Stats 102A - Homework 2 Instructions

Homework questions and instructions copyright Miles Chen, Do not post, share, or distribute without permission.

Acknowledgments: Michael Tsiang and Jake Kramer for the idea of simulating chutes and ladders. Data Genetics for the Monte Carlo analysis http://datagenetics.com/blog/november12011/

Homework 2 Requirements

You will submit three files.

The files you submit will be:

1. `102a_hw_02_script_First_Last.R` Your R script file containing the functions you write for the homework assignment. Write comments as necessary. **Submit this file to Canvas / Bruin Learn**
2. `102a_hw_02_output_First_Last.qmd` Take the provided Quarto file and make the necessary edits so that it generates the requested output. The first line of your .qmd file should be to source the R script file you wrote. **Submit this file to Canvas / Bruin Learn**
3. `102a_hw_02_output_First_Last.pdf` Your output PDF file. This is the primary file that will be graded. Make sure all requested output is visible in the output file. **Submit this file to Gradescope**

Failure to submit all files will result in an automatic 40 point penalty.

At the top of your Quarto file, be sure to include the academic integrity statement.

Chutes and Ladders

In this homework assignment, you will create a simulation of the children's board game, Chutes and Ladders.

The game is simple. Players spin a spinner with number 1 through 6. Players move their piece that many spaces. Some spaces have ladders, and the player moves to a new location on the board. If you land at the bottom of a ladder, you move to the top of the ladder. Some spaces have chutes (slides) which causes the player to move backward. If you land at the top of a chute, you move to the bottom of the chute. The start/end of a chute/ladder has a picture on the given space (there is no ambiguity).

The first player to land exactly on space 100 wins the game.

Encouragement: The task at first can seem daunting (especially because the instructions themselves are so long!). Break it down into smaller pieces that you can handle.

Encouragement: The goal of the course and this assignment is your learning. The challenge you experience is important in your development as a coder. The more you practice coding, the better you will become at

coding. Time spent thinking and writing code is not wasted, even if you are unable to fully achieve all the requirements.

Advice: Follow the strategies I've laid out on writing functions given in lecture.

Here are the official rules for how to play:

Chutes and Ladders®

INSTRUCTIONS

For 2 to 4 Players/AGES 3+

This delightful game is simple and easy to play, even for children who can't read. Fun pictures help kids understand the rewards of doing good deeds as they climb up the ladders and the consequences of naughty ones as they slide down the chutes.

CONTENTS

Adult Assembly Required

• Gameboard • Spinner with plastic arrow
• 4 Pawns with plastic stands

OBJECT

Be the first player to reach "Winner" square #100.

THE FIRST TIME YOU PLAY

1. Punch out the 4 pawns from the cardboard sheet. Put each pawn into a plastic stand.

2. Punch out the spinner board from the paper sheet. Discard the waste. Carefully remove the spinner arrow and base from the plastic frame. If needed, use an emery board or sandpaper to remove the excess plastic from the game pieces. Discard the frame after removing all of the game pieces. Then assemble the spinner as shown in Figure 1.

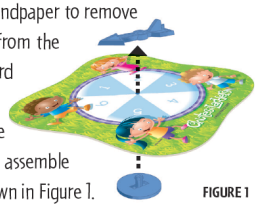


FIGURE 1

SETUP

Position the gameboard so all the players can easily move their pawns from square to square. Everyone chooses a pawn to play. Any extra pawns are out of play. Chosen pawns start off the board near square #1. Now get ready for the fun!

ALL ABOUT THE SQUARES:

Take a peek at the gameboard. The squares are numbered from 1 to 100. Players' pawns will move back and forth across the board, following the numbers upward – starting at square #1 and moving right toward square #10, then up to square #11 and left toward square #20, etc.

Of course, you can also move up by climbing ladders and sometimes go down, too, by sliding down chutes. More about that later.

HOW TO PLAY

Everyone spins the spinner. The player with the highest number goes first. Play proceeds to the left.

