Instructions:

- Follow the instructions indicated in this project carefully.
- This project should be considered to be an exam. That means that you should do your own work and not work with others.
- You should not speak about this exam to other students in the class.
- You may use class resources and other `R` and `Python` related resources found in the library or internet.
- Any indications of cheating on this project will result in a failure in the course.
- The project should be presented as a document using the `R` markdown environment for `R` and the same environment used in the `Python` homework assignments for the `Python` portion. All code should be included along with any relevant output and commentary on the output. This markdown document should also include the help narrative.
- There should be one PDF file for the `R` code, one for the `Python` code, and one for additional information requested (such as the `R` help file and the answers to questions 4 and 5).
- This project needs to be completed in both `R` and `Python` to test what you have learned during this course! The Help file is ONLY in `R`.
- Some concepts may be new to you, like making a help file, review other help files to learn what goes into them. Also, anova is new, example code for `R` that is needed is provided below. Note: All example code provided below is for `R`.
- Relax, take your time, and have fun!

The snow has begun to fall. You find yourself in an extreme mood of annoyance as you know that the drive from your consulting office to your home will be cold, slow, and full of wintery shadows of stranded motorists. You hope that you are not among them. But alas, you sigh. You know that randomness is a cruel beast that is not easily tamed. The meteorologists that were so confident that the prevailing winds would switch direction and spare your life of the fluffy flakes have had to express the usual shrug of a shoulder and admit that things do not always work out as imagined. The tea is hot, a nice Earl grey that was given to you by your last client. You decide to wait out the commute a little longer. Maybe the plows will do their job and the traffic will ease and you can expend less effort if you wait for an hour before starting for home. Your desk phone rings, shattering your silent contemplation of the gathering layers of snow on the window sill. The desk phone hardly ever rings any longer. Most clients have your cell number, and so you realize this must be a client from your past.

A strong and resourceful voice is on the line, and you realize it is your longtime friend Gertrude Cox, who was the founder of the Department of Experimental Statistics at North Carolina State University. Her most important and influential research deals with experimental design, and no doubt she is looking for someone to do some numerical research for her. Professor Cox has a long history of working with designed experiments, particularly those related to agricultural experiments. She tells you that she is interested in empirically studying the effect of different types of error distributions on testing in analysis of variance situations.


Gertrude Mary Cox (1900 – 1978)

The experiment that she is interested in is looking at data that comes from field experiments with corn. In the experiment there are three possible fertilizers that are to be studied in terms of their effect on the yield of a standard hybrid of feed corn. Each of the three fertilizers are to be used on a single corn field. At the end of the season, the yield for each fertilizer will be observed. It is well known that because of soil variation and localized environmental conditions the yield can also be affected by the part of the field that the corn is planted in. Because of this the three fertilizers are planted according to a *Latin square* design. This design divides the field into three rows and three columns yielding nine squares where the different fertilizers are planted. The design is structured in such a way that each fertilizer will appear once in each row and each column of the field. The particular design used is given in the table below:

	Column		
Row	1	2	3
1	A	B	C
2	C	A	B
3	B	C	A

In the table the three fertilizers are labeled as A, B, and C. Hence, Fertilizer A is used in Row 1 and Column 1 of the field. Similarly, Fertilizer B is used in Row 1 and Column 2 of the field, and so on until the entire field is fertilized by one of the three fertilizers. The observed data is then represented by $Y_{k\ell m}$, which represents the observed yield (in bushels per acre) using fertilizer k in row ℓ and column m where $k \in \{A, B, C\}$, $\ell \in \{1, 2, 3\}$, and $m \in \{1, 2, 3\}$. Hence, Y_{A22} will represent the yield of Row 2 and Column 2 in the field where fertilizer A was used. In `R`, such an experiment would yield a data frame with nine rows:

Fertilizer	Row	Column	Yield
A	1	1	Y_{A11}
B	1	2	Y_{B12}
C	1	3	Y_{C13}
C	2	1	Y_{C21}
A	2	2	Y_{A22}
B	2	3	Y_{B23}
B	3	1	Y_{B31}
C	3	2	Y_{C32}
A	3	3	Y_{A33}

In this data frame the `Fertilizer`, `Row`, and `Column` columns are factors and the `Yield` column is numeric. If the name of the data frame is `x` then such a design is easy to analyze in `R` using the `lm` function using the command

```
fit <- lm(Yield ~ Fertilizer + Row + Column, data=x)
```

and the command `anova(fit)` would produce an analysis of variance table, where the p-value for the test of the significance of `Fertilizer` is of importance.

