# STAT 491: Final Exam
# Name:

1. **Format**: Submit the exam to GitHub and include the R Markdown code and a PDF file with output. Please verify that all of the code has compiled and the graphics look like you think they should on your files, you are welcome to upload image files directly if they look distorted in the output file.
2. **Advice**: Be sure to adequately justify your answers and appropriately reference any sources used. Even if you are not able to answer a question completely, do your best to provide an answer and discuss solutions that you tried. For full credit, include your code and graphics for each question and create neat output by using options like `kable()` for tables and writing results in line with R commands.
3. **Computer Code / Reproducibility:** Please turn in all relevant computer code to reproduce your results; a reproducible document is a requirement. Include all relevant code and output needed to answer each question and write an answer to each question. Even if the answer seems obvious from the output, make sure to state it in your narrative as well.
4. **Resources and Citations:** While the exam is open book and you can use any resources from class or freely available on the internet, this is strictly an individual endeavor and **you should not discuss the problems with anyone outside the course instructor including class members.** All resources, including websites, should be acknowledged.
5. **Exam Questions:** If clarification on questions is required, please email the course instructor: andrew.hoegh@montana.edu.
6. **A note on sharing / reusing code:** This is a huge volume of code is available on the web to solve any number of problems. For this exam you are allowed to make use of any online resources (e.g., StackOverflow) but you must explicitly cite where you obtained any code you directly use (or use as inspiration). Any recycled code that is discovered and is not explicitly cited will be treated as plagiarism. All communication with classmates is explicitly forbidden.

## Academic Honesty Statement

Include the following statement at the beginning of your submission.

> I, ___ (your full name here) ___, hereby state that I have not communicated with or gained information in any way from my classmates or anyone other than the course instructor during this exam, and that all work is my own.

In the event that you have inadvertently violated the above statement, you should not sign above and instead discuss the situation with the course instructor.

## Bunco Monte Carlo Question (12 points)

Bunco is a game where a group of players compete by rolling dice. The game consists of 6 rounds where players take turns rolling three dice. Each round ends when a player rolls a bunco (three dice that all match the round number) or a player reaches 21 points. Points can be scored in the following manner:

- Each die that matches the round number results in a point.
- Three dice of the same number (but not the round number) earn five points.

If a player has a roll that scores points, they get another roll before the next player's turn.

---

Here is an example of a round that takes 2 turns:

**Turn 1:**

John:

- rolls `1`, `2`, `4` - scores 1 point: Score John 1, Ian 0
- rolls `2`, `3`, `6` - turn ends: Score John 1, Ian 0

Ian:

- rolls `3,5,5` - turn ends: Score John 1, Ian 0

**Turn 2:**

John:

- rolls `1`, `1`, `4` - scores 2 points: Score John 3, Ian 0
- rolls `4`, `4`, `4` - scores 5 points: Score John 8, Ian 0
- rolls `3`, `6`, `6` - turn ends: Score John 8, Ian 0

Ian:

- rolls `1,1,1` - bunco!: Ian wins

---

We are interested in estimating how many turns we'd expect a round of bunco to last. In particular, use a Monte Carlo method to estimate a distribution for the number of turns it takes for a round of bunco to end. Note: this won't depend on how many players are playing, so you can simplify your analysis by focusing on a single player

**A. (4 points)** Describe how simulation can be used to answer this question and why this is a valid approach.

**B. (8 points)** Write code to estimate the distribution of the number of turns a round of bunco will last. To display your final answer, create a histogram. To get you started, I've written a function that will return the number of points for a roll of three dice - given the round number.