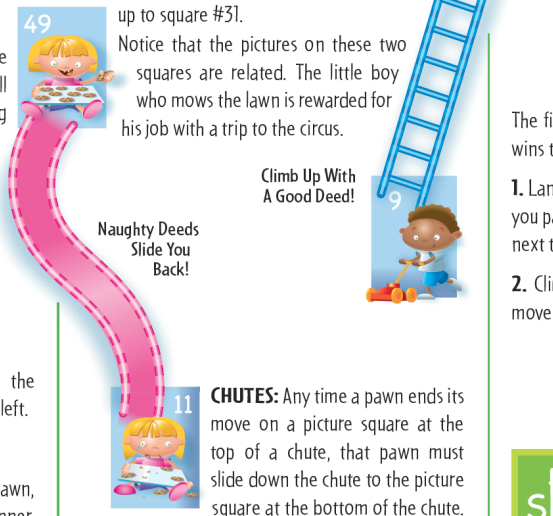
WHAT TO DO ON YOUR TURN:

On your turn, spin the spinner and move your pawn, square by square, the number shown on the spinner. For example, on your first turn, if you spin a 5, move to square #5 on the board. Once you move your pawn, your turn is over. NOTE: Two or more pawns may be on the same space at the same time.

GOING UP A LADDER OR DOWN A CHUTE

LADDERS: Any time a pawn ends its move on a picture square at the bottom of a ladder, that pawn must climb up to the picture square at the top of the ladder. For example, if you end your move on square #9, you can immediately move up to square #31.

Notice that the pictures on these two squares are related. The little boy who mows the lawn is rewarded for his job with a trip to the circus.



CHUTES: Any time a pawn ends its move on a picture square at the top of a chute, that pawn must slide down the chute to the picture square at the bottom of the chute.

For example, if you end your move on square #49, you must immediately move down to square #11. Again, the pictures are related. Eating too many cookies can give you a tummy-ache!

If your pawn ends its turn on any of the following spaces, your turn is over:

- a square with no picture
- a square with no picture and just an arrow
- a square that a ladder or chute just passes through
- a picture square at the top of a ladder
- a picture square at the bottom of a chute

WINNING THE GAME

The first player to reach the "Winner" square #100 wins the game. You can get there 2 ways:

