Problem Set 3

Due February 14, 2023

Instructions

- Read all of these instructions closely.
- This problem set is due Tuesday, February 14, 2023 at 4pm.
- Submit files via Github:
  1. the .Rmd (R Markdown) file
  2. the knitted .pdf file
  3. anything else the particular problem set might require
- Use a copy of this file, perhaps with your name or initials appended to the file name, to write your answers to the questions. You'll see there is a designated space where your answers should begin.
- Knitting the .Rmd file to a .pdf file *as you work* will ensure your code runs without errors and is working how you expect. Knit early and often. You've already read the instruction that a knitted .pdf is required when you submit.
- Per the syllabus, I will not accept any late work. Keep in mind the two lowest problem set scores are dropped. Turn in what you have.
- Clarification on the expectations for problem set submissions (posted in Slack, copied here):
  - Always print the output of the code I'm requesting.
    - Ex: If I want you to create a vector x with elements 1 through 10, print x after creating it so I can see it worked.
  - Write any written answers in the space outside the code chunk, not inside with an R comment.
    - R comments are great to clarify code, but not for answering the question.
  - Make sure any code or written content is not cut off in the pdf.
    - This really should only apply to code, because if you follow item 2 in this list, the pdf will compile your written answers nicely.

Question 1–Loops and if/else (from Day03 inclass activity)

Write code to flip a coin ("H" and "T"), printing each flip to the console, until you get the same result three times in a row, at which point the coin flips should stop. This code can be written in many different ways, but I expect you to at least use one of the kinds of loops and the if/else commands we talked about during Day03 class.

Hints: Use the `sample` function for each coin flip.

```r
flip_old <- sample(c("H", "T"), 1)
repeat_count <- 0
repeat{
  print(flip_old)
  flip_new <- sample(c("H", "T"), 1)
  if(flip_old == flip_new){
    repeat_count <- repeat_count + 1
  }else{
    repeat_count <- 0
  }
  if(repeat_count == 3){
    break
  }
  flip_old <- flip_new
}
```

```
## [1] "H"
## [1] "T"
## [1] "T"
## [1] "T"
```

Question 2–Functions

In the game show Let's Make a Deal, participants choose one of three closed doors and receive the prize behind the door they choose. Behind one door is a new car; behind the other two doors are goats. After the participant selects one of the 3 doors, the host opens one of the other two doors, and reveals a goat. Now, the participant has the option of either sticking with the door they originally selected, or switching to the only other door that is still closed. What should the candidate do, and why? What are the probabilities of winning the car if they stay versus if they switch? This question is known as the Monty Hall Problem.

For this question, you will not answer the Monty Hall Problem. Instead, you'll code a slightly simplified version of the Let's Make a Deal game.

2a

Create a function that does the following:

- take one argument called `door` indicating the participant's chosen door (either 1, 2, or 3)
- check to make sure the `door` value given is appropriate for the function
  - if not, return a string from the function that tells the user what they did wrong
- draws a random number $[1, 2, 3]$ that presents the door behind which the car is hidden
- compares the user's guess with where the car is hidden
  - if the user is correct, return a string with a message congratulating the user that they were correct
  - if the user is wrong, return a strong saying they chose the wrong door

Also make sure to do the following to make sure your code is polished and readable:

- give the function an informative name
- give all objects created within the function informative names
- write succinct, informative comments in your code as needed to explain what is going on
- use best practices in styling the code as discussed in class so it is readable to someone else

```
montyhallproblem <- function(door){
  # the user's guess needs to be
  # a numeric and can only be 1,2,or 3
  if(!door %in% c(1,2,3)){
    # create message to return if problem
    user_msg <- "'door' must be an integer value of 1, 2, or 3."
  }else{
    # if no problem with input, code continues here
    rand_door <- sample(x = 1:3, size = 1) #draw random door
    if(rand_door == door){
      # if the random door and guess are equal, create message
      user_msg <- "Congratulations! You picked the correct door!"
    }else{
      # if the random door is not equal to guess, create message
      user_msg <- "Sorry, you picked the wrong door!"
    }
  }
  # return the specific message
  return(user_msg)
}
```

