christmas\_songs <- read.csv("https://raw.githubusercontent.com/rfordatascience/tidytuesday/master/data/2019/2019-12-24/christmas\_songs.csv")

mean(christmas\_songs$peak\_position) #this is a mean function to get the mean peak position of songs on the list

plot(x=christmas\_songs$peak\_position, y=christmas\_songs$weeks\_on\_chart) #using plot function to make a scatter plot

abline(lm(christmas\_songs$weeks\_on\_chart ~ christmas\_songs$peak\_position), col = "blue") # abline adds a line of regression to out plot and colors its blue. how can we chnage the color

#1: intro 6+7

6+7

9+1

365+1986

#2 - Math

#a. 3-4

3-4

#b. 7 divided by 10

7/10

#c. 8 raised tot he 7th power

8^7

#d. the square root of 52

sqrt(52)

#3 - Assigning variables

x <- 42

#a. How would you create a name y and assign it the value 334

y <- 334

#b. how would you add the x and y together to get the sum of the two numbers?

x+y

#c. how would you create a new variable (z) that stores the result of x +y

z <- x+y

#d. now, change the value of x to be 500 and see what happens when you add x + y together? if you ask for the value of z now what do you get? why?

x <- 500

z

# i think this is because we have to do z <- x+y again, so that is what i am going to try

z <- x+y

z

#yup, e. I am guessing the answer is that R is dumb, and you make it smart, by

#making sure you do everything consequently as it will still remember values

#less you go back and change it and set it again so it can get that sum of the

#new number

#4 types of data

x <- 42

class(x)

xx <- 42L

is.integer(xx)

my\_value <- 1+3

is.numeric(my\_value)

my\_name <- "Edgar Villeda"

is.numeric(my\_name)

is.character(my\_name)

#this is because it is using my name, which is a character so it is true, if I

#were to give it a number then it would of said FALSE

a <- 1.333

b <- TRUE

c <- "my name is"

d <- Sys.Date() # tricky

#PART 5

my\_vector <- c(1,2,3,4,5)

my\_new\_vector <- c(6,7,8,9,10)

my\_study <- c("male","male","female","male","female","female","female")

my\_study[2]

#b. how would you get the sex of the third skeleton

my\_study[3]

#c. can you figure out how, in one line of code, to get the sex of the 1st and

#4th skeleton?

my\_study[1][4]

#idk

names(my\_study) <- c("one", "two", "three", "four", "five", "six", "seven")

my\_study[2]

my\_study["two"]

my\_study[2] == my\_study["two"]

#6 matrices

cx1980 <- c(7, 13, 8, 13, 5, 35, 9)

cx1988 <- c(9, 11, 15, 8, 9, 38, 0)

chimp <- cbind(cx1980, cx1980) #cbind binds the vector together a columns

class(chimp)

chimp[1,] #note the comma

chimp[,1]

chimp[,2]

chimp[2,3]

chimp[3,1]

chimp[3,2]

#i dont know why is giving me 8 instead of 13 maybe you can help me out with that.

cx2002 <- c(1,3,5,7,9)

cx2003 <- c(2,4,6,8,10)

lemur <- rbind(cx2002, cx2003)

class(lemur)

lemur[1,]

lemur[1,2]

#i would say using rbind() but i a not sure if im doing it right

#B

cx1980 <- c(7, 13, 8, 13, 5, 35, 9)

cx1988 <- c(9, 11, 15, 8, 9, 38, 0)

chimp <- cbind(cx1980, cx1988) # cbind binds the vectors together a columns

class(chimp)

chimp

freqmat

pop. <- c(34,22,10,0,2, 0,3,5,1,6, 22,40,25,10,8, 12,11,3,40,20)

sp. <- c("V. varieagata", "D. madagascariensis", "I. indri", "P. diadema")

Census <- c("Ranomafana", "Betampona", "Nosy Be", "Kirindy forest")

tote <- matrix(pop., nr=4, nc=4, byrow=TRUE)

dimnames(tote)[[1]] <- sp.

dimnames(tote)[[2]] <- Census

tote

bone <- c("humerus", "radius", "ulna", "femur", "tibia", "fibula")

size\_inches <- c(14.4, 10.4, 11.1, 19.9, 16.9, 15.9)

injury <- sample(c("yes","no"),6,replace=TRUE)

sample\_letter <- LETTERS[1:6]

my\_sample <- data.frame(bone, size\_inches, injury, sample\_letter)

my\_sample

#if you can let me know how does it know which ones are yes or no

# for the injury?

View(my\_sample)

my\_sample$bone

num <- c(1,2,3,4,5,6)

food <- c("bread", "butter", "milk", "cheese","coffee", "tea")

quantity <- c(1,1,3,5,7,1)

shopping <- data.frame(num, food, quantity)

shopping

#b. what is the class type of the different vectors in the my\_sample dataframe?

#its a dataframe.

#Part 8: comparing values

#a. is a bigger than b?

#b. is c equal to d (careful with this one..)

#c. is c less than or equal to b?

# d. make a new vector called temp with the values of 1,5,7,9,11,14,6,8. then write a single line of code that evaluates if 3 is greater than each of the values in the vector

#e. how would you ask R if the 5th value in temp is larger than 5?

a <- 5

b <- 9

c <- 7

d <- sqrt(49)

a > b

c == d

c <= b

temp <- c(1,5,7,9,11,14,6,8)

3 > temp

3 > temp[5]

my\_data <- read.csv(file.choose(), header = T)

my\_data

View(my\_data)

#Part 10: how did you do

#1. What the most challenging part of this homework?

#not anything hard persae, I would say more tedious, some of the more complex probelms just need a bit more analyzing.

# 2. What could be more clear? What info might be helpful to include?

#Honestly this was perfect, I really appreciated this, maybe going into more complex methods?

#3. After sitting with this, do you think you have a better idea of what R is all about?

#I mean yes, especially the basics

#4. Probably the most awful thing about R are the classes. If you had to explain what a class was how would you do that?

#Honestly, this right here. teach them the basics and have others follow me along either on paper or on a zoom video.

#5. Now that we have an idea of how R thinks, next week we are gonna talk a bit about how to store data. Based on what you now know, what might be good practices for recording data on a spreadsheet so that others can use them?

#Make sure that they are easily accessible, and the information is all uniform so we do not have to mess with the data on R too much.