Note: I will award partial credit for this problem. If you get stuck, describe the problem you are running into and potential solutions you explored.

```
bunco_points <- function(roll, round){
  if (length(unique(roll)) == 1 & roll[1] == round){
    return(21)
  } else if (length(unique(roll)) == 1){
    return(5)
  } else {
    return(sum(roll == round))
```

```r
  }
}

bunco_points(c(1,1,1),1)
```

```
## [1] 21
```

```r
bunco_points(c(1,3,1),1)
```

```
## [1] 2
```

```r
bunco_points(c(3,3,3),1)
```

```
## [1] 5
```

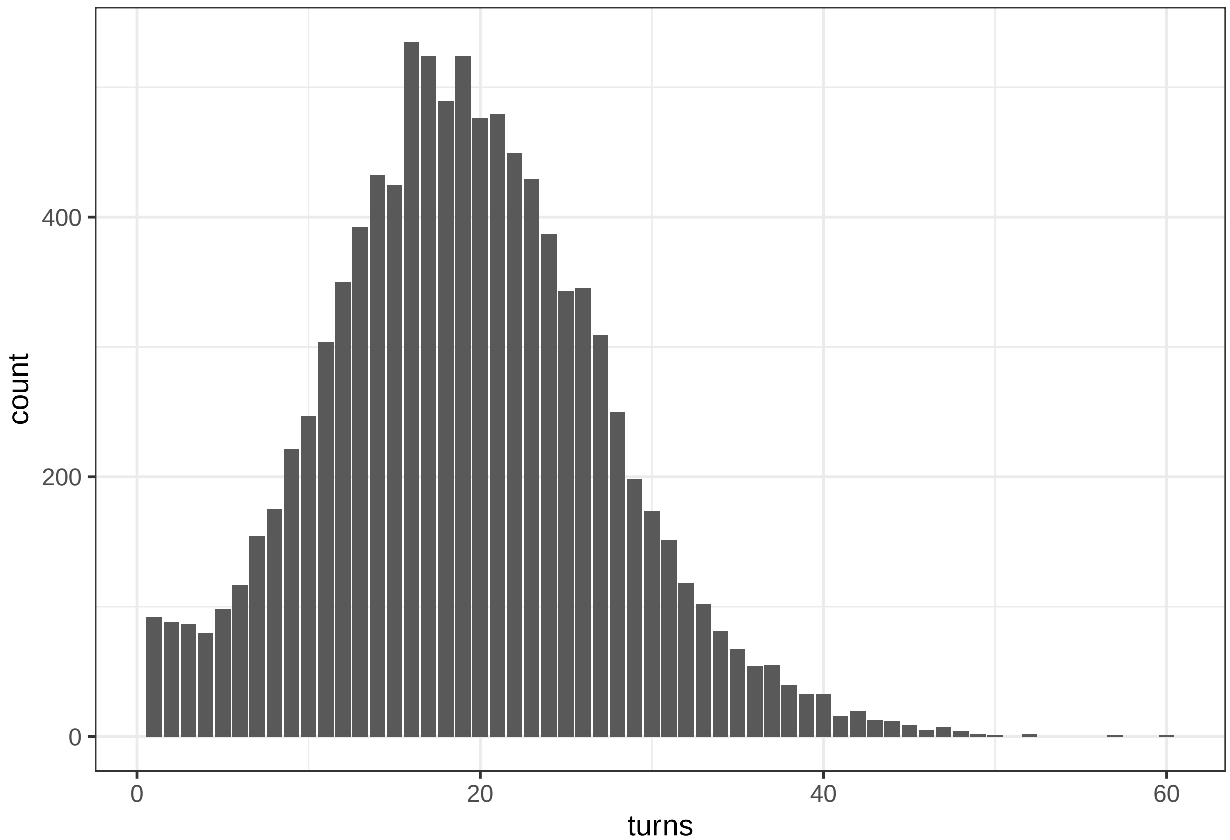
```r
bunco_points(c(3,3,4),1)
```

```
## [1] 0
```

```r
round <- 1
bunco_function <- function(round){
  turns <- 0
  total <- 0
  while (total < 21){
    turns <- turns + 1
   # print(paste('Turn #', turns))
    roll <- sample(6,3,T)
   # print(roll)
    new_points <- bunco_points(roll, round)
    total <- total + new_points
    while (new_points > 0){
      roll <- sample(6,3,T)
     # print(roll)
      new_points <- bunco_points(roll, round)
      total <- total + new_points
    }
   # print(paste(total, ' points'))
  }
  return(turns)
}

tmp <-replicate(10000, bunco_function(1))

library(tidyverse)
tibble(turns = tmp) %>%
  ggplot(aes(x = turns)) +
  geom_bar()  +
  theme_bw()
```




