Stats 102A - Homework 2 - Output File

Daren Sathasivam – 306229580

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, **Daren Sathasivam**, 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.

```
source("102a_hw_02_script_Daren_Sathasivam.R")  # edit with your file name
```

```
## Error in eval(ei, envir): object 'board' not found
```

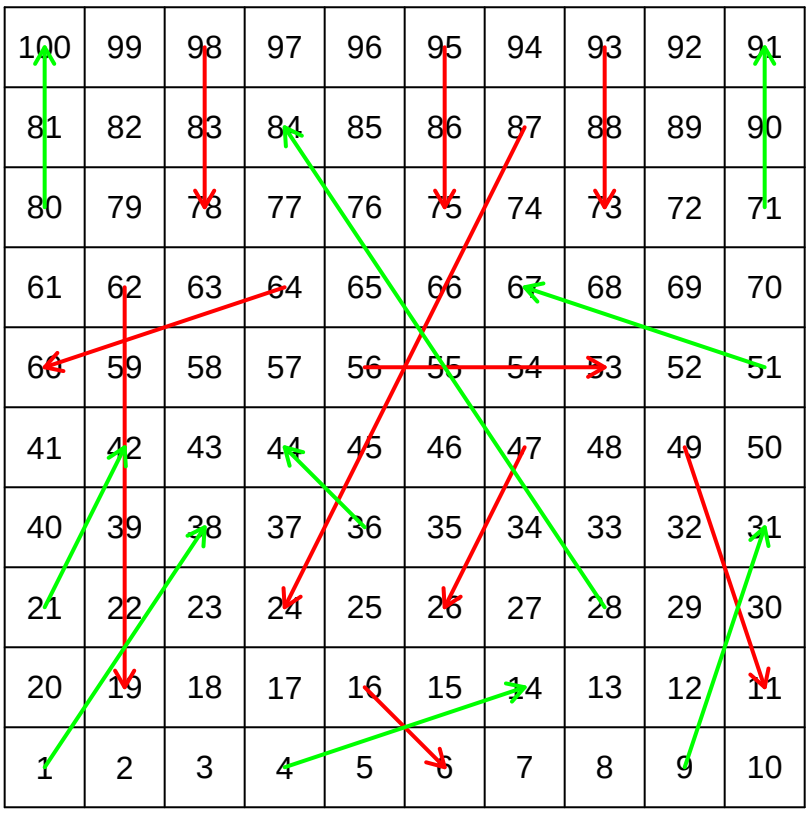
Part 1: Board representation

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

```
# chutes, ladders, and board_dim are list objects
board <- list(chutes = chutes <- list(
  start = c(16, 47, 49, 56, 62, 64, 87, 93, 95, 98),
  end = c(6, 26, 11, 53, 19, 60, 24, 73, 75, 78)
), ladders = ladders <- list(
  start = c(1, 4, 9, 21, 28, 36, 51, 71, 80),
  end = c(38, 14, 31, 42, 84, 44, 67, 91, 100)
), dim = board_dim <- list(row = 10, col = 10))
```

Part 2: Plot of Game board

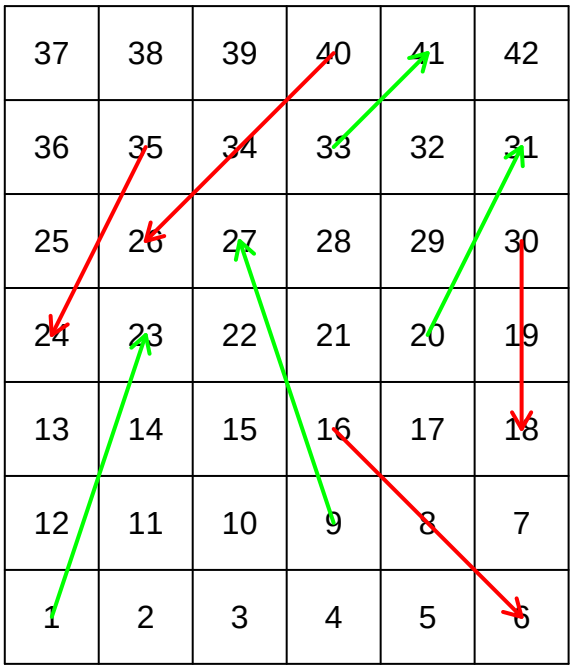
```
# par() should help the plot be more visible. you can adjust this as necessary
board_dim <- list(row = 10, col = 10)
chutes <- list(
  start = c(16, 47, 49, 56, 62, 64, 87, 93, 95, 98),
  end = c(6, 26, 11, 53, 19, 60, 24, 73, 75, 78)
)
ladders <- list(
  start = c(1, 4, 9, 21, 28, 36, 51, 71, 80),
  end = c(38, 14, 31, 42, 84, 44, 67, 91, 100)
)
board <- list(chutes = chutes, ladders = ladders, dim = board_dim)
par(mar = c(0, 0, 0, 0))
show_board(board)
```



