Applied Statistical Programming - Spring 2022

Problem Set 3

Due Wednesday, March 2, 10:00 AM (Before Class)

Instructions

1. The following questions should each be answered within an R script. Be sure to provide many comments in the script to facilitate grading. Undocumented code will not be graded.
2. Work on git. Fork the repository found at https://github.com/johnsontr/AppliedStatisticalProgramming2022 and add your code for Problem Set 3, committing and pushing frequently. Use meaningful commit messages because these will affect your grade.
3. You may work in teams, but each student should develop their own Rmarkdown file. To be clear, there should be no copy and paste. Each keystroke in the assignment should be your own.
4. For students new to programming, this may take a while. Get started.

Let's Make a Deal[1]

In the game show "Let's Make a Deal' ', the candidate gets to choose one of three closed doors, and receives the prize behind the door they choose. Behind one door is a new car; behind the other two doors are goats. After the contestant selects one of the 3 doors, the host opens one of the other two doors, and reveals a goat. Now, the candidate 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.

Your tasks

For this problem set, you will not solve the Monty Hall Problem, but you will have to code a slightly simplified version of the "Let's Make a Deal" game. More specifically, you will set up a new class, which contains information regarding the door a player chooses, and a method that simulates a modified version of the game. You will have to do this using the S3 class system. Here are the specific instructions:

1. Define a new class: `door`. Objects of this class simply take on one numeric value: 1, 2, or 3 – indicating which door a candidate chooses.
2. Create a method for `door` objects that is called `PlayGame`. This method is supposed to do the following:
   - take the numeric value that is stored in the `door` object,
   - draw a random number between 1 and 3 that presents the door behind which the car is hidden,
   - compare the two numbers, and print a message congratulating a winning candidate that chose the correct door, or expressing sympathies for a losing candidate that chose the wrong door.
3. Write:
   - a construction function that allows the user to create a `door` object,
   - and a validation function that checks whether the value stored in `door` is actually an integer

[1]https://en.wikipedia.org/wiki/Let's_Make_a_Deal

```r
#Start by creating a constructor function for our doors.
newdoor <- function(door.number){
  stopifnot(is.numeric(door.number))#Forces them to construct with a number
  door <- list(number = door.number, #gives the class door the number attribute
               has.handle = TRUE, #doors have handles
               has.hinge = TRUE,#and hinges
               is.rectangular = TRUE, #Any door on this show is a rectangle
               blocks.dogs = TRUE) #and my dog Mio can't open them
  class(door) <- "door" #assigns the class
  return(door) #returns the door
}

#Construct each of the doors
door1 <- newdoor(1)
door2 <- newdoor(2)
door3 <- newdoor(3)

#Make a method for playgame
playgame <- function(door) {
  UseMethod("playgame")
}

#Specify the method for a door
playgame.door <- function(door){
  x = sample(1:3,1) #sample to decide the winning door
  if(x==door$number){"Congratulations! You have won a brand new car!"} #Winner message
  else{"Awww too bad! You chose the wrong door. Better luck next time!"} #Loser message
}

#Test playgame
playgame.door(door1)
```

```
## [1] "Awww too bad! You chose the wrong door. Better luck next time!"
```

```r
playgame.door(door2)
```

```
## [1] "Awww too bad! You chose the wrong door. Better luck next time!"
```

```r
playgame.door(door3)
```

```
## [1] "Awww too bad! You chose the wrong door. Better luck next time!"
```

```r
#Build a function to validate doors
validate_door <- function(door){
  #Check if it is a door
  if(door$has.handle==FALSE){
    stop("This isn't a door, where is the handle?")
  }
  #Are we sure it is a door?
  if(door$is.rectangular==FALSE){
    stop("It isn't even the right shape for a door!")
  }
  #Make absolutely sure it is a door
  if(class(door)!="door"){
    stop("It looks like a door and quacks like a door, but I'm pretty sure it isn't a door...")
  }
  #Make sure it has a number
  if(is.numeric(door$number)==FALSE){
    stop("That's the door to leave the studio...you're welcome to choose it, but I'd recommend somethin
  }
  #Validate if it has an integer
  if(door$number != as.integer(door$number)){
    stop("That's not a door! Doors have integers on them!")
    }
  #Validate if it is a door that can win
  if(door$number>3 | door$number<1){
    stop("Hey, you weren't supposed to find that door hidden backstage!")
  }
  return(door)
}

validate_door(door1)
```

```
## $number
## [1] 1
##
## $has.handle
## [1] TRUE
##
## $has.hinge
## [1] TRUE
##
## $is.rectangular
## [1] TRUE
##
## $blocks.dogs
## [1] TRUE
##
## attr(,"class")
## [1] "door"
```

```
#While I had a group of other tests, they fail so I had to comment them out
#doorpi <- newdoor(pi)
#validate_door(doorpi)
#door.exit <- newdoor("exit")
#validate_door(door.exit)
#validate_door(pi)
```