Stats 102A - Homework 2 - Output File

Eli Radparvar

To receive full credit the functions you write must pass all tests. We may conduct further tests that are not included on this page as well.

Academic Integrity Statement

By including this statement, I, Eli Radparvar, declare that all of the work in this assignment is my own original work. At no time did I look at the code of other students nor did I search for code solutions online. I understand that plagiarism on any single part of this assignment will result in a 0 for the entire assignment and that I will be referred to the dean of students.

```
setwd("C:/Users/eliis/Desktop/stats 101a/stats102a")
source("102a_hw_02_script_Eli_Radparvar.R")
```

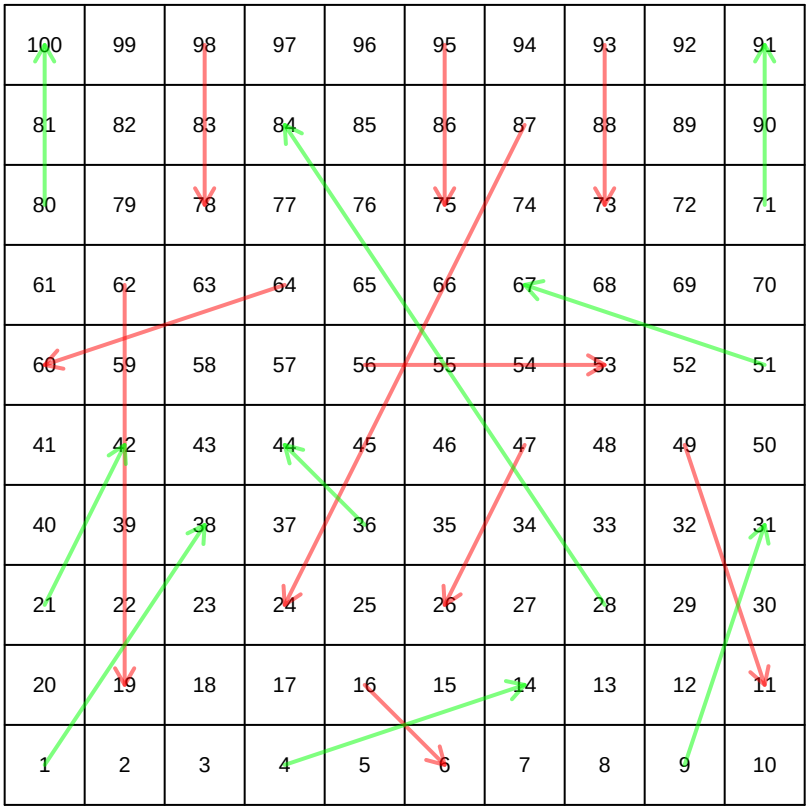
Part 1: Board representation

Create a single list object called `board` where you store the features of the game board in R.

```
board <- list(
  n_rows = 10,
  n_cols = 10,
  chutes = list(
    "16" = 6,
    "47" = 26,
    "49" = 11,
    "56" = 53,
    "62" = 19,
    "64" = 60,
    "87" = 24,
    "93" = 73,
    "95" = 75,
    "98" = 78
  ),
  ladders = list(
    "1" = 38,
    "4" = 14,
    "9" = 31,
    "21" = 42,
    "28" = 84,
    "36" = 44,
    "51" = 67,
    "71" = 91,
    "80" = 100
  )
)
```