2b

Demonstrate your function works as expected by passing it an integer value and showing the output in the console. Also show your function works as expected by displaying your custom error message when the user supplies an unexpected input, such as the string `"Door 1"`.

```
montyhallproblem(door = 1)
```

```
## [1] "Sorry, you picked the wrong door!"
```

```
montyhallproblem(door = "Door 1")
```

```
## [1] "'door' must be an integer value of 1, 2, or 3."
```

2c

Write a for loop that calls your Monty Hall Problem function 1000 times with a guess of door 1 each time and stores the function output. Then, use the `table` function on the 1000 stored outputs to display how many times the user got the message that they won vs. lost. This simple simulation will help you know your function worked as expected, because the user should win approximately 1/3 of the time and lose approximately 2/3 of the time.

```
mh_msg <- rep(NA, 1000)
for(i in 1:1000){
  mh_msg[i] <- montyhallproblem(door = 1)
}
table(mh_msg)
```

```
## mh_msg
## Congratulations! You picked the correct door!
##                                           316
##             Sorry, you picked the wrong door!
##                                           684
```

Question 3–Final Project

I've made this problem set a bit shorter so you can spend some time exploring final project topics. This question asks you to not only have an idea, but start the process of flushing out the steps needed to execute the idea. What this looks like depends on the project. For example, if you want to do a data collection project, you'll have to starting looking around to see if anyone else has already done it. Or, if you're working with me on one of my projects, we'll have to have a meeting. The more effort you put in to this question to be as thorough as you can at this stage, the more I can understand and help!

You might ask – how long should my answers be? That is totally up to you. I strongly believe flushing out your ideas thoroughly in writing is never a waste of time.

3a

What is the project idea?

Answer:

3b

How would this project develop your skills as a programmer?

Answer:

3c

What is the broader motivation behind this project? I am only interested in the programming element when it comes to fulfilling the course objectives, but I am still interested in your broader research! Also, this will provide important context for me to help.

Answer:

3d

This there any other necessary context you haven't mentioned in 3a or 3b. For example: Would you like a certain job after graduating and these skills are necessary for it? Is this necessary work for your dissertation? Is this project going to be used for another course as well? (If so, you should consider checking with the other professor to make sure that is okay.). Is this an ongoing project you've been working on for some time? Is this work for another professor as an RA? Etc.

Answer:

3e

List the steps you imagine you'll need to take for this project from the most basic initial tasks to the end goal (that may or may not be reached in our one semester together).

The amount of effort a coding project might take is hard to predict. Some steps might be much quicker than you imagined. Perhaps someone else has already done the work for you! Or, some might be much more difficult than you imagined where overcoming a single unexpected roadblock turns into the entire course project. Try to set the unpredictability of your project aside for a moment, and do your best to list the steps you think you'll need to execute the project idea.

This is where you can start to flush out the idea and take some initial steps. This is the most important question of the problem set and where I want you to put in some effort this week beginning to explore how to execute your project. For example, for a scraping project, you should at least answer these questions: If there's a website you need to scrape, provide the link so I can look. What pages on the website? What parts of the pages are you most interested in? Have you looked to make sure no one has done this yet? When you configure the data, what will each observation (i.e., row) be – a country, a year, a person, a document? Etc.

(Sidenote: the asterisk and plus sign is one way to make a nice bulleted list in Rmarkdown. You can also create sub-bullets, see resources here.)

Answer:

- Step 1: ....
  - Perhaps you want to use
  - sub-bullets to
  - better organize your steps
- Step 2: ....
- Step 3: ....

3f

At this point, what steps of the project you think you can realistically accomplish for this class. (I will provide feedback on this too, and we will adjust expectations together as the semester continues.)

Answer: