Exam 1 Study Guide by Itzel Gonzalez			
LAB 2			
Function	**Description**	**Example**	**Example**
mean(c())	calculates the average mean of a set of numbers		
class()	returns the type of the variable. Ex. Integer, Character, Boolean		
is.integer	checks if variable is an integer		
as.integer	converts variable to an integer		
as.numeric()	checks if variable is numeric		
is.numeric()	coverts variable into a numeric variable		
is.na()	checks if dataframe contains any NA		
anyNA()	checks if there is at least one NA (TRUE/FALSE)		
c(1,2,3,4)	numerical vector		
c('cat', 'dog', 'bird')	character vector		
c(1:100)	creates a sequence from 1 to 100		
[ ]	pulls out elements in a vector	days_of_the_week[3]	my_vector_sequence[10]
<	less than		
>	greater than		
==	equal too		
>=	greater than or equal to		
<=	less than or equal to		
names()	lists column names		
dim()	dimensions of a data frame		
str()	tells you the structure		
data.frame()	creates a table with values or strings		
tibble()	creates a table with values or strings		
=	assigns a value or can be used to rename a variable	mean(hbirds$length)	
$	access a column in a data frame		
write.csv()	allows us to save a data frame	write.csv(hbirds, "hbirds_data.csv", row.names = FALSE)	

LAB 3	**Description**	**Example**	**Example**
Function			
read_csv()	reads a csv file into R as a dataset		
view()	opens the dataset in a table format		
summary()	summarize data frame		
glimpse()	another useful way to summarize, tells you what class each vairable belongs to		
nrow()	shows number of rows		
ncol()	shows number of columns		
head()	reads the first n rows of the data frames		
tail()	reads the last n rows of the data frams		
table()	gives the total number of observations for each factors/variables		
as.factor()	coverts a column into a factor (categorical variable)	input: animals <- c("cat", "dog", "bird", "cat", "dog") animal_factor <- as.factor(animals) print(animal_factor)	[1] cat dog bird cat dog output: Levels: bird cat dog
levels()	lists the unique categories in a factor column		
getwd()	shows current working directory		
select()	pulls put variables(columns)		
filter()	pulls out observations		

LAB 4	Description	Example	Example
Function			
library(tidyverse)	loads multiple data manipulation packages		
library(janitor)	provides extra dating cleaning functions		
clean_names()	cleanes names in files and ensures consitency across variable names	select(fish,fish_id:length)	
select()	pulls put variables(columns)	select(fish, -"fish_id", -"annnumber")	
select(dataframe, colname:colname)	adds a range of columns	gives me everything that contains the strings	select(fish, contains('length')
select(dataframe, -"colname", ...)	select everthing except (-)		select(fish, starts_with('radii')
select(dataframe, contains())	gives names that contain strings for a certain variable		
select(dataframe, starts_with())	will put out variables that start with ..		
select(dataframe, ends_with)	select columns that end with a string character		
select(dataframe, matches())	select columns that match a refular expression		
select(dataframe, one_of())	select columns names that are froma group of names		
select_if(dataframe, is.numeric)	selects only numeric columns		
select_if(dataframe, ~!is.numeric(.))	selects only non-numeric columns		
filter(dataframe, colname != '')	removes observations within a specific column name		
filter(dataframe, near(colname, #, tol = x))	selects observations where columnname is approxamely equal to a number with a tolerance of x		
filter(dataframe, between(colname, #, #))	selects observatiosn where columnname is between two values		
filter(dataframe, colname %in% c(x,y))	selects observations where columnname matches one of the given values		
LAB 5	**Description**	**Example**	
Function			
filter(dataframe, col name & col name)	do two things at once	filter(fish, lakedid == 'AL' & length>350)	
filter(dataframe, col name | col name)	equivilant to "or", gives you observations that are colname or colname	filter(fish, lakedid == 'AL' | length>350)	
filter(dataframe, condition1, !condition2)	will return all rows where condition one is true bund condition 2 is not		
filter(xor(condition1, condition2)	will return all rows where only one of the conditions is met, and not when not when both conditions are met		
arrange(-colname) or arrange(desc(colname))	specifies the variable to be sorted by descending order		
arrange(colname)	specifies the variable to be sorted by ascending order		
mutate(new_colname=colname)	create a new column from existing columns in a data frame		

LAB 6	Description	Example	
Function			
mutate(across(everything( ), tolower))	mtate across all the varaibales and observations and change them to lo	rder", "family"), tolower))	
mutate(across(-#, ~ tolower(.)))	apply this function to every cell (.) except the first column (-1)		
mutate(new_column = ifelse(condition, value_if_true, value_if_false))	if new_column is equal to condition replace with value_if_true, rename with new_column with the following info after ifelse	mutate(newborn_new = ifelse(newborn == -999.00, NA, newborn)	
summarize( )	will produce summary statisitcs for a given variable in a dataframe	summarize(mean_sleep_lg=mean(sleep_total))	summarize(mean_sleep_lg=mean(sleep_total), min_sleep_lg=min(sleep_total), max_sleep_lg=max(sleep_total), sd_sleep_lg=sd(sleep_total),
total=n()	total number of observations		
summarize(across(everything(), mean, na.rm=TRUE))	summarizes the mean across all numberic and observation variables		
n_distinct()	count the number of unique (distinct) values in a column	summarize(n_genera=n_distinct(genus))	input: data <- c(1, 2, 2, 3, 4, 4, 4)
unique()	list with duplicate values removed	unique(bodyweight_small$genus)	input: data <- c(1, 2, 2, 3, 4, 4, 4)
distinct()	removes duplicate observations	name = c("Alice", "Bob", "Alice", "Charlie"),	output: Alice, Bob, Charlie
pull()	take out one column from table and turn it into list		
count()	count the number of unique observations in each variable	msleep %>% count(vore)	
tabyl()	makes a quick summary table of how often values appear		
count(observation, sort = T)	sorts by descending order by count		

LAB 7	Description	Example		
Function				
group_by()	group data by a categorical variable(s) of interest	group_by(species)	group_by(species, island)	
summarize(n=n(), .groups= 'keep')	counts the numbers of observaations in each group, and keeps the grouping structure after summarizing			
na.rm=T	removes NA (missing) values when performin operations like mean,sum,min, max	summarize(mean_bill_length_mm=mean(bill_length_mm, na.rm=T)		
across()	allows to filter and select across multiple variables	summarize(across(c(species, island, sex), n_distinct))	summarize(across(c(species, island, sex), \(x) n_distinct(x, na.rm = TRUE))) ** x represents each column	