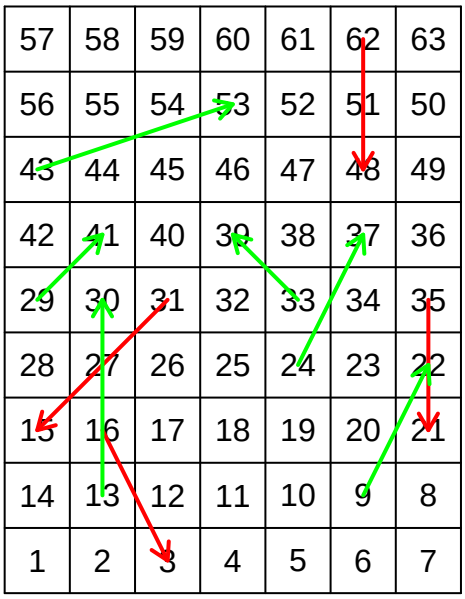
Part 3: Miniboards

Create the `miniboard` objects and plots.

```
board_dim1 <- list(row = 7, col = 6)
ladders1 <- list(start = c(1, 9, 20, 33), end = c(23, 27, 31, 41))
chutes1 <- list(start = c(16, 30, 35, 40), end = c(6, 18, 24, 26))
miniboard1 <- list(chutes = chutes1, ladders = ladders1, dim = board_dim1)
par(mar = c(1, 7, 1, 7))
show_board(miniboard1)
```




```
board_dim2 <- list(row = 9, col = 7)
ladders2 <- list(start = c(9, 13, 24, 29, 33, 43), end = c(22, 30, 37, 41, 39, 53))
chutes2 <- list(start = c(16, 31, 35, 62), end = c(3, 15, 21, 48))
miniboard2 <- list(chutes = chutes2, ladders = ladders2, dim = board_dim2)
par(mar = c(1, 8, 1, 8))
show_board(miniboard2)
```




```
board_dim3 <- list(row = 9, col = 7)
ladders3 <- list()
chutes3 <- list()
miniboard3 <- list(chutes = chutes3, ladders = ladders3, dim = board_dim3)
par(mar = c(1, 8, 2, 8))
show_board(miniboard3)
```

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

Part 4: Verbose output of one single player game

```
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
##
```

