SOLUTION: Assignment - logical operators and dataframes

# Run the following code to generate the data for this assignment. (You don't need to understand this code in order to successfully complete this assignment).

# The questions in this assignment refer to a dataframe that contains information about stocks.
# You can use the following code to generate some random data about ficticious stocks which you can use as
# the data for the assignment. Note that the actual values in the data will should affect the code that you
# write for your answers. However, it does help to have some data on which to test out your answers.

generateRandomStockData <- function (numberOfStocks=2000, seed = 1) {

 set.seed(seed) # this will "prime" the random number generator

 # Generate the tickers

 getTicker <- function(...) {

 # Generate tickers

 numberOfLetters <- sample(2:4, size=1, prob=c(.15, .2, .65) )

 tickerLetters <- sample(letters, size=numberOfLetters, replace=TRUE)

 paste0(tickerLetters, collapse="")

 }

 uniqueTickers <- vector()

 while (length(uniqueTickers) < numberOfStocks){

 #cat("length=",length(uniqueTickers))

 tickers <- c(uniqueTickers, sapply(1:numberOfStocks, getTicker))

 uniqueTickers <- unique(tickers)

 uniqueTickers <- uniqueTickers[1:min(length(uniqueTickers),numberOfStocks)]

 }

 # Generate the exchange

 exchanges <- sample(c("nasdaq","nyse"), size=numberOfStocks, replace=TRUE, prob=c(.6, .4) )

 # Generate the sector

 sectors <- sample(c("tech","energy","pharm","consumer"), size=numberOfStocks, replace=TRUE, prob=c(.3, .3, .2, .2) )

 # Generate the open prices

 numberOfCheapStocks <- floor(.15 \* numberOfStocks)

 numberOf100Stocks <- floor(.40 \* numberOfStocks)

 numberOf1000Stocks <- floor(.18 \* numberOfStocks)

 numberOfRemainingStocks <- numberOfStocks - numberOfCheapStocks -

 numberOf100Stocks - numberOf1000Stocks

 openPrices <- sample(095:105, numberOfCheapStocks, replace=TRUE) / 100

 openPrices <- c(openPrices, sample(9800:10200, numberOf100Stocks, replace=TRUE) / 100)

 openPrices <- c(openPrices, sample(99000:101000, numberOf1000Stocks, replace=TRUE) / 100)

 openPrices <- c(openPrices, sample(1:200000, numberOfRemainingStocks, replace=TRUE) / 100)

 news <- sample(c(TRUE, FALSE), numberOfStocks, replace=TRUE)

 closePrices <- openPrices \* (1 + rnorm(length(openPrices), mean=0, sd=.10))

 closePrices <- round(closePrices, 2)

 df <- data.frame( ticker=uniqueTickers, exchange=exchanges, sector=sectors, open=openPrices, close=closePrices, news=news)

 df <- df[order(df$ticker),]

 rownames(df) <- 1:nrow(df)

 df

}

stocks <- generateRandomStockData(100)

stocks

# The Assignment

Assume that a dataframe named stocks contains information about stocks that traded today. The data is in the format shown below. Note that what is shown below is just some sample data. The dataframe may actually contain thousands of rows. Answer the questions below by writing the R code that will return the correct result.

* The open and close columns represent the opening and closing prices for that day.
* The "news" column contains logical TRUE/FALSE values. The value is TRUE if significant news about that stock came out that day (the news might be good news or bad news - you don't know). If no news came out that day then the value in the "news" column is FALSE.

