

Applied Statistical Programming - Spring 2022

Annie Jarman

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 Deal¹

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.

```
# 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  
chosenDoor <- list(choice = sample(1:3, 1))  
class(chosenDoor) <- "door"
```

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

```
# check class
class(chosenDoor)
```

```
## [1] "door"
```

```
# check value is a single number
chosenDoor
```

```
## $choice
## [1] 2
##
## attr(,"class")
## [1] "door"
```

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.

```
# Advanced R would call this the low-level constructor
```

```
# first generic
```

```
PlayGame <- function(chosenDoor) {
  UseMethod("PlayGame")
}
```

```
# take the numeric value that is stored in the door object
```

```
PlayGame <- function(chosenDoor) {
  # draw a random number between 1 and 3 that presents the door behind which
  # the car is hidden
  car <- sample(1:3, 1)
  # compare the two numbers
  if (chosenDoor == car) {
    # print a message congratulating a winning candidate that chose the
    # correct door
    print("Huzzah! You did it! Folks, you're looking at the owner of a brand new car!")
  } else {
    # express sympathies for a losing candidate that chose the wrong door
    print("Ah shucks! So close!")
  }
}
```

```
# test it - I reran the code until I had successfully received both messages
```

```
PlayGame(chosenDoor)
```

```
## [1] "Ah shucks! So close!"
```

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

```
# write a construction function that allows the user to create a door object
# Advanced R would call this the helper, recommends calling it new_myclass()
new_chosenDoor <- function(choice) {
  output <- list(chosenDoor = choice)
  class(output) <- "door"
  return(output)
}

# test helper
door1 <- new_chosenDoor(1)
PlayGame(door1)

## [1] "Ah shucks! So close!"

# validation function that checks whether the value stored in door is actually
# an integer Advanced R would call this the validator
validateDoor <- function(choice) {
  `!%notin%` <- Negate(`%in%`)
  # must be an integer
  if (!is.numeric(choice)) {
    stop("Not an integer")
  }
  if (choice %notin% c(1:3)) {
    stop("Not a valid door, choose 1, 2, or 3 ")
  }
}

# test validator validateDoor(choice=6) #doesn't work
# validateDoor(choice='meow') #doesn't work
validateDoor(choice = 1) #works
validateDoor(choice = 2) #works
validateDoor(choice = 3) #works
```