The data from such an experiment are assumed to follow the model

$$Y_{k\ell m} = \mu + \alpha_k + \beta_\ell + \gamma_m + \varepsilon_{k\ell m}, \tag{1}$$

where μ is the overall mean yield (or effect), α_k is the fertilizer effect, β_ℓ is the row effect, γ_m is the column effect, $\varepsilon_{k\ell m}$ is a random error, $k = A, B, C$, $\ell = 1, 2, 3$, and $m = 1, 2, 3$. This model can be equivalently written in matrix form as

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\varepsilon}, \tag{2}$$

where

$$\mathbf{Y} = \begin{bmatrix} Y_{A11} \\ Y_{B12} \\ Y_{C13} \\ Y_{C21} \\ Y_{A22} \\ Y_{B23} \\ Y_{B31} \\ Y_{C32} \\ Y_{A33} \end{bmatrix},$$

$$\mathbf{X} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{A11} \\ \varepsilon_{B12} \\ \varepsilon_{C13} \\ \varepsilon_{C21} \\ \varepsilon_{A22} \\ \varepsilon_{B23} \\ \varepsilon_{B31} \\ \varepsilon_{C32} \\ \varepsilon_{A33} \end{bmatrix},$$

and

$$\boldsymbol{\theta} = \begin{bmatrix} \mu \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta_1 \\ \beta_2 \\ \beta_3 \\ \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix}.$$

Usually $\boldsymbol{\varepsilon} \overset{\text{iid}}{\sim} N(0, \sigma^2)$, but Professor Cox is interested in what happens when the errors do not follow a normal distribution. This is a difficult analytical task, and therefore empirical evidence via computer simulations are used. This project consists of the following three tasks:

1. Professor Cox would like you to write a function called `simCorn` which will simulate a data frame of observed yields for the situation described above. The arguments for this function should be

   `overallEffect` which is the specified overall mean yield with a default value equal to 0. This is the value of μ in the model above, and is a numeric vector with length 1.

   `fertilizerEffect` which is the specified fertilizer effect with a default value of 0. This is a numeric vector of length three that contains the values α_1, α_2, and α_3, with a default value equal to `c(0,0,0)`.

   `rowEffect` which is the specified row effect. This is a numeric vector of length three that contains the values β_1, β_2, and β_3, with a default value equal to `c(0,0,0)`.

   `colEffect` which is the specified column effect with a default value equal to 0. This is a numeric vector of length three that contains the values γ_1, γ_2, and γ_3, with a default value equal to `c(0,0,0)`.

   `seed` which is an option random seed with a default value equal to `NULL`.

   `dist` should be a function name that generates a random sample from a distribution in `R`. The default function name should be `rnorm`.

   `...` additional arguments used for specifying the distribution of the error terms in the model for `R`. Use proper notation for `Python` also.

   Important Details:

   - The value returned by the function should be a data frame that contains the simulated data. The column names in the data frame should match those given above (`Fertilizer`, `Row`, `Column`, and `Yield`). The `Fertilizer`, `Row`, and `Column` columns should be factors while `Yield` should be numeric.
   - If `seed` is `NULL` then the seed of the random number generator should not be set. If `seed` is equal to an integer then the `set.seed` function should be executed using the specified seed. If `seed` is neither an integer or `NULL` then an error should be returned.
   - `dist` is the name of an `R` function that simulates random samples. You may assume that the first argument of this function is the sample size, and that all of the remaining arguments of the function will be supplied by the user though the special argument `...`.
   - The function should do error checking to insure that the user supplied arguments are of the correct type and shape. Errors should be reported to the user with detailed error messages and should stop your function from executing. Complete the following examples to show some of the errors that should be included work. The part that should result in an error is notated in blue.