1. Land there by exact count. If your spin would take you past square #100, don't move. Try again on your next turn.

2. Climb there by ending your move on ladder square #80.

MB MILTON BRADLEY

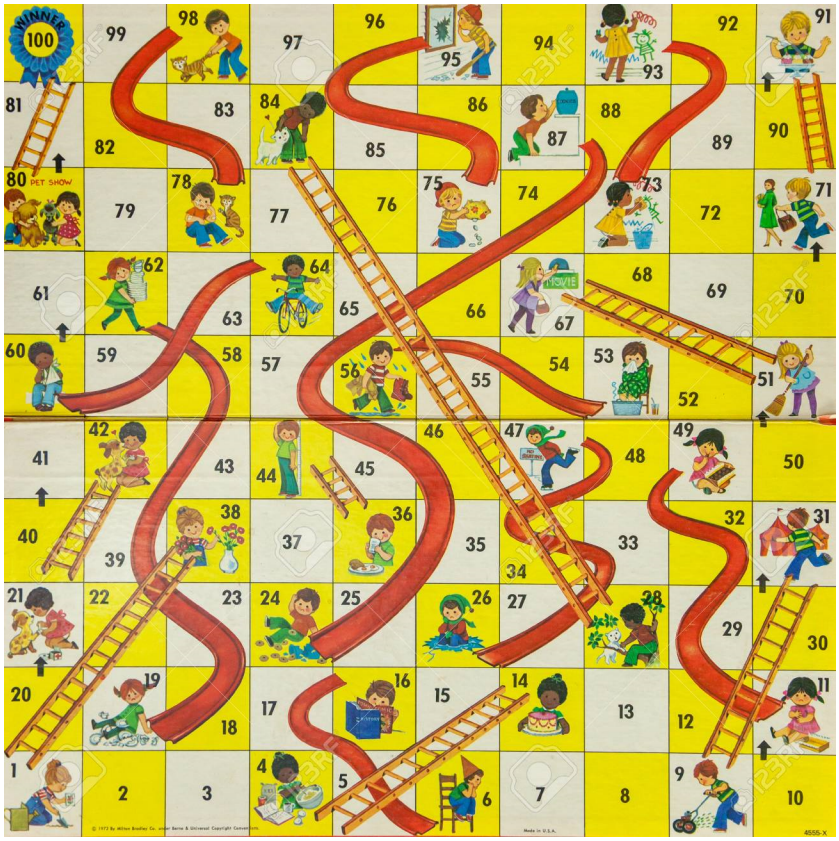
1144555MBU

⚠ Not suitable for children under 3 years because of small parts – choking hazard.

GAME PARTS STORED BELOW

Part 1

Here is a picture of the official game board.



In the Qmd file, create a single list object called `board` where you store the features of the game board in R.

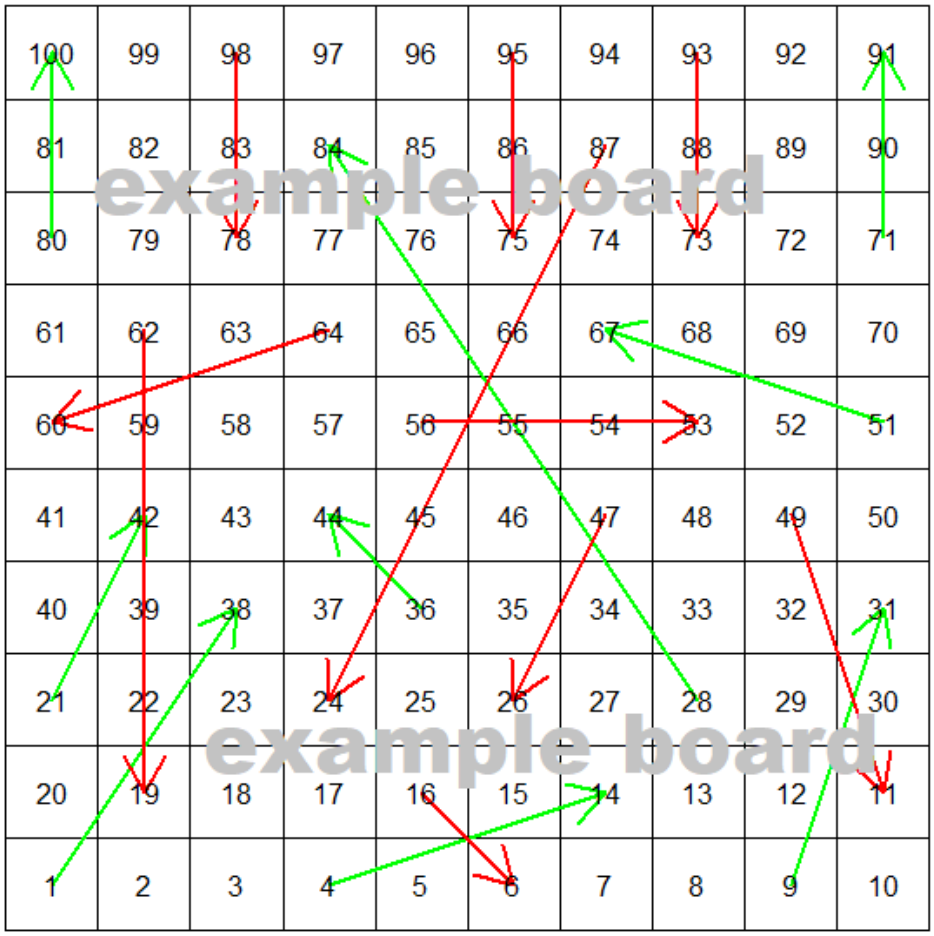
You will need to figure out how to represent and store the features of the game board in R.

You'll need to keep track of chutes, ladders, and the dimensions of the board.