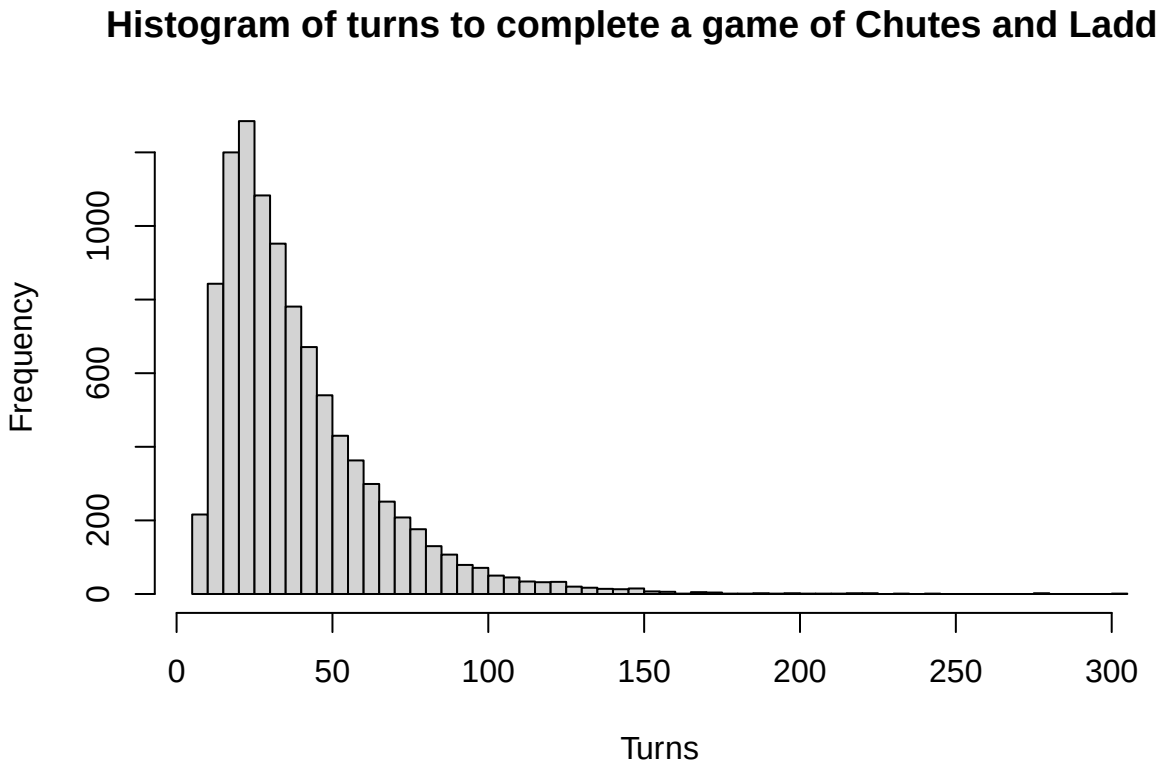
```
## $ladder_tally
## [1] 0 0 1 0 0 0 0 0 1
##
## $move_log
##  [1]   2   5   6  31  32  33  38  44  26  29  35  37  42  46  48  53  53  54  60
## [20]  60  63  65  70  72  74  77  78 100
```

Part 5: Monte Carlo Simulation Study

```
# run 10,000 games
simulation <- replicate(10000, play_solo(board), simplify = FALSE)
```

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

```
total_turns <- sapply(simulation, function(x){
  x$turns
})
hist(total_turns, breaks = 50, main = "Histogram of turns to complete a game of Chutes and Ladders", xla
```



- 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 <- min(total_turns)
# Minimum number of turns in a game
min_turns
```

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

```
# For amount of games that finished with the minimum amount of turns
sum(total_turns == min_turns)
```

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

- Find the maximum number of turns.

```
max_turns <- max(total_turns)
max_turns
```

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

- What is the median number of turns?

```
median_turns <- median(total_turns)
median_turns
```

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

- What is the mean number of turns?

```
mean_turns <- mean(total_turns)
mean_turns
```

```
## [1] 39.3484
```

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

```
prop_morethan_100 <- mean(total_turns >= 100)
prop_morethan_100
```

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

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

```
prop_lessthan_10 <- mean(total_turns <= 10)
prop_lessthan_10
```

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

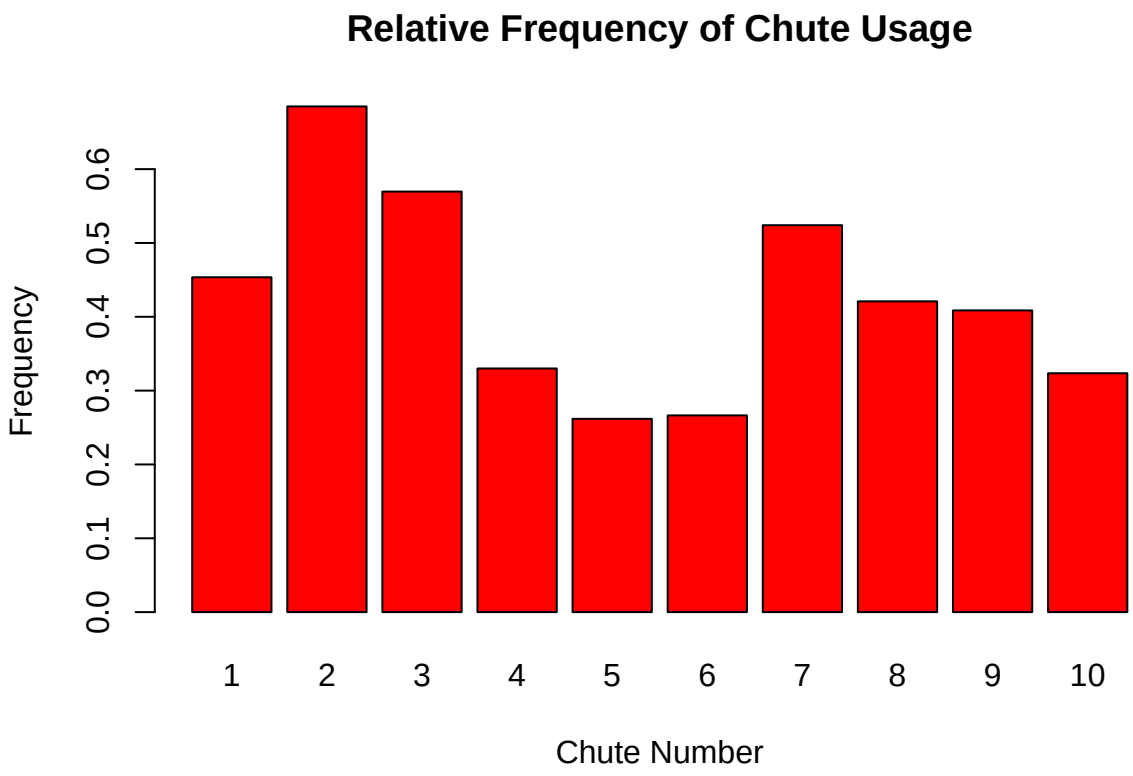
- What proportion of games utilize ladder 9 (the shortcut to win on space 80)?