Error Testing	μ	$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	$\boldsymbol{\gamma}$	Error Distribution	Seed
1	10	$(0,0,0,1)$	$(0,0,0)$	$(0,0,0)$	N(0,1)	42544
2	(10,0)	$(1,2,3)$	$(0,0,1)$	$(0,0,1)$	N(0,1)	7524
3	10	$(1,2,3)$	$(3,1,0,1)$	$(0,1,1)$	N(0,1)	75
4	10	$(1,2,3)$	$(0,1,0)$	$(0,0)$	N(0,1)	NULL

   - The data can be simulated using either Equation (1) or Equation (2), whichever you are more comfortable with.

   Below are some example uses of the function along with the intended actions. This should give you a better idea on how the arguments are to work.

   Example 1 `y <- simCorn()` would execute the function with all of the default values. hence `overallEffect=0`, `fertilizerEffect=c(0,0,0)`, `rowEffect=c(0,0,0)`, and `colEffect=c(0,0,0)`. Further, the `set.seed` function is *not* executed, and the random error vector is generated using the command `rnorm`.

Example 2 `y <- simCorn(overallEffect=10,seed=2123,dist=rgamma,shape=2)` would execute the function with `overallEffect=10`, `fertilizerEffect=c(0,0,0)`, `rowEffect=c(0,0,0)`, and `colEffect=c(0,0,0)`. Further, the `set.seed` will be executed using the seed `2123`, (that is, the command `set.seed(2123)` is executed) and the random error vector is generated using the command `rgamma` with argument `shape=2`.

Example 3 The `R` code

```
mu <- 7
alpha <- c(1,2,3)
beta <- c(2,2,1)
gamma <- c(3,3,2)
y <- simCorn(overallEffect=mu, fertilizerEffect=alpha, rowEffect=beta, colEffect=gamma,
       seed=29429, rnorm, mean=3, sd=2)
```

would execute the function with `overallEffect=7`, `fertilizerEffect=c(1,2,3)`, `rowEffect=c(2,2,1)`, and `colEffect=c(3,2,1)`. Further, the `set.seed` will be executed using the seed `29429`, and the random error vector is generated using the command `rnorm` with arguments `mean=3` and `sd=2`.

Put the code for this function in your project write-up, and use it to run the three examples given above. Both the code and the output from the examples must be included in your project write-up.

2. Write a help narrative for your function using the exact same format that is used for all `R` help narratives. That is, the help should include the sections description, usage, arguments, details, value and examples. Consult the help files in `R` to get an idea of what information is included. Remember, for this question you only NEED TO DO THIS FOR `R`, and not for `Python`. Your help file does not need to work in `R` (meaning help(simCorn) does NOT return the file you created). It is just a file you will submit with your project.

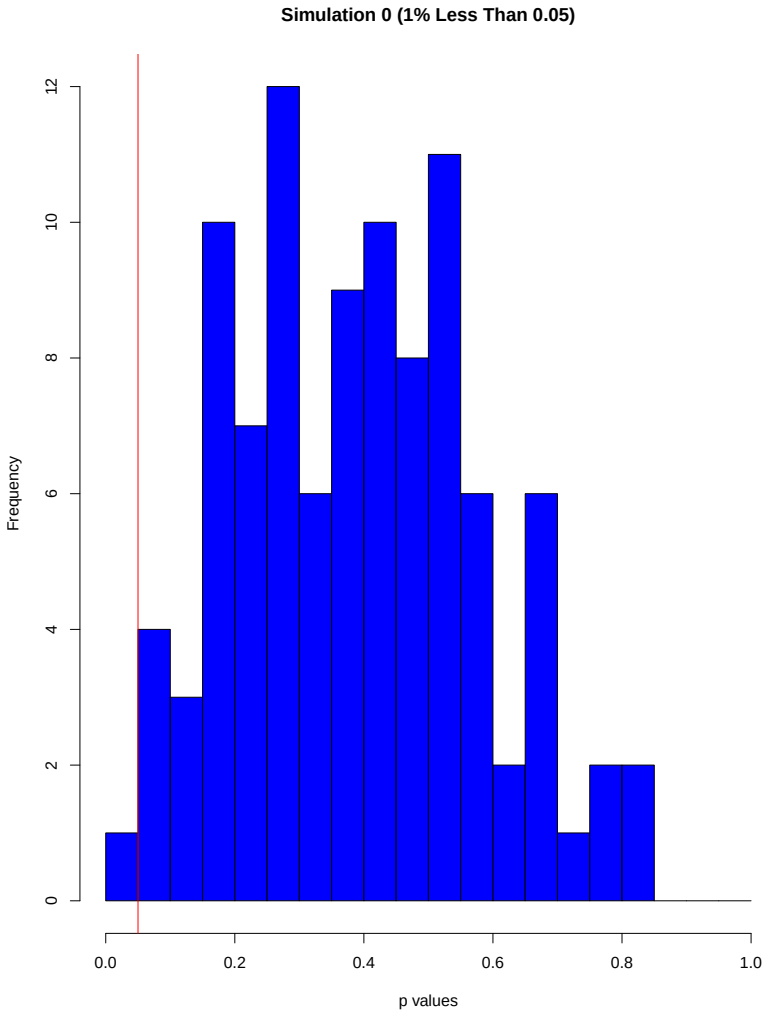
3. You will now use the function you wrote in Part 1 to run a small simulation. This simulation will explore the potential effect of changing the error distribution on the p-value of the test that determines if there is a significant difference between the effect of at least one of the three fertilizers. This p-value can be extracted from the `lm` object created by the `lm` function by using the `anova` function. For example, the `R`-code that generates some data, fits the model, and extracts the proper p-value is

```
y <- simCorn(overallEffect=10,dist=rgamma,shape=2)
fitCorn <- lm(Yield ~ Fertilizer + Row + Column, data=y)
pValue <- anova(fitCorn)$"Pr(>F)"[1]
```