Part 2: Plot of Game board

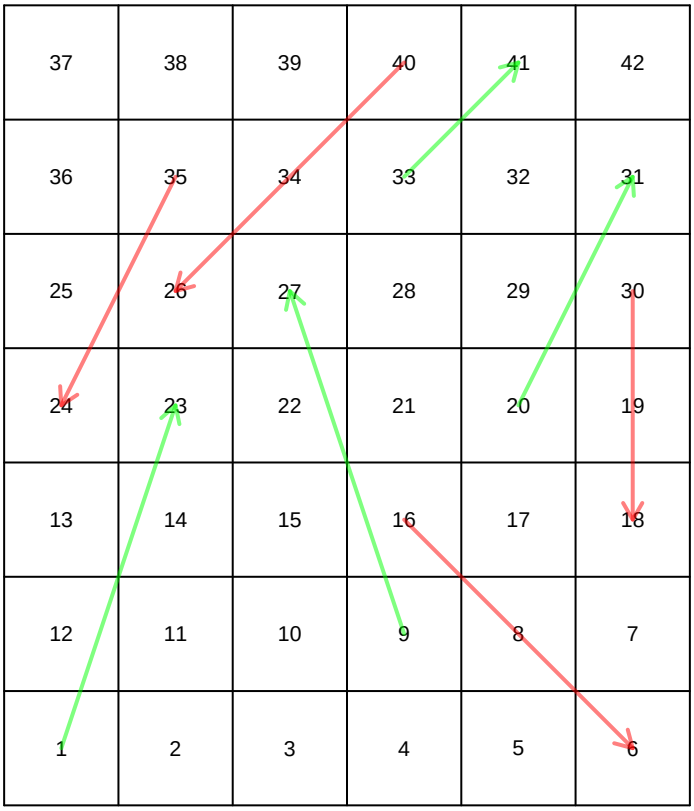
```
# par() should help the plot be more visible. you can adjust this as necessary
par(mar = c(0, 0, 0, 0))
show_board(board)
```



Part 3: Miniboards

Create the `miniboard` objects and plots.

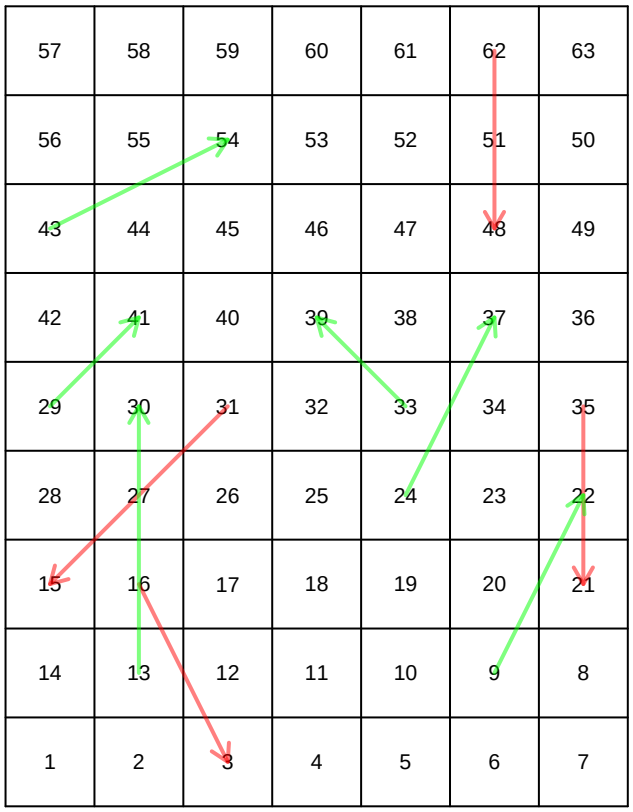
```
miniboard1 <- list(
  n_rows = 7,
  n_cols = 6,
  chutes  = list(
    "16"=6,
    "30"=18,
    "35"=24,
    "40"=26
  ),
  ladders = list(
    "1"=23,
    "9"=27,
    "20"=31,
    "33"=41
  )
)
par(mar = c(0, 0, 0, 0))
show_board(miniboard1)
```




```
miniboard2 <- list(
  n_rows = 9,
  n_cols = 7,
  chutes  = list(
```

```
    "16"=3,
    "31"=15,
    "35"=21,
    "62"=48
  ),
  ladders = list(
    "9"=22,
    "13"=30,
    "24"=37,
    "29"=41,
    "33"=39,
    "43"=54
  )
)

par(mar = c(0, 0, 0, 0))
show_board(miniboard2)
```




```
miniboard3 <- list(
  n_rows = 9,
  n_cols = 8
)
par(mar = c(0, 0, 0, 0))
show_board(miniboard3)
```