Part 2

In the script file, create a function called `show_board()` that takes one argument `board` (the object you created in part 1). The function will then produce a plot showing the board.

The plot will produce a square grid with the spaces properly numbered. Note that the numbers reverse directions each row. Chutes will be represented with red arrows, and ladders should be represented with green arrows (`lwd = 2`). You may alternatively consider red and blue arrows (for colorblind accessibility). Try to maintain the aspect ratio for your board (i.e., each tile should be a square).



You should be able to achieve this using base graphics in R. You are not limited to using only the following functions, but I was able to create my game board with only: `plot.new()`, `plot.window()`, `segments()`, `arrows()`, and `text()`.

The function `show_board()` must be flexible enough to produce boards with different specifications, as seen in Part 3. You will not be given a board specification that is not rectangular. The smallest possible board size that might be tested is 2 x 2, and the largest that might be tested is 12 x 12.

In the Qmd file, call the `show_board()` function to produce the plot of the game board.

Part 3

This part is to test that your function is flexible to represent different boards.

Create another object **in the Qmd file** called `miniboard1` to represent another board with the following properties:

- Dimensions: 6 col x 7 row
- ladders: 1 to 23, 9 to 27, 20 to 31, 33 to 41
- chutes: 16 to 6, 30 to 18, 35 to 24, 40 to 26

In the Qmd file, call the `show_board()` function to produce the plot of the `miniboard1`.

Create another object in the Qmd file called `miniboard2` to represent another board with the following properties:

- Dimensions: 7 col x 9 rows
- ladders: 9 to 22, 13 to 30, 24 to 37, 29 to 41, 33 to 39, 43 to 54
- chutes: 16 to 3, 31 to 15, 35 to 21, 62 to 48

In the Qmd file, call the `show_board()` function to produce the plot of the `miniboard2`.

Create another object in the Qmd file called `miniboard3` to represent another board with the following properties:

- Dimensions: 8 col x 9 rows
- ladders: none
- chutes: none

In the Qmd file, call the `show_board()` function to produce the plot of the `miniboard3`.

Part 4

Create a function `play_solo()` that will simulate a game of chutes and ladders for one player on the full 10 x 10 board from parts 1 and 2. For each game, a player starts at position 0. You must land exactly on 100 to win–if you go over, you don't move and take another turn.

The function will accept the following arguments:

- `board` to represent the board (the same argument used in `show_board()`)
- `verbose` with default value `FALSE` to represent whether the function should display every move to the screen

The function will output the following values in a list:

- `turns`: how many turns it took to complete a game
- `chute_tally`: a vector of length `j` with the count of how many times each chute was used. `j` is the number of chutes that exist in the board. If a chute is never used in the game, the tally for that chute is 0. Number the chutes and ladders in order of their starting square.
- `ladder_tally`: a vector of length `k` with the count of how many times each ladder was used. `k` is the number of ladders that exist in the board.
- `move_log`: a vector containing a record or log of all the spaces a player landed on. If a player rolls over 100 and doesn't move, record the same position in consecutive turns (e.g., 99, 99, …).

The spinner has values 1 through 6 and each number shows up with equal probability. You can use the following function to simulate a spin.

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

More Details

The argument `verbose` will 'narrate' the moves of a player.

When `verbose` is `TRUE`, the output of the function should show for each turn:

- The start of each turn begins with the turn number: `Turn x`
- It announces where the player is located at the start of the turn: `Start at x`
- It announces what the player spun: `Spinner: x`
  - **If** the player lands on a special space, it says `Landed on: x`
  - It will announce `Chute!` or `Ladder!` depending on what type of special space it is.
- It announces where the player ends the turn: `Turn ends at: x`