```
chute_tallies <- tapply(unlist(lapply(simulation, function (x) {
  x$chute_tally
})), rep(seq_along(board$chutes$start), times = 10000), sum)
ladder_tallies <- tapply(unlist(lapply(simulation, function (x) {
  x$ladder_tally
})), rep(seq_along(board$ladders$start), times = 10000), sum)

prop_ladder9 <- ladder_tallies[9] / 10000
prop_ladder9
```

```
##      9
## 0.4814
```

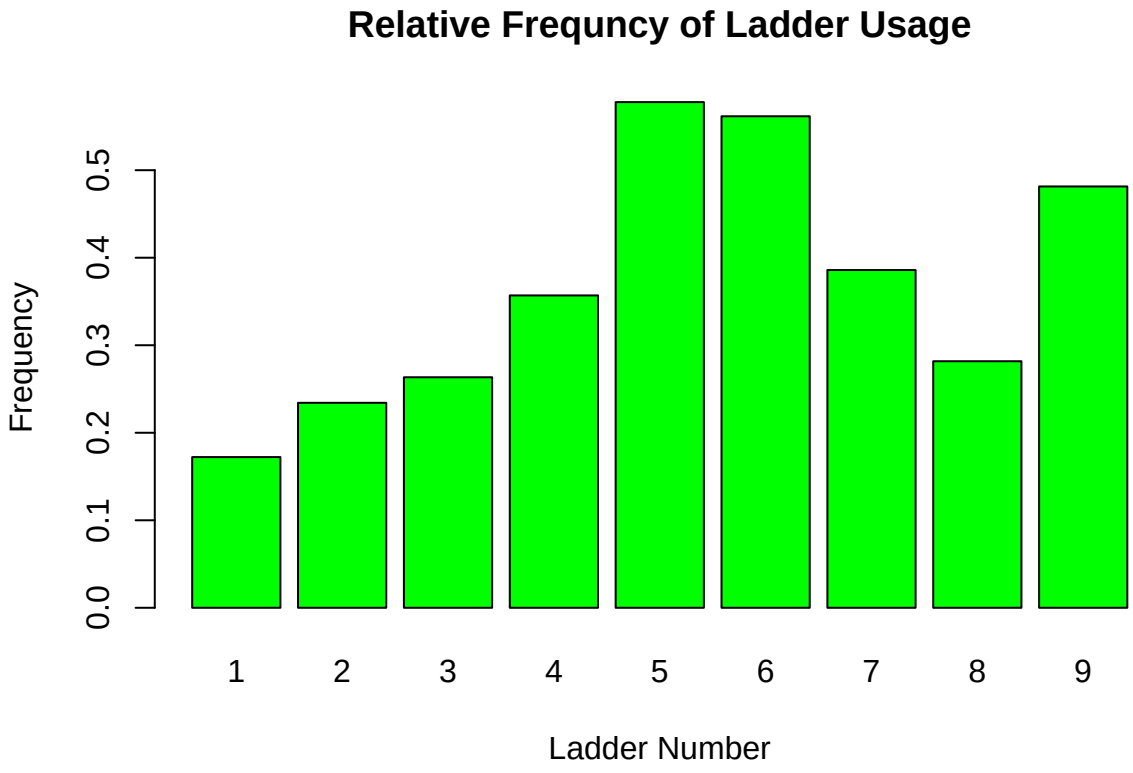
- Barplot of the relative frequency of how often each chute is utilized.