65	66	67	68	69	70	71	72
64	63	62	61	60	59	58	57
49	50	51	52	53	54	55	56
48	47	46	45	44	43	42	41
33	34	35	36	37	38	39	40
32	31	30	29	28	27	26	25
17	18	19	20	21	22	23	24
16	15	14	13	12	11	10	9
1	2	3	4	5	6	7	8

Part 4: Verbose output of one single player game

```
# Spinner function
spin <- function() {
  sample(6, 1)
}
```

```
set.seed(5)
play_solo(board, verbose = TRUE)
```

```
## Turn 1
## Start at 0
## Spinner: 2
## Turn ends at: 2
##
## Turn 2
## Start at 2
## Spinner: 3
## Turn ends at: 5
##
## Turn 3
## Start at 5
## Spinner: 1
## Turn ends at: 6
##
## Turn 4
## Start at 6
## Spinner: 3
## Landed on: 9
## Ladder!
## Turn ends at: 31
##
## Turn 5
## Start at 31
## Spinner: 1
## Turn ends at: 32
##
## Turn 6
## Start at 32
## Spinner: 1
## Turn ends at: 33
##
## Turn 7
## Start at 33
## Spinner: 5
## Turn ends at: 38
##
## Turn 8
## Start at 38
## Spinner: 6
## Turn ends at: 44
##
```

```
## Turn 9
## Start at 44
## Spinner: 3
## Landed on: 47
## Chute!
## Turn ends at: 26
##
## Turn 10
## Start at 26
## Spinner: 3
## Turn ends at: 29
##
## Turn 11
## Start at 29
## Spinner: 6
## Turn ends at: 35
##
## Turn 12
## Start at 35
## Spinner: 2
## Turn ends at: 37
##
## Turn 13
## Start at 37
## Spinner: 5
## Turn ends at: 42
##
## Turn 14
## Start at 42
## Spinner: 4
## Turn ends at: 46
##
## Turn 15
## Start at 46
## Spinner: 2
## Turn ends at: 48
##
## Turn 16
## Start at 48
## Spinner: 5
## Turn ends at: 53
##
## Turn 17
## Start at 53
## Spinner: 3
## Landed on: 56
## Chute!
## Turn ends at: 53
##
## Turn 18
## Start at 53
## Spinner: 1
## Turn ends at: 54
##
```

```
## Turn 19
## Start at 54
## Spinner: 6
## Turn ends at: 60
##
## Turn 20
## Start at 60
## Spinner: 4
## Landed on: 64
## Chute!
## Turn ends at: 60
##
## Turn 21
## Start at 60
## Spinner: 3
## Turn ends at: 63
##
## Turn 22
## Start at 63
## Spinner: 2
## Turn ends at: 65
##
## Turn 23
## Start at 65
## Spinner: 5
## Turn ends at: 70
##
## Turn 24
## Start at 70
## Spinner: 2
## Turn ends at: 72
##
## Turn 25
## Start at 72
## Spinner: 2
## Turn ends at: 74
##
## Turn 26
## Start at 74
## Spinner: 3
## Turn ends at: 77
##
## Turn 27
## Start at 77
## Spinner: 1
## Turn ends at: 78
##
## Turn 28
## Start at 78
## Spinner: 2
## Landed on: 80
## Ladder!
## Turn ends at: 100
```

