```
# Yannique Hecht
# Harvardx: PH125.3 - (3) Data Science: Probability
# SECTION 3: RANDOM VARIABLES, SAMPLING MODELS, & CENTRAL LIMIT
 THEOREM
# ASSESSMENTS


# # # ASSESSMENT 3.1: RANDOM VARIABLES & SAMPLING MODELS - DATA CAMP

# # EXERCISE 1 - American Roulette probabilities
# The variables 'green', 'black', and 'red' contain the number of
 pockets for each color
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Print the variable `p_green` to the console
[redacted]


# # EXERCISE 2 - American Roulette payout
# Use the `set.seed` function to make sure your answer matches the
 expected result after random sampling.
[redacted]
# The variables 'green', 'black', and 'red' contain the number of
 pockets for each color
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Create a model to predict the random variable `X`, your winnings
 from betting on green. Sample one time.
[redacted]
# Print the value of `X` to the console
[redacted]

# # EXERCISE 3 - American Roulette expected value
# The variables 'green', 'black', and 'red' contain the number of
 pockets for each color
```

```
[redacted]
# Assign a variable `p_green` as the probability of the ball landing in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not landing in a green pocket
[redacted]
# Calculate the expected outcome if you win $17 if the ball lands on green and you lose $1 if the ball doesn't land on green
[redacted]

# # EXERCISE 4 - American Roulette standard error
# The variables 'green', 'black', and 'red' contain the number of pockets for each color
[redacted]

# Assign a variable `p_green` as the probability of the ball landing in a green pocket
[redacted]

# Assign a variable `p_not_green` as the probability of the ball not landing in a green pocket
[redacted]

# Compute the standard error of the random variable
[redacted]

# # EXERCISE 5 - American Roulette sum of winnings
# The variables 'green', 'black', and 'red' contain the number of pockets for each color
[redacted]
# Assign a variable `p_green` as the probability of the ball landing in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not landing in a green pocket
[redacted]
# Use the `set.seed` function to make sure your answer matches the expected result after random sampling
set.seed(1)
```

```
# Define the number of bets using the variable 'n'
[redacted]
# Create a vector called 'X' that contains the outcomes of 1000
 samples
[redacted]
# Assign the sum of all 1000 outcomes to the variable 'S'
[redacted]
# Print the value of 'S' to the console
[redacted]

# # EXERCISE 6 - American Roulette winnings expected value
# The variables 'green', 'black', and 'red' contain the number of
 pockets for each color
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Define the number of bets using the variable 'n'
[redacted]
# Calculate the expected outcome of 1,000 spins if you win $17 when
 the ball lands on green and you lose $1 when the ball doesn't land
 on green
[redacted]

# # EXERCISE 7 - American Roulette winnings expected value
# The variables 'green', 'black', and 'red' contain the number of
 pockets for each color
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Define the number of bets using the variable 'n'
[redacted]
# Compute the standard error of the sum of 1,000 outcomes
[redacted]
```

```
# # # ASSESSMENT 3.2: CENTRAL LIMIT THEOREM - DATA CAMP

# # EXERCISE 1 - American Roulette Monte Carlo simulation
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Define the number of bets using the variable 'n'
[redacted]
# Calculate 'avg', the expected outcome of 100 spins if you win $17
 when the ball lands on green and you lose $1 when the ball doesn't
 land on green
[redacted]
# Compute 'se', the standard error of the sum of 100 outcomes
[redacted]
# Using the expected value 'avg' and standard error 'se', compute the
 probability that you win money betting on green 100 times.
[redacted]


# # EXERCISE 2 - American Roulette probability of winning money
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Define the number of bets using the variable 'n'
[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 `S` that replicates the sample code for `B`
 iterations and sums the outcomes.
[redacted]
```

```
[redacted]
# Compute the average value for 'S'
[redacted]
# Calculate the standard deviation of 'S'
[redacted]

# # EXERCISE 3 - American Roulette Monte Carlo vs CLT
# Calculate the proportion of outcomes in the vector `S` that exceed
 $0
[redacted]

# # EXERCISE 4 - American Roulette Monte Carlo vs CLT comparison


# # EXERCISE 5 - American Roulette average winnings per bet
# Use the `set.seed` function to make sure your answer matches the
[redacted]
# Define the number of bets using the variable 'n'
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Create a vector called `X` that contains the outcomes of `n` bets
[redacted]
# Define a variable `Y` that contains the mean outcome per bet. Print
 this mean to the console.
[redacted]

# # EXERCISE 6 - American Roulette per bet expected value
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Use the expected value formula to calculate the expected outcome of
 `Y`, the mean outcome per bet in 10,000 bets
[redacted]
```

```
# # EXERCISE 7 - American Roulette per bet standard error
# Define the number of bets using the variable 'n'
[redacted]
# Assign a variable `p_green` as the probability of the ball landing
 in a green pocket
[redacted]
# Assign a variable `p_not_green` as the probability of the ball not
 landing in a green pocket
[redacted]
# Compute the standard error of 'Y', the mean outcome per bet from
 10,000 bets.
[redacted]

# # EXERCISE 8 - American Roulette winnings per game are positive
# We defined the average using the following code
[redacted]
# We defined standard error using this equation
[redacted]
# Given this average and standard error, determine the probability of
 winning more than $0. Print the result to the console.
[redacted]

# # EXERCISE 9 - American Roulette Monte Carlo again
## Make sure you fully follow instructions, including printing values
 to the console and correctly running the `replicate` loop. If not,
 you may encounter "Session Expired" errors.
# The variable `n` specifies the number of independent bets on green
[redacted]
# The variable `B` specifies the number of times we want the
 simulation to run
[redacted]
# Use the `set.seed` function to make sure your answer matches the
 expected result after random number generation
[redacted]
# Generate a vector `S` that contains the the average outcomes of
 10,000 bets modeled 10,000 times
[redacted]
# Compute the average of `S`. Print this value to the console.
[redacted]
# Compute the standard deviation of `S`. Print this value to the
 console.
[redacted]
```

```
# # EXERCISE 10 - American Roulette comparison
# Compute the proportion of outcomes in the vector 'S' where you won
 more than $0
[redacted]



# # # ASSESSMENT 2.2: CONTINUOUS PROBABILITY - Questions 1 and 2: ACT
 scores, part 1

# # EXERCISE 1a -
[redacted]

# # EXERCISE 1b -
[redacted]

# # EXERCISE 1c -
[redacted]

# # EXERCISE 1d -
[redacted]

# # EXERCISE 1e -
[redacted]

# # EXERCISE 2 -
[redacted]



# # # SECTION 3.3: RANDOM VARIABLES, SAMPLING MODELS, & CENTRAL LIMIT
 THEOREM ASSESSMENT


# # Questions 1 and 2: SAT testing

# # Q1a
[redacted]
```

[redacted]

Q1b
[redacted]

Q1c
[redacted]

Q1d
[redacted]

Q1e
[redacted]

Q1f
[redacted]

Q2a
[redacted]

Q2b
[redacted]

Q2c
[redacted]

[redacted]

[redacted]

Question 3: Betting on Roulette

Q3a
[redacted]

Q3b
[redacted]

Q3c
[redacted]

Q3d
[redacted]

Q3e
[redacted]

Q3f
[redacted]

Q3g
[redacted]