Max Hammond, Problem Set 2

Due February 7, 2023

I turned evaluation off when knitting because my code after dropping the variables would show the entire dataframe in the answer

Instructions

- Read all of these instructions closely.
- This problem set is due Tuesday, February 7, 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.

Overview

This problem set uses a subset of expenditures data for all campaigns and PACs available from Open Secrets for 2002 cycle. The reduced dataset is available here. (While not the point of this question, I encourage you to visit the link to see how data shared on Dropbox can be imported directly into R via its url.)

Before you being this question, you should familiarize yourself with the variables. The codebook is available here.

```
expenditures_url <- "https://www.dropbox.com/s/z6gw9lvve6jogi5/Expends2002.txt?raw=1"
df <- read.csv(expenditures_url)
```

Question 1–Working with logicals

Use R code to answer the following questions.

1a

Are any `Amount` values missing? No

```
sum(is.na (df$Amount))
```

```
## [1] 0
```

1b

How many observations are for refunds? 276

Hint: Read the codebook carefully for the `Amount` variable.

```
sum((df$Amount) < 0)
```

```
## [1] 276
```

1c

What are the row indices for observations that indicate an amount spent of $1,000,000 or more?

[1] 9169 14586 14868 14886 17290 17367

```
which (df$Amount >= 1000000)
```

```
## [1]  9169 14586 14868 14886 17290 17367
```

1d

Double check that all of the `Cycle` values equal 2002.

All 20,000 observations have the same year.

```
table(df$Cycle)
```

```
##
##  2002
## 20000
```

1e

How many observations are for “Club for Growth” OR the “Madison Project” OR the “Republican National Cmte”?

[1] 1337

```
sum(df$Pacshort %in% c("Club for Growth" , "Republican National Cmte" , "Madison Project"))
```

```
## [1] 1337
```

Question 2–Working with dataframes

2a

Using R functions, describe the following properties of the `df` object: class, dimensions, columnnames, rownames*, and anything else you think is pertinent.

*rownames(df) will list too many observations so I chose to omit this from the assignment.

```
dim(df)
```

```
## [1] 20000    21
```

```
colnames(df)
```

```
##  [1] "Cycle"         "ID"            "TransID"       "CRPFilerid"    "Recipcode"
##  [6] "Pacshort"      "CRPRecipname" "Expcode"       "Amount"        "Date"
## [11] "City"          "State"         "Zip"           "CmtelD_EF"     "Candid"
## [16] "Type"          "Descrip"       "PG"            "ElecOther"     "EntType"
## [21] "Source"
```

```
class(df)
```

```
## [1] "data.frame"
```

2b

For the `TransID` variable, change its column name to `Useless_Var`.

Bonus: If you want to challenge yourself, try to write code that is flexible, meaning it will work correctly if `TransID` is the 3rd variable, 20th variable, or any position in the dataframe.

```
colnames(df)[which(names(df) == 'TransID')] <- 'Useless_Var'
```

2c

Remove the variables `Useless_Var` and `Source` from the dataframe.

Bonus: Make this code flexible as well.

```
drops <- c("Useless_Var","Source")
df = df[ , !(names(df) %in% drops)]
```

2d

The variable `State` has many obvious errors. I've created the variable `StateWrong` with `NA` placeholders. Recode `StateWrong` to be `TRUE` if the `State` variable contains an error or a missing value, and `FALSE` otherwise.

Hint: We did a recoding exercise in the inclass activity.

Bonus: Try to use the `%in%` function. We haven't used it in class yet. It is similar to `==`. The syntax is `x %in% y`, which assesses each value of vector `x` and asks, is it equal to any of the values in vector `y`? I've included a simple example below.

```
df$StateWrong <- NA
x <- c("  ", "LL","St","VI","ZZ")
df$StateWrong <- df$State %in% x
```

2e

Using the `StateWrong` variable, report how many observations in the dataset have a wrong or missing value. Then remove these observations. Confirm that you've removed the correct number of rows by checking the dimensions of the data.

There were 92 missing values or errors in the dataset.There are now 19908 observations in the dataset.

```
sum(df$StateWrong == TRUE)
```

```
## [1] 92
```

```
df <- df[!df$StateWrong, ]
```

2f

Create the variable in the dataframe called `Payroll`. It should be a logical indicating whether the `Descrip` variable contains the string "payroll" *regardless* of capitalization.

Report the number of `TRUE` values in this variable. [1] 158

Hint: Use the `grepl` function and read the helpfile closely.

```
z <- c("payroll", "Payroll", "PAYROLL")
df$Payroll <- df$Descrip %in% z
sum(df$Payroll == TRUE)
```

```
## [1] 158
```

2g

Write a function named `sum_state_exp` that takes one character argument called `state_code`. The function should return the total amount of expenditures in given state.

[1] 478163 [1] 1165688 [1] 1994622

```
sum_state_exp <- function(state_code) {
  state_exp <- df[df$State== state_code, ]
  return(sum(state_exp$Amount))
}
# After writing the function, run it for IA, IL, and CA
sum_state_exp(state_code = "IA")
```

```
## [1] 478163
```

```
sum_state_exp(state_code = "IL")
```

```
## [1] 1165688
```

```
sum_state_exp(state_code = "CA")
```

```
## [1] 1994622
```