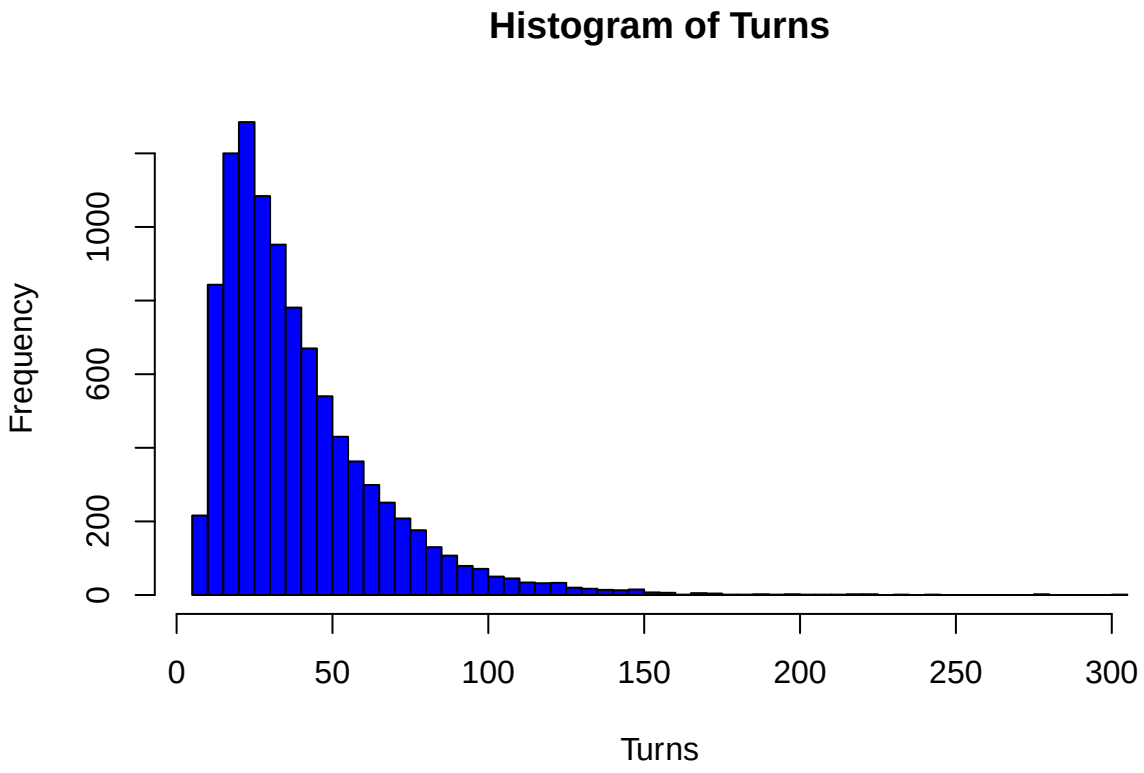
```
## $turns
## [1] 28
##
## $chute_tally
##  [1] 0 1 0 1 0 1 0 0 0 0
##
## $ladder_tally
## [1] 0 0 1 0 0 0 0 0 1
##
## $move_log
##  [1]   2   5   6  31  32  33  38  44  26  29  35  37  42  46  48  53  53  54  60
## [20]  60  63  65  70  72  74  77  78 100
```

Part 5: Monte Carlo Simulation Study

```
# run 10,000 games
set.seed(5)
simulations <- replicate(10000, play_solo(board),  simplify=FALSE)
turns <- c()
  for (i in seq_along(simulations)){
    game <- simulations[[i]]
    turns[i] <- game$turns
  }
```

- Create a histogram (breaks = 50) of the turns.

```
hist(turns, breaks = 50, main = "Histogram of Turns", xlab = "Turns",
     col = "blue")
```



- Find the minimum number of turns. How many times out of 10,000 did a game finish with the minimum number of turns?

```
min(turns)
```

```
## [1] 7
```

```
sum(turns == min(turns))
```

```
## [1] 10
```

- Find the maximum number of turns.

```
max(turns)
```

```
## [1] 301
```

- What is the median number of turns?

```
median(turns)
```

```
## [1] 32
```

- What is the mean number of turns?

```
mean(turns)
```

```
## [1] 39.3468
```

- What proportion of games take 100 or more turns to complete?

```
mean(turns >= 100)
```

```
## [1] 0.0329
```

- What proportion of games take 10 or fewer turns to complete?

```
mean(turns <= 10)
```

```
## [1] 0.0216
```