```
barplot(chute_tallies / 10000, main = "Relative Frequency of Chute Usage", xlab = "Chute Number", ylab =
```



- Barplot of the relative frequency of how often each ladder is utilized.

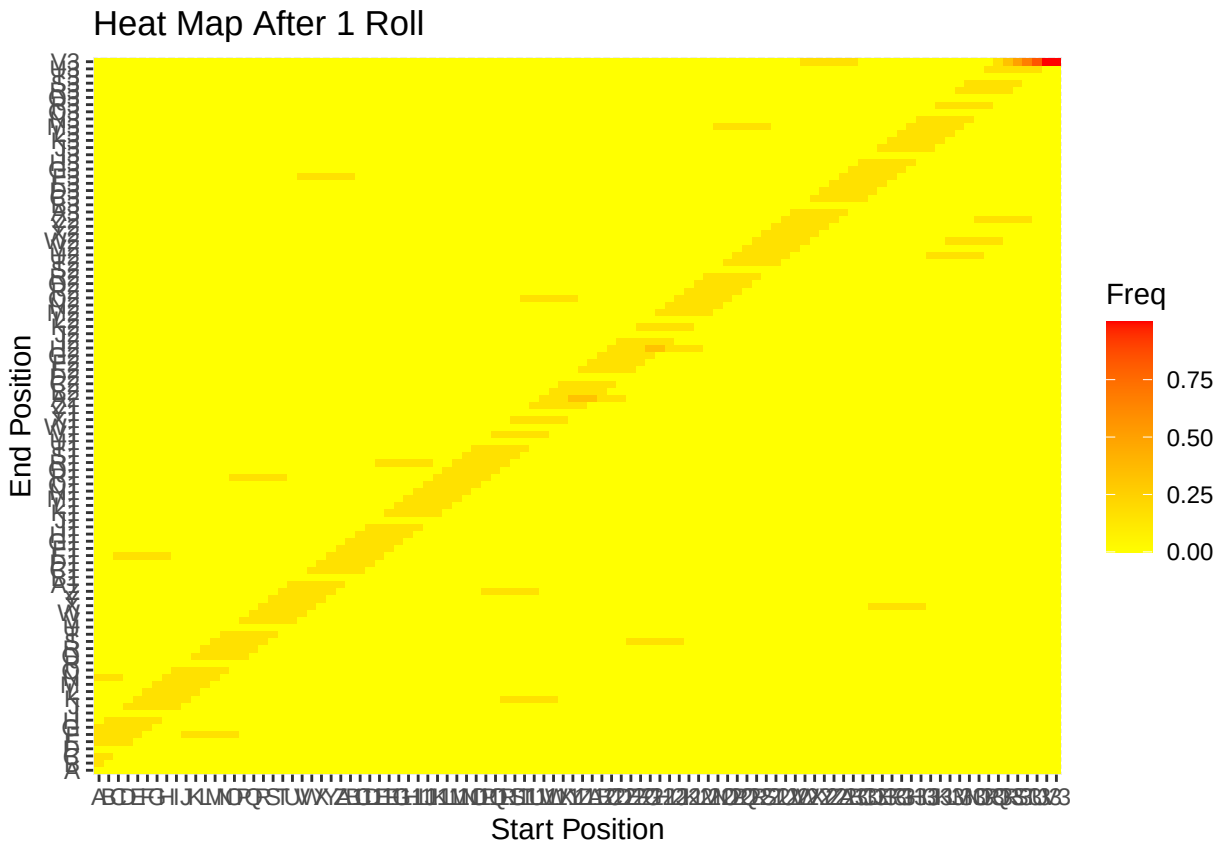
```
barplot(ladder_tallies / 10000, main = "Relative Frequncy of Ladder Usage", xlab = "Ladder Number", ylab
```



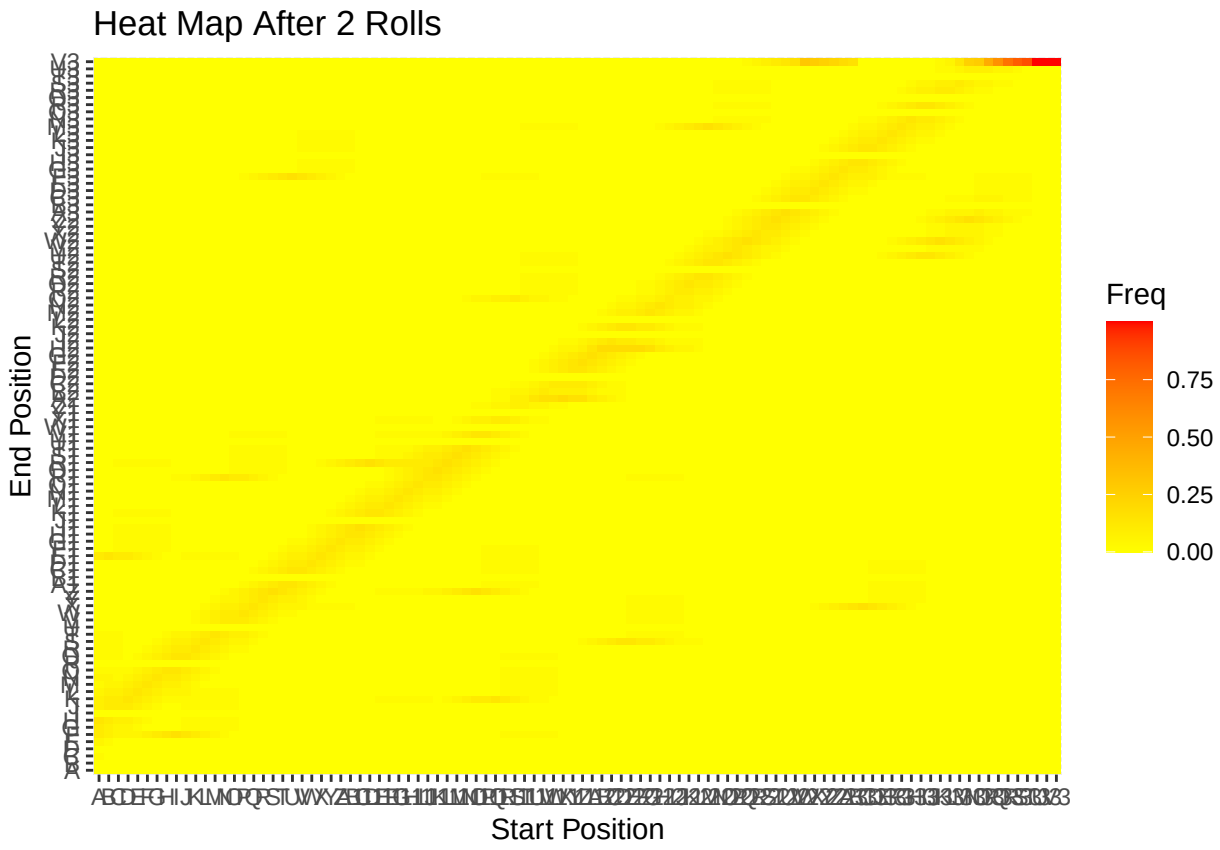
Extra Credit Challenge

Pages to keep note of:
