Lab 2

YOUR NAME HERE

2023-08-30

You should edit this .Rmd using RStudio, then click *Render* in the menu bar of the Source window (above the text of this .Rmd). Remember, you must submit *both* your .Rmd and the compiled .html in order to receive full credit!

Collaborators

INSERT NAMES OF ANY COLLABORATORS

A. Functions and Vectors

In this problem, we will revisit the `sample` function to simulate the birthday problem. The birthday problem asks for the probability that in a set of n randomly chosen people, at least two will share a birthday. For the purpose of this problem, suppose that we live in a universe where there are exactly 365 days in a year (sorry to those of you with leap day birthdays!) and each individual has an equal probability of being born on any given day.

1. Write code to create a vector `birthdays` of the numbers 1:365.

```r
# Make a vector containing 1 to 365 numbers
birthday <- 1:365
```

2. Use the `sample()` function to sample $n = 5$ birthdays (with replacement) from `birthdays` and save it as a vector `sample_5`. Repeat this with $n = 10, 20, 30, 50, 100$. You may want to use `set.seed()` to set a random seed.

```r
# Set seed to have reproducible results.
# Make 6 vectors containing random samples of the birthday vector, the vectors' sizes are 5, 10, 20, 30

set.seed(42)

sample_5 <- sample(birthday, size = 5, replace = TRUE)
sample_10 <- sample(birthday, size = 10, replace = TRUE)
sample_20 <- sample(birthday, size = 20, replace = TRUE)
sample_30 <- sample(birthday, size = 30, replace = TRUE)
sample_50 <- sample(birthday, size = 50, replace = TRUE)
sample_100 <- sample(birthday, size = 100, replace = TRUE)


sample_5
```

```
```

```
## [1]  49 321 153  74 228
```

```r
sample_10
```

```
##  [1] 146 122  49 128 303  24 327 356  89 165
```

```r
sample_20
```

```
##  [1] 110  20 297  89 283 109   5 212 348 360 259 314 298  24 158 299 314 136 292
## [20] 324
```

```r
sample_30
```

```
##  [1] 146 109 348 197   4 226 355 215 245 114 262 130   3 258 358 186 138  40   5
## [20]  33 103 228 109 329 157  76 265  35 221  16
```

```r
sample_50
```

```
##  [1] 357 220 248 325 118 130  82 325 110 360 296 149  57 100 298  91 269 181  54
## [20] 339 288 208 246  60 285 337 108 341 126 112  72 285   1 141 206 311 299  42
## [39]  42 248 353 251 157  25 191  32 337 238  14 111
```

```r
sample_100
```

```
##   [1] 262 299 208 246 242 287 350 224 262 214 294  95   6 340 252 271  34 188
##  [19] 254 268 128  41 193  66 312 152  98  25 348 107 317  14 162 124 194  95
##  [37]  32 185 284 118  37 138 261  35  78  14 156  97 229 182 127 287  43 180
##  [55] 337  27  30 180 356 212  75 201  62 269 124  63  49  61 116 257  28   2
##  [73] 159 136 127 208 259 229 269  12 193 158  51 351 165  47 312 198  10 255
##  [91] 201 233   3 338 181 239 284 252  93 313
```

3. Did you sample the same birthday twice in `sample_5`? What about for the other samples? Write code that can be used to test whether any of the days appears twice in each of your samples. There are many

ways to do this and you can use any functions. In particular, `length()` and `unique()` may be useful.

```r
# creating vectors of duplicate days for each sample sets.
repeat_5 <- unique(sample_5[duplicated(sample_5)])
repeat_10 <- unique(sample_10[duplicated(sample_10)])
repeat_20 <- unique(sample_20[duplicated(sample_20)])
repeat_30 <- unique(sample_30[duplicated(sample_30)])
repeat_50 <- unique(sample_50[duplicated(sample_50)])
repeat_100 <- unique(sample_100[duplicated(sample_100)])

repeat_5
```

```
## integer(0)
```

```r
repeat_10
```

```
## integer(0)
```

```r
repeat_20
```

```
## [1] 314
```

```r
repeat_30
```

```
## [1] 109
```

```r
repeat_50
```

```
## [1] 325 285  42 248 337
```

```r
repeat_100
```

```
##  [1] 262  95  14 287 180 124 127 208 229 269 193 312 201 284 252
```

4. What do you observe? Discuss any patterns or surprising findings.

For the smaller sets, sample_5 and sample_10, there were no two birthdays that were the same. It is observed that there is a higher likelihood of a day appearing more than once in a larger sized sample sets.

B. Working with Data Frames

Use the following code to load the `penguins` data.

```r
# load palmer penguins package
library(palmerpenguins)

# open penguins data as a data frame
data(penguins)
penguins <- as.data.frame(penguins)
```

5. Using the `mean()` function, compute the mean body mass of penguins in the dataset, dropping any missing values.

```r
# omitting any rows in the body mass column that have NA value before taking the mean of the body mass(g
clean_mass <- na.omit(penguins$body_mass_g)
mean_mass <- mean(clean_mass)
```

The average body mass of the penguins is 4201.754386g.

6. Using the `max` function, compute the maximum flipper length of penguins in the dataset, dropping any missing values.

```r
# omitting any rows in the flipper length column before finding the max flipper length(mm)
flipper_clean <- na.omit(penguins$flipper_length_mm)
max_length <- max(flipper_clean)
```

The maximum flipper length is 231mm.

7. Using the `hist` function, create a histogram of the ratio of the penguins' bill length to bill depth, dropping any missing values. What is the shape of the resulting distribution?

```r
# remove all of the NA values from both the bill length and bill depth columns
# then find the ratio of the penguins' bill length to bill depth by dividing the length by the depth
# plot the ratio using the histogram.

L_clean <- na.omit(penguins$bill_length_mm)
D_clean <- na.omit(penguins$bill_depth_mm)

ratio_ld <- L_clean/D_clean

hist(ratio_ld, main = "Ratio of bill length to bill depth")
```