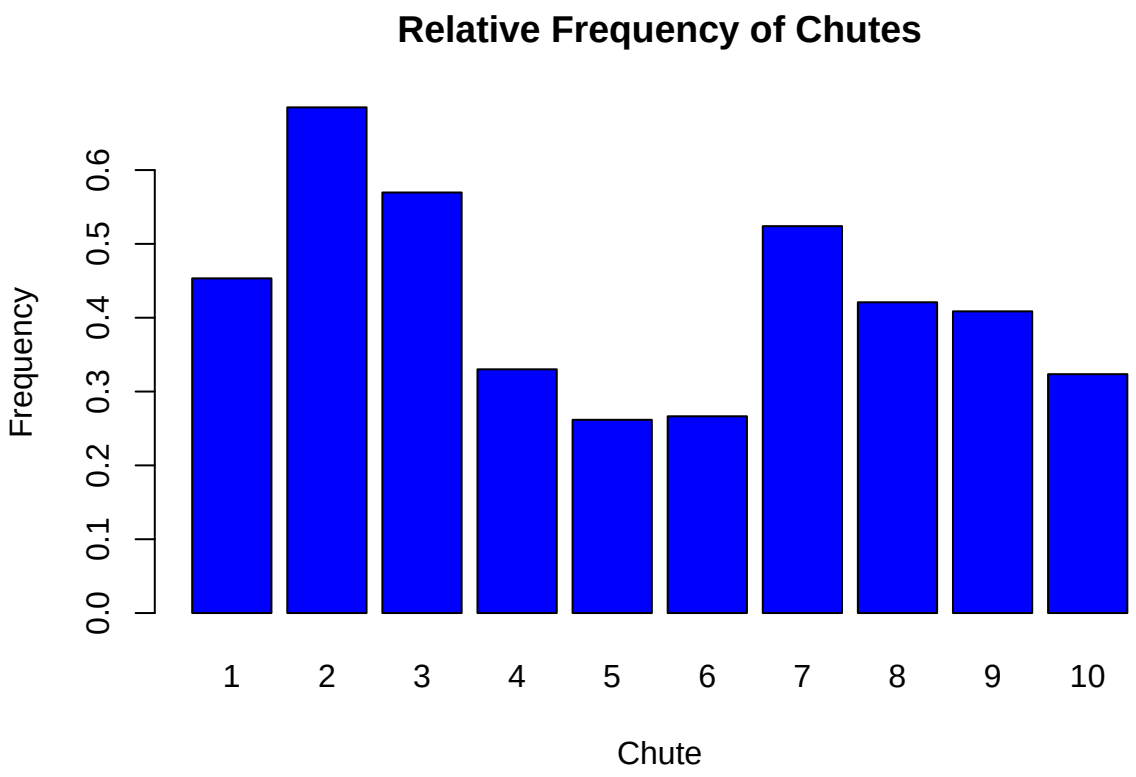
- What proportion of games utilize ladder 9 (the shortcut to win on space 80)?

```
ladder9_used <- logical(0)
  for (i in seq_along(simulations)){
    game <- simulations[[i]]
    ladders <- game$ladder_tally
    ladder9_used[i] <- (ladders[9] >= 1)
  }
mean(ladder9_used)
```

```
## [1] 0.4815
```