You can look at my sample output to get an idea of how it should appear. Your output does not need to be in the exact order as mine, but all required components do need to be present.

```
> set.seed(10)
> play_solo(board, TRUE)
Turn 1
Start at 0
Spinner: 3
Turn ends at: 3

Turn 2
Start at 3
Spinner: 1
Landed on: 4
Ladder!
Turn ends at: 14

Turn 3
Start at 14
Spinner: 2
Landed on: 16
Chute!
Turn ends at: 6

Turn 4
Start at 6
Spinner: 4
Turn ends at: 10
```

```
Turn 5
Start at 10
Spinner: 6
Landed on: 16
Chute!
Turn ends at: 6

Turn 6
Start at 6
Spinner: 3
Landed on: 9
Ladder!
Turn ends at: 31

Turn 7
Start at 31
Spinner: 2
Turn ends at: 33

Turn 8
Start at 33
Spinner: 2
Turn ends at: 35

Turn 9
Start at 35
Spinner: 2
Turn ends at: 37

Turn 10
Start at 37
Spinner: 5
Turn ends at: 42

Turn 11
Start at 42
Spinner: 6
Turn ends at: 48

Turn 12
Start at 48
Spinner: 6
Turn ends at: 54

Turn 13
Start at 54
Spinner: 3
Turn ends at: 57

Turn 14
```

```
Start at 57
Spinner: 6
Turn ends at: 63

Turn 15
Start at 63
Spinner: 2
Turn ends at: 65

Turn 16
Start at 65
Spinner: 5
Turn ends at: 70

Turn 17
Start at 70
Spinner: 5
Turn ends at: 75

Turn 18
Start at 75
Spinner: 5
Landed on: 80
Ladder!
Turn ends at: 100

$turns
[1] 18

$chute_tally
 [1] 2 0 0 0 0 0 0 0 0 0

$ladder_tally
[1] 0 1 1 0 0 0 0 0 1

$move_log
 [1]   3  14   6  10   6  31  33  35  37  42  48  54  57  63  65  70  75 100
```

When `verbose = FALSE`, it will only return the contents of the list:

```
> set.seed(10)
> play_solo(board, FALSE)
$turns
[1] 18

$chute_tally
 [1] 2 0 0 0 0 0 0 0 0 0

$ladder_tally
[1] 0 1 1 0 0 0 0 0 1
```

```
$move_log
 [1]   3  14   6  10   6  31  33  35  37  42  48  54  57  63  65  70  75 100
```

To demonstrate that your `play_solo()` function works, produce the verbose output for a game with `set.seed(5)` which has a total of 28 moves.

If you use the same random seed as me, you should get the following output for the sequence of spins.

```
set.seed(5)
replicate(28, sample(6,1))
 [1] 2 3 1 3 1 1 5 6 3 3 6 2 5 4 2 5 3 1 6 4 3 2 5 2 2 3 1 2
```

If you are not getting the same results for `set.seed(5)`, double check your RStudio Global Options. Make sure you have `Restore .Rdata into workspace at startup` unchecked and restart RStudio.

Part 5 - Monte Carlo Study

After you finish programming `play_solo()`, you will use it to perform a Monte Carlo study of the game Chutes and Ladders.

We wish to learn the distribution of how many turns it takes to complete a game. We also want to learn which chutes are used most frequently, and which ladders are used most frequently.

Simulate 10,000 solo games. (Make sure `verbose = FALSE`!) Do not reset the seed from Part 4.

- Keep track of how many turns it took to complete the game. Also keep track of the chute and ladder tallies.
- Create a histogram (breaks = 50) of the turns.
- Find the minimum number of turns. How many times out of 10,000 did a game finish with the minimum number of turns?
- Find the maximum number of turns.
- What is the median number of turns?
- What is the mean number of turns?
- What proportion of games take 100 or more turns to complete?
- What proportion of games take 10 or fewer turns to complete?
- What proportion of games utilize ladder 9 (the shortcut to win on space 80)?
- 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.)
- 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.)

[A Monte Carlo study like this has already been performed and the results are published here: http://datagenetics.com/blog/november12011/. Take note one of the chutes (snakes) is different - our chute goes from 47 to 26, while theirs goes from 48 to 26. Also, the author numbers the chutes in reverse order, while I have numbered them in ascending order.]

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 .Qmd 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.