**Part 2. (24 points)**

Now we will consider a dataset that contains rice prices from May 2019 and May 2020. The data has been collected across markets spread across the globe. The prices are standardized, but you can consider them to be in US Dollars / kilogram.

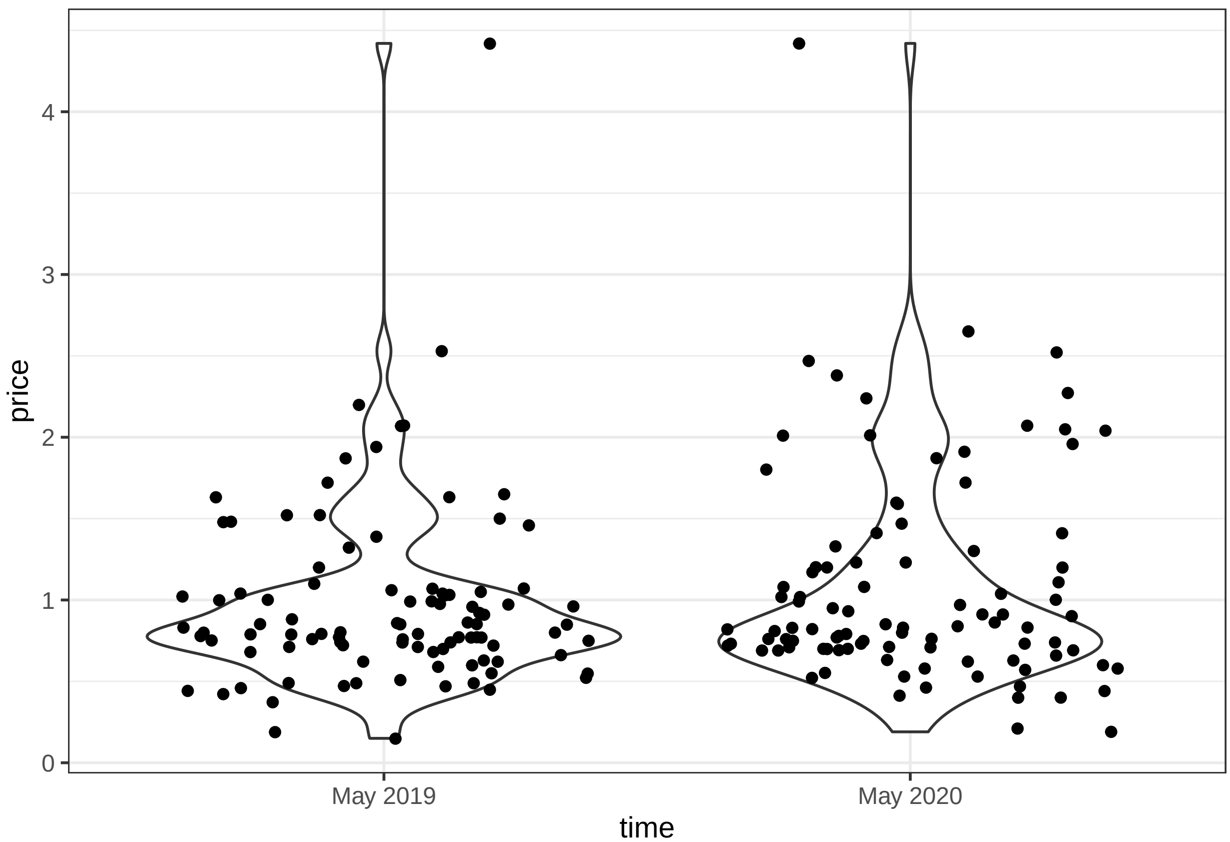
```
# food_data <- read_csv("food_prices.csv") %>%
#   filter(commodity == 'Rice', price_type == 'Retail') %>%
#   mutate(date = mdy(month), Month = month(date), Year = year(date)) %>%
#   filter(Year %in% c(2019, 2020), Month == 5)
#
# removals <- food_data %>% filter(is.na(price))
#
# removals2 <- food_data %>% group_by(market) %>% tally() %>% filter(n == 1)
#
# food_data <- food_data %>% filter(!market %in% removals$market,
#                                   !market %in% removals2$market) %>%
#   mutate(time = paste('May', Year)) %>%
#   dplyr::select(country, market, commodity, price,time)
#
#
#
# write_csv(food_data, 'rice_prices.csv')
rice_prices <- read_csv('rice_prices.csv')
```

```
## Rows: 194 Columns: 5
## -- Column specification --------------------------------------------------------
```

```
## Delimiter: ","
## chr (4): country, market, commodity, time
## dbl (1): price
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

**A. (4 points)** Create a side-by-side visualization of rice prices from 2019 and 2020.