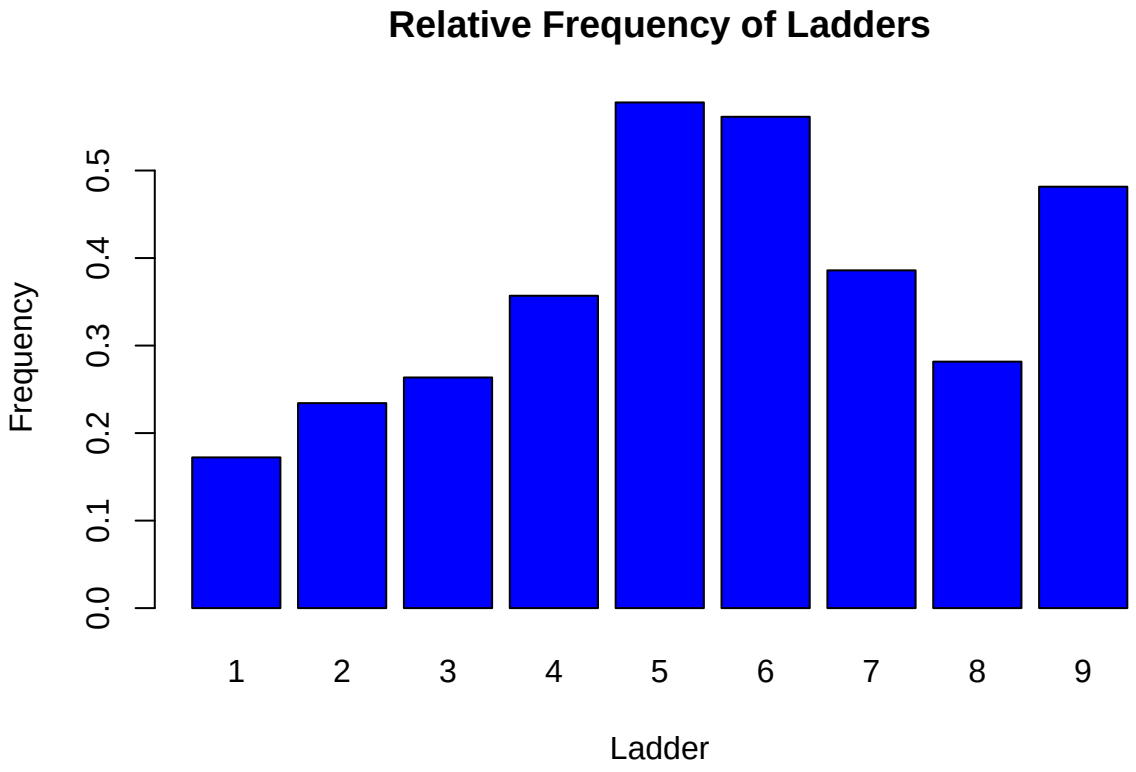
- Create a barplot of the relative frequency of how often each chute is utilized. (Number the chutes in order based on their starting square. The chute with lowest starting number, 16 to 6, is chute 1. The chute going from 98 to 78 is chute 10.)

```r
chutes_sum <- rep(0, 10)
  for (i in seq_along(simulations)){
    game <- simulations[[i]]
    chutes <- game$chute_tally
    chutes_sum <- chutes_sum + chutes
  }
barplot(chutes_sum/10000, main = "Relative Frequency of Chutes",
        xlab = "Chute", ylab = "Frequency", col = "blue", names.arg = 1:10)
```



- Create a barplot of the relative frequency of how often each ladder is utilized. (Number the ladders in order based on their starting square. The ladder with lowest starting number, 1 to 38, is ladder 1. The Ladder going from 80 to 100 is ladder 9.)

```r
ladders_sum <- rep(0, 9)
  for (i in seq_along(simulations)){
    game <- simulations[[i]]
    ladders <- game$ladder_tally
    ladders_sum <- ladders_sum + ladders
  }
barplot(ladders_sum/10000, main = "Relative Frequency of Ladders",
        xlab = "Ladder", ylab = "Frequency", col = "blue", names.arg = 1:9)
```



Extra Credit Challenge (Up to 10 bonus points)

- On the datagenetics web page, under the section "Transition Matrix" there are plots showing a 'heat map' of which squares are landed on after one turn, after two turns, after three rolls, etc. Recreate these plots in R for where the player may land after 1, 2, or 3 rolls. While ideally in base R graphics, for the extra credit only you may use ggplot to assist with transparency. You may reuse code written earlier in this homework if applicable. Type your extra credit code directly in your .Rmd file. This is not an easy task, and the effort/point-benefit ratio is probably not worth it if you are doing it for points alone. It's mostly presented as a challenge for students who want to go above and beyond what is expected of them.

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
# Transition Matrix function

create_transition_matrix <- function(board) {
  # Calculates indices of valid positions
  states <- 0:100
  exclude <- (c(as.numeric(names(board$chutes)), as.numeric(names(board$ladders))))
  valid_positions <- states[!(states %in% exclude)]
  position_index <- setNames(seq_along(valid_positions), valid_positions)

  # Helper function for chutes and ladders
  get_next_position <- function(position) {
    if (as.character(position) %in% names(board$chutes)) {
```

```
      board$chutes[[as.character(position)]]
    } else if (as.character(position) %in% names(board$ladders)) {
      board$ladders[[as.character(position)]]
    } else {
    position
    }
  }

