```
# Yannique Hecht
# Harvardx: PH125.3 - (3) Data Science: Probability
# SECTION 1: DISCRETE PROBABILITY
# ASSESSMENTS


# # # ASSESSMENT 1.1: INTRODUCTION TO DISCRETE PROBABILITY

# # EXERCISE 1 - Probability of cyan
[redacted]

# # EXERCISE 2 - Probability of not cyan
[redacted]

# # EXERCISE 3 - Sampling without replacement
[redacted]

# # EXERCISE 4 - Sampling with replacement
[redacted]



# # # ASSESSMENT 1.1: INTRODUCTION TO DISCRETE PROBABILITY - DATACAMP

# # EXERCISE 1 - Probability of cyan - generalized
[redacted]
# Assign a variable `p` as the probability of choosing a cyan ball
 from the box
[redacted]
# Print the variable `p` to the console
[redacted]

# # EXERCISE 2 - Probability of not cyan - generalized
# `p` is defined as the probability of choosing a cyan ball from a
 box containing: 3 cyan balls, 5 magenta balls, and 7 yellow balls.
# Using variable `p`, calculate the probability of choosing any ball
 that is not cyan from the box
[redacted]

# # EXERCISE 3 - Sampling without replacement - generalized
[redacted]
# The variable `p_1` is the probability of choosing a cyan ball from
```

```
 the box on the first draw.
[redacted]
# Assign a variable `p_2` as the probability of not choosing a cyan
 ball on the second draw without replacement.
[redacted]
# Calculate the probability that the first draw is cyan and the
 second draw is not cyan using `p_1` and `p_2`.
[redacted]

# # EXERCISE 4 - Sampling with replacement - generalized
[redacted]

# The variable 'p_1' is the probability of choosing a cyan ball from
 the box on the first draw.
[redacted]

# Assign a variable 'p_2' as the probability of not choosing a cyan
 ball on the second draw with replacement.
[redacted]
# Calculate the probability that the first draw is cyan and the
 second draw is not cyan using `p_1` and `p_2`.
[redacted]



# # # ASSESSMENT 1.2: COMBINATIONS & PERMUTATIONS - DATACAMP

# # EXERCISE 1 - Independence


# # EXERCISE 2 - Sampling with replacement
[redacted]
# Assign the variable 'p_yellow' as the probability that a yellow
 ball is drawn from the box.
[redacted]
# Using the variable 'p_yellow', calculate the probability of drawing
 a yellow ball on the sixth draw. Print this value to the console.
[redacted]


# # EXERCISE 3 - Rolling a die
# Assign the variable 'p_no6' as the probability of not seeing a 6 on
```

a single roll.
[redacted]

# Calculate the probability of not seeing a 6 on six rolls using `p_no6`. Print your result to the console: do not assign it to a variable.
[redacted]

# # EXERCISE 4 - Probability the Celtics win a game
# Assign the variable `p_cavs_win4` as the probability that the Cavs will win the first four games of the series.
[redacted]
# Using the variable `p_cavs_win4`, calculate the probability that the Celtics win at least one game in the first four games of the series.
[redacted]

# # EXERCISE 5 - Monte Carlo simulation for Celtics winning a game
# This line of example code simulates four independent random games where the Celtics either lose or win. Copy this example code to use within the `replicate` function.
[redacted]
# The variable 'B' specifies the number of times we want the simulation to run. Let's run the Monte Carlo simulation 10,000 times.
[redacted]
# Use the `set.seed` function to make sure your answer matches the expected result after random sampling.
[redacted]
# Create an object called `celtic_wins` that replicates two steps for B iterations: (1) generating a random four-game series `simulated_games` using the example code, then (2) determining whether the simulated series contains at least one win for the Celtics. Put these steps on separate lines.
[redacted]
# Calculate the frequency out of B iterations that the Celtics won at least one game. Print your answer to the console.
[redacted]

```
# # # ASSESSMENT 1.3: THE ADDITION RULE & MONTY HALL - DATACAMP

# # EXERCISE 1 - The Cavs and the Warriors
# Assign a variable 'n' as the number of remaining games.
[redacted]
# Assign a variable `outcomes` as a vector of possible game outcomes,
 where 0 indicates a loss and 1 indicates a win for the Cavs.
outcomes <- c(0,1)
# Assign a variable `l` to a list of all possible outcomes in all
 remaining games. Use the `rep` function on `list(outcomes)` to
 [redacted]
[redacted]
# Create a data frame named 'possibilities' that contains all
 combinations of possible outcomes for the remaining games.
[redacted]
# Create a vector named 'results' that indicates whether each row in
 the data frame 'possibilities' contains enough wins for the Cavs to
 win the series.
[redacted]
# Calculate the proportion of 'results' in which the Cavs win the
 series. Print the outcome to the console.
[redacted]

# # EXERCISE 2 - The Cavs and the Warriors - Monte Carlo
# The variable `B` specifies the number of times we want the
 simulation to run. Let's run the Monte Carlo simulation 10,000
 times.
[redacted]
# Use the `set.seed` function to make sure your answer matches the
 expected result after random sampling.
[redacted]
# Create an object called `results` that replicates for `B`
 iterations a simulated series and determines whether that series
 contains at least four wins for the Cavs.
[redacted]
   [redacted]
   [redacted]
[redacted]
# Calculate the frequency out of `B` iterations that the Cavs won at
 least four games in the remainder of the series. Print your answer
 to the console.
[redacted]

# # EXERCISE 3 - A and B play a series - part 1
# Let's assign the variable 'p' as the vector of probabilities that
```

team A will win.

[redacted]

# Given a value 'p', the probability of winning the series for the underdog team B can be computed with the following function based on a Monte Carlo simulation:

[redacted]

# Apply the 'prob_win' function across the vector of probabilities that team A will win to determine the probability that team B will win. Call this object 'Pr'.

[redacted]

# Plot the probability 'p' on the x-axis and 'Pr' on the y-axis.

[redacted]

# # EXERCISE 4 - A and B play a series - part 2

# Given a value 'p', the probability of winning the series for the underdog team $B$ can be computed with the following function based on a Monte Carlo simulation:

[redacted]

# Assign the variable 'N' as the vector of series lengths. Use only odd numbers ranging from 1 to 25 games.

[redacted]

# Apply the 'prob_win' function across the vector of series lengths to determine the probability that team B will win. Call this object `Pr`.

[redacted]

# Plot the number of games in the series 'N' on the x-axis and 'Pr' on the y-axis.

[redacted]