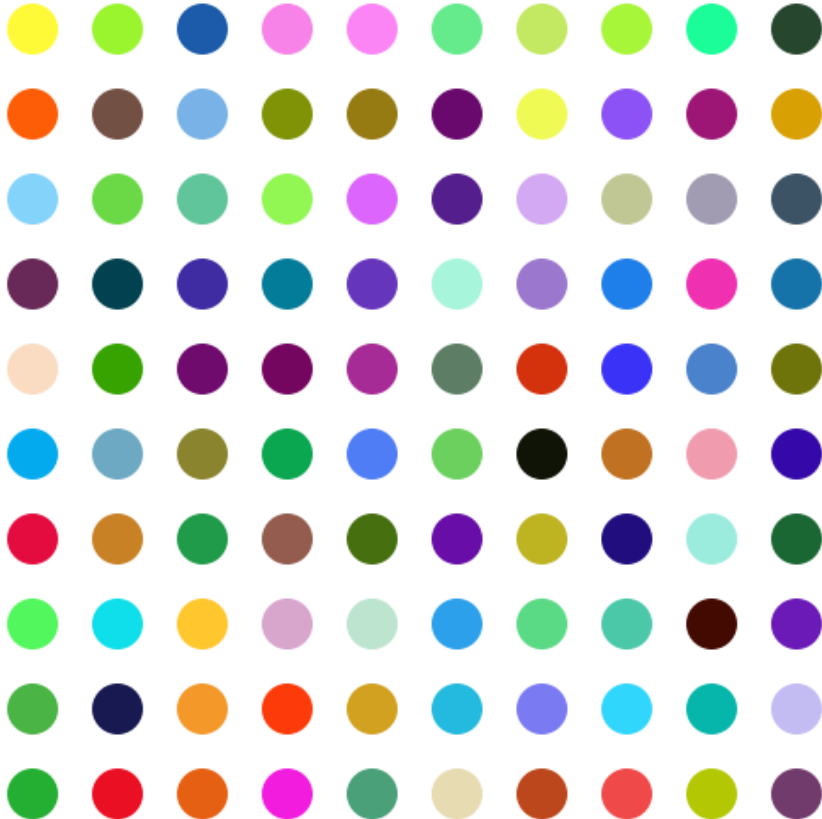
## Task 3 – Hirst-Dots

```
#lang racket
```

```
Welcome to DrRacket, version 8.2 [cs].
```

```
Language: racket, with debugging; memory limit: 128 MB.
```

```
> (require 2htdp/image)
> (define (colors) (random 256))
> (define (rand) (color(colors) (colors) (colors)))
> (define background (square 50 "solid" "white"))
> (define (picture) (define dot (circle 15 "solid" (rand))) (overlay dot background))
> (define (dot-row r) (cond ((= r 0) empty-image) ( ( > r 0) (beside (dot-row(- r 1)) (picture) ) ) ) )
> (define (graph r e) (cond ((= r 0) empty-image) ( ( > r 0) (above (graph (- r 1) e) (dot-row e) ) ) ) )
> (define (dotted-program d) (graph d d))
> (dotted-program 10)
```



```
> (dotted-program 4)
```



## Task 4 – Stella Thing

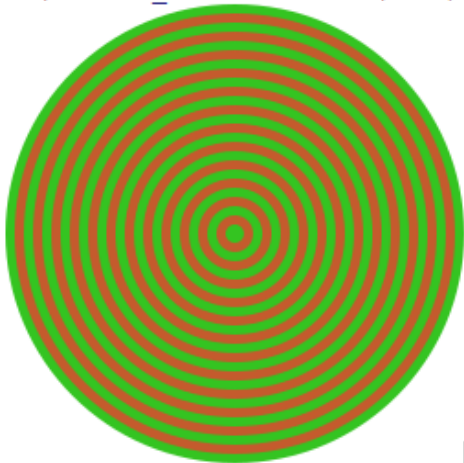
```
#lang racket
```

Welcome to [DrRacket](#), version 8.2 [cs].

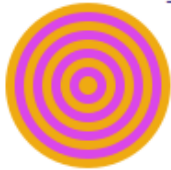
Language: racket, with debugging; memory limit: 128 MB.

```
> > ( require 2htdp/image )
> ( define ( colored_circle s count c1 c2 )
  ( define triangle ( / s count ) )
  ( paint-nested-circles-two 1 count triangle c1 c2 )
  )
> ( define ( paint-nested-circles-two from to triangle c1 c2 )
  ( define s-length ( * from triangle ) )
  ( cond
    ( ( = from to )
      ( if ( even? from )
        ( circle s-length "solid" c1 )
        ( circle s-length "solid" c2 )
        )
      )
    ( ( < from to )
      ( if ( even? from )
        ( overlay
          ( circle s-length "solid" c1 )
          ( paint-nested-circles-two ( + from 1 ) to triangle c1 c2 )
          )
        ( overlay
          ( circle s-length "solid" c2 )
          ( paint-nested-circles-two ( + from 1 ) to triangle c1 c2 )
          )
        )
      )
    )
  )
)
)
)
)
)
)
> ( define ( color_number ) ( random 256 ) )
> ( define ( rand ) ( color ( color_number ) ( color_number ) ( color_number ) ) )
_
```

```
> (colored_circle 125 25 (rand) (rand))
```



```
> ( colored_circle 45 9 ( rand ) ( rand ) )
```



```
>
```

## Task 5 – Creation

```
#lang racket
```

```
Welcome to DrRacket, version 8.2 [cs].
```

```
Language: racket, with debugging; memory limit: 128 MB.
```

```
> > (require 2htdp/image)
```

```
( define ( row-of-squares n )
```

```
( cond
```

```
(( = n 0)
```

```
empty-image)
```

```
(( > n 0)
```

```
( beside ( row-of-squares ( - n 1 ) ) ( random-color-tile ) ) ) ) )
```

```
( define ( dominoes r c )
```

```
( cond
```

```
(( = r 0)
```

```
empty-image)
```

```
(( > r 0)
```

```
( above ( dominoes ( - r 1 ) c ) ( row-of-squares c ) ) ) ) )
```

```
> ( define ( hirst-dots n )
```

```
( dominoes n n )
```

```
> ( define ( random-color-tile )
```

```
(define a( circle 15 "solid" ( random-color ) ) )
```

```
(define b(square 60 "solid" "black"))
```

```
(overlay a b)
```

```
> ( define ( random-color ) ( color ( rgb-value ) ( rgb-value ) ( rgb-value ) ) )
```

```
> ( define ( random-blue-color ) ( color 0 0 255 ( rgb-value ) ) )
```

```
> ( define ( rgb-value ) ( random 256 ) )
```

```
#<random>
```



```
> (dominoes 1 1)
```



```
> (dominoes 1 2)
```



```
> (dominoes 3 2)
```



```
> (dominoes 2 2)
```



```
>
```