  # Creates transition matrix
  transition_matrix <- matrix(0, nrow = 82, ncol = 82)

  for (start_pos in valid_positions) {
    start_idx <- position_index[as.character(start_pos)]

    for (roll in 1:6) {
      # Calculates landing position
      land_pos <- start_pos + roll

      # Applies helper to deal with chutes and ladders
      final_pos <- get_next_position(land_pos)

      # Updates transition matrix
      end_idx <- position_index[as.character(final_pos)]
      transition_matrix[start_idx, end_idx] <- transition_matrix[start_idx, end_idx] + 1/6
    }
  }

  # Where you start
  rownames(transition_matrix) <- valid_positions
  #Where you end up
  colnames(transition_matrix) <- valid_positions

  transition_matrix
}

transition_matrix <- create_transition_matrix(board)
```

```
#After 1 roll
roll0 <- c(1, rep(0, 81))
result_t1 <- roll0 %*% transition_matrix
print(result_t1)
```

```
##      0         2         3         5         6 7 8 10 11 12 13        14 15 17
## [1,] 0 0.1666667 0.1666667 0.1666667 0.1666667 0 0  0  0  0  0 0.1666667  0  0
##      18 19 20 22 23 24 25 26 27 29 30 31 32 33 34 35 37        38 39 40 41 42
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 0.1666667  0  0  0  0
##      43 44 45 46 48 50 52 53 54 55 57 58 59 60 61 63 65 66 67 68 69 70 72 73 74
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##      75 76 77 78 79 81 82 83 84 85 86 88 89 90 91 92 94 96 97 99 100
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   0
```

```r
#After 2 rolls
result_t2 <- result_t1 %*% transition_matrix
print(result_t2)
```

```
##      0 2          3          5         6         7         8         10
## [1,] 0 0 0.02777778 0.05555556 0.1111111 0.1111111 0.1111111 0.05555556
##              11         12 13         14         15         17         18
## [1,] 0.05555556 0.02777778  0 0.05555556 0.02777778 0.02777778 0.02777778
##              19         20 22 23 24 25 26 27 29 30         31 32 33 34 35 37 38
## [1,] 0.02777778 0.02777778  0  0  0  0  0  0  0  0 0.08333333  0  0  0  0  0  0
##              39         40         41         42         43         44 45 46 48
## [1,] 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778 0.02777778  0  0  0
##      50 52 53 54 55 57 58 59 60 61 63 65 66 67 68 69 70 72 73 74 75 76 77 78 79
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##      81 82 83 84 85 86 88 89 90 91 92 94 96 97 99 100
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0   0
```

```r
#After 3 rolls
result_t3 <- result_t2 %*% transition_matrix
print(result_t3)
```

```
##      0 2 3          5          6          7          8         10         11
## [1,] 0 0 0 0.00462963 0.05092593 0.03240741 0.05092593 0.06481481 0.08333333
##              12         13        14         15         17         18
## [1,] 0.07407407 0.06018519 0.0462963 0.03240741 0.02777778 0.02314815
##              19         20         22         23         24         25
## [1,] 0.02314815 0.02777778 0.01851852 0.01851852 0.01388889 0.009259259
##              26 27 29 30         31         32         33         34         35
## [1,] 0.02314815  0  0  0 0.06944444 0.01388889 0.01388889 0.01388889 0.01388889
##              37 38 39         40          41         42         43         44
## [1,] 0.01388889  0  0 0.00462963 0.009259259 0.03703704 0.01851852 0.03703704
##              45         46         48         50 52 53 54 55 57 58 59 60 61 63
## [1,] 0.02777778 0.02314815 0.01388889 0.00462963  0  0  0  0  0  0  0  0  0  0
##      65 66 67 68 69 70 72 73 74 75 76 77 78 79 81 82 83 84 85 86 88 89 90 91 92
## [1,]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##      94 96 97 99 100
## [1,]  0  0  0  0   0
```

Ran out of time for heatmap but hope I can still get points because I believe the transition matrix was the hardest step and I just didn't have time for heatmap. I really put a lot of effort and time.