```
rice_prices %>%
  ggplot(aes(y = price, x = time)) +
  geom_violin() +
  geom_jitter() +
  theme_bw()
```




**b. (4 points)** Write out a statistical model to investigate price differences between 2019 and 2020.

$$price = \beta_0 + \beta_1 I_{2020} + \epsilon; \epsilon \sim N(0, \sigma^2)$$

**c. (4 points)** Identify the parameters from the model in 2B and justify prior distributions.

- $\beta_0 \sim N(1, 2)$
- $\beta_1 \sim N(0, 2)$
- $\sigma \sim Unif(0, 10)$

**d. (4 points)** Write JAGS code to fit this model

```
rice_model <- 'model{
for (i in 1:n){
y[i] ~ dnorm(beta0 + beta1 * x[i], 1/sigma^2)
}

beta0 ~ dnorm(1, 1/2)
beta1 ~ dnorm(0, 1/2)
sigma ~ dunif(0, 10)
}'

library(rjags)
```

```
## Loading required package: coda
```

```
##
## Attaching package: 'coda'
```

```
## The following object is masked from 'package:arm':
##
##     traceplot
```

```
## Linked to JAGS 4.3.0
```

```
## Loaded modules: basemod,bugs
```

```
x <- model.matrix(price ~ time, data = rice_prices)[,2]

dataList = list(y = rice_prices$price, n = nrow(rice_prices),
                x = x)

model <- jags.model(file = textConnection(rice_model), data = dataList)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 194
##    Unobserved stochastic nodes: 3
##    Total graph size: 403
##
## Initializing model
```

```
update(model, n.iter = 5000) # warmup


posterior_sample <- coda.samples(model,
                       variable.names = c("beta0", 'beta1', 'sigma'),
                       n.iter = 10000)
```

**e. (4 points)** Summarize your parameters in the model.

```
summary(posterior_sample)
```

```
##
## Iterations = 6001:16000
## Thinning interval = 1
```

```
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##         Mean      SD  Naive SE Time-series SE
## beta0 0.9757 0.06253 0.0006253      0.0010574
## beta1 0.1027 0.08880 0.0008880      0.0014978
## sigma 0.6174 0.03184 0.0003184      0.0004063
##
## 2. Quantiles for each variable:
##
##           2.5%     25%    50%    75%  97.5%
## beta0  0.85324 0.93334 0.9758 1.0183 1.0982
## beta1 -0.07129 0.04308 0.1024 0.1619 0.2786
## sigma  0.55803 0.59536 0.6160 0.6380 0.6833
```

**f. (4 points)** Using your results from Q2e, address how rice prices from 2019 and 2020 differ.