* https://r4ds.had.co.nz/data-visualisation.html
* https://r-charts.com/correlation/heat-map-ggplot2/
* https://datacarpentry.org/R-ecology-lesson/04-visualization-ggplot2.html
* https://stats.stackexchange.com/questions/26722/calculate-transition-matrix-markov-in-r
* https://www.datacamp.com/tutorial/markov-chain-analysis-r
* https://ggplot2.tidyverse.org/reference/aes.html

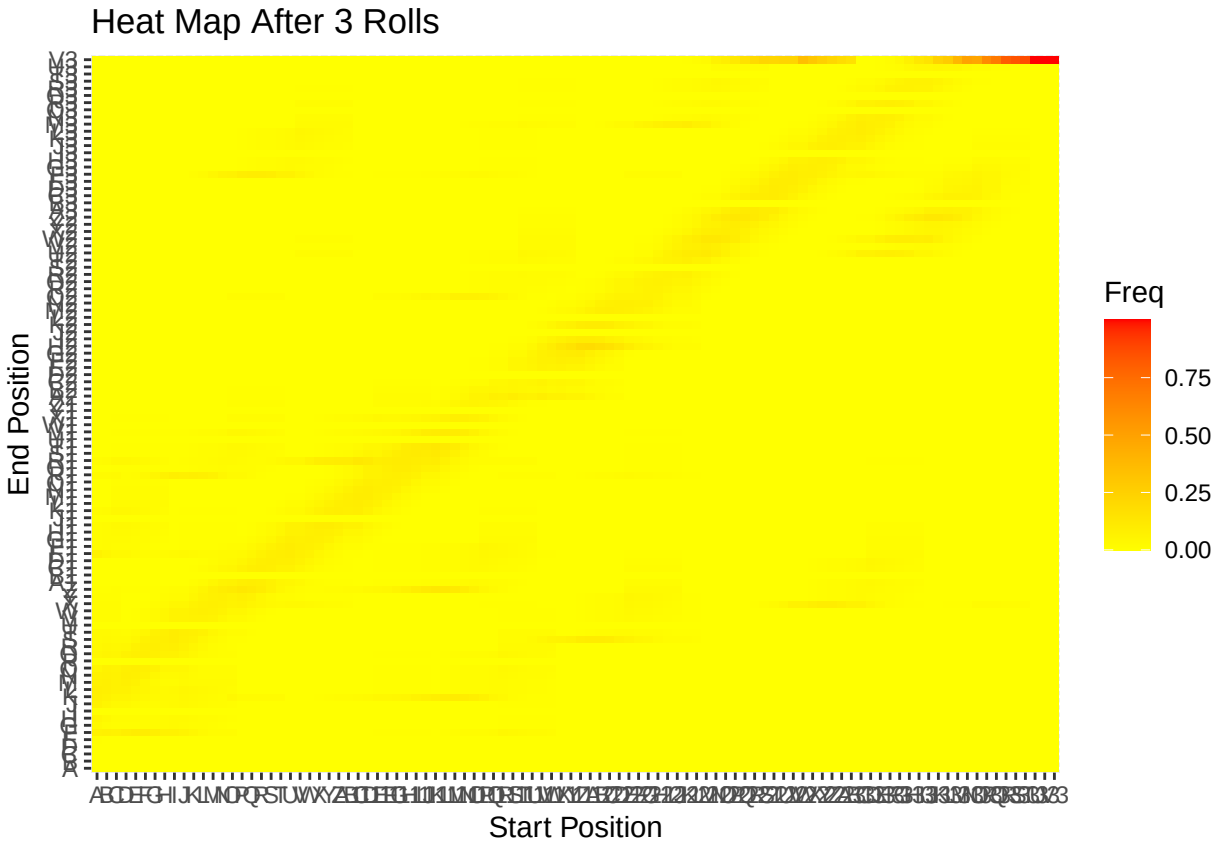
```r
library(ggplot2)