Remember, example code is given in `R` here. In this case the error distribution corresponds to a $\text{GAMMA}(2,1)$ distribution. For the simulation Professor Cox would like you to simulate 100 data sets for each of the situations given in the table below.

Simulation	μ	$\boldsymbol{\alpha}$	$\boldsymbol{\beta}$	$\boldsymbol{\gamma}$	Error Distribution	Seed
1	10	$(0,0,0)$	$(0,0,0)$	$(0,0,0)$	$\text{N}(0,1)$	1331
2	10	$(1,2,3)$	$(0,0,1)$	$(0,0,1)$	$\text{N}(0,1)$	18694
3	10	$(1,2,3)$	$(1,0,1)$	$(0,1,1)$	$\text{N}(0,1)$	6516
4	10	$(1,2,3)$	$(0,1,0)$	$(0,1,0)$	$\text{N}(0,1)$	5
5	10	$(1,2,3)$	$(0,0,1)$	$(0,0,1)$	$\text{EXPONENTIAL}(1)$	574
6	10	$(1,2,3)$	$(1,0,1)$	$(0,1,1)$	$\text{EXPONENTIAL}(1)$	9476
7	10	$(1,2,3)$	$(0,1,0)$	$(0,1,0)$	$\text{EXPONENTIAL}(1)$	9743

The p-value for testing the effect of fertilizer will be computed for each simulated data set, and then a histogram will be created for the 100 p-values. Include the code needed to run these simulations and the histograms. You do not need to print out the simulated data or the individual p-values. The horizontal axis of the histograms should have a range from 0 to 1 and a vertical red line should be placed at 0.05. The bars of the histograms should be solid blue. The title of each histogram should be of the form `Simulation x (y% less than 0.05)` where x refers to the simulation number and y is the percentage of the p-values that are less than or equal to 0.05. The break points of the histogram cells should be $\{0.00, 0.05, 0.10, \ldots, 1.00\}$. An example histogram is given below (the resulting data may not look anything like your results, this is an example):



4. Did the resulting 7 simulations appear to be extremely different in `R` versus `Python`? Simulation should be compared based on the inputs being the same. Explain. Note, you are not explaining why this would be the case. Only if a difference in your simulations appears.

5. Now that you have completed the coding portion of this project, I would like you to reflect a little. Please answer the following questions regarding the final project. Note: Answers to these questions have no required length but a paragraph at minimum would be best. However, as long as you answered the question fully, you will receive full credit regardless of length.

   (a) What, if any, were the main differences you noticed in the `R` versus the `Python` coding? Did these differences make coding this problem easier, harder, or about the same?

   (b) Which language do you prefer and why?