roll1 <- matrix(0, nrow = 100, ncol = 100)
# Find chute or ladder end --> find_special from part 5
special_end <- function(pos, board) {
  if (pos %in% board$chutes$start) {
    return(board$chutes$end[which(board$chutes$start == pos)])
  } else if (pos %in% board$ladders$start) {
      return(board$ladders$end[which(board$ladders$start == pos)])
  } else {
    return(pos)
  }
}
# Fill in the matrix
for (start in 1:100) {
  for (roll in 1:6) {
    end <- min(start + roll, 100)
    end <- special_end(end, board)
    roll1[start, end] <- roll1[start, end] + (1 / 6)
  }
}
# Calculate roll 2 and 3
roll2 <- roll1 %*% roll1
roll3 <- roll2 %*% roll1
# Create the heat map
plot_heat_map <- function(transition_mat, title) {
  transition_df <- as.data.frame(as.table(transition_mat))
  # Plot df
  ggplot(transition_df, aes(Var1, Var2, fill = Freq)) +
    geom_tile() +
    scale_fill_gradient(low = "yellow", high = "red") +
    labs(title = title, x = "Start Position", y = "End Position")
}
# Plot heat maps on rolls
par(mfrow = c(1, 3))
plot_heat_map(roll1, "Heat Map After 1 Roll")
```




```
plot_heat_map(roll2, "Heat Map After 2 Rolls")
```




```
plot_heat_map(roll3, "Heat Map After 3 Rolls")
```


Heat Map After 3 Rolls
End Position
Start Position
Freq
0.75
0.50
0.25
0.00