**Sample stock trading data:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ticker** | **exchange** | **sector** | **open** | **close** | **news** |
| goog | nasdaq | tech | 1050.00 | 1062.50 | FALSE |
| msft | nasdaq | tech | 998.50 | 1001.50 | TRUE |
| mrk | nyse | pharm | 59.50 | 48.00 | TRUE |
| ziop | nasdaq | pharm | 30.20 | 39.20 | FALSE |
| wmt | nyse | consumer | 102.00 | 99.50 | FALSE |
| xon | nyse | energy | 80.00 | 80.05 | TRUE |
| fang | nasdaq | energy | 99.75 | 101.25 | TRUE |
| etc ... | etc ... | etc ... | etc ... | etc ... | etc ... |

You may use multiple R commands to build up to the final answer, however, the final answer must be executed using a single command. For example:

**Sample Question**: Show those stocks that are in the consumer sector and also those stocks that are in the pharm sector

**The following is a CORRECT answer that takes a single command**

> stocks [ stocks$sector == "consumer" | stocks$sector == "pharm" , ]

 ticker exchange sector open close news

3 mrk nyse pharm 59.5 48.0 TRUE

4 ziop nasdaq pharm 30.2 39.2 FALSE

5 wmt nyse consumer 102.0 99.5 FALSE

(CONTINUED ON NEXT PAGE ...)

**The following is also a CORRECT answer but this version uses more than one command**:

> consumerRows <- stocks$sector == "consumer"

> pharmRows <- stocks$sector == "pharm"

> stocks [ consumerRows| pharmRows , ] # this line returns all of the data

 ticker exchange sector open close news

3 mrk nyse pharm 59.5 48.0 TRUE

4 ziop nasdaq pharm 30.2 39.2 FALSE

5 wmt nyse consumer 102.0 99.5 FALSE

**HOWEVER, the following answer is WRONG. It's true that between the two commands all of the data is shown. However, to answer correctly your FINAL COMMAND must SHOW ALL OF THE DATA THAT WAS ASKED FOR**:

> stocks [ stocks$sector == "consumer" , ]

 ticker exchange sector open close news

5 wmt nyse consumer 102 99.5 FALSE

> stocks [ stocks$sector == "pharm" , ]

 ticker exchange sector open close news

3 mrk nyse pharm 59.5 48.0 TRUE

4 ziop nasdaq pharm 30.2 39.2 FALSE

## Questions (WITH SAMPLE SOLUTIONS)

Write R code that answers each question. Remember that the order of operations for logical operators are ! is first & is second | is third. For some answers, it might be necessary for you to use (parentheses) to override R's natural order of operations. For example:

 Just as in math order of operations is important
 2+3\*4 is 14 but (2+3)\*4 is 20.

 Similarly in R
 TRUE|FALSE&FALSE is TRUE but (TRUE|FALSE)&FALSE is FALSE

If you are having trouble, it might help for you to review the contents of the Notes files for "dataframes" and for "logical operators". See the "Notes" folder on Canvas for these and other Notes files.

NOTE : The solutions shown below are just sample solutions. It is possible that other code would also be correct.

1. Show all nasdaq stocks.

 stocks [ stocks$exchange == "nasdaq" , ]

1. Show all nasdaq stocks that went up in price.

stocks [ stocks$exchange == "nasdaq" & stocks$close > stocks$open , ]

1. Write a command that returns the number (i.e. how many) nasdaq tech stocks there are.

sum ( stocks$exchange == "nasdaq")

1. Show all **tech** and **energy** stocks.

# One possible answer:
stocks [ stocks$sector == "tech" | stocks$sector == "energy" , ]

# Another possible answer
stocks [ stocks$sector %in% c( "tech" , "energy") , ]

1. Show all tech and energy stocks on the nasdaq that had news today.

# One possible answer:
# NOTE that the parentheses in the answer ARE NECESSARY due to the fact that the order of operations
# is (1) ! (2) & (3) |
# If the parentheses weren't there the command below would return those energy stocks
# that had news, and also the "tech" stocks whether they had news or not.
stocks [ (stocks$sector == "tech" | stocks$sector == "energy" ) & stocks$news , ]

# Another possible answer, note that ==TRUE is not necessary but will work
stocks [ (stocks$sector == "tech" | stocks$sector == "energy" ) & stocks$news == TRUE, ]

1. Show all nasdaq stocks that went up by at least $10.00.

# One possible answer:
stocks[ stocks$exchange == "nasdaq" & stocks$close - stocks$open >= 10 , ]

1. Show all nasdaq stocks that went up by at least 10%.

# One possible answer:
stocks[ stocks$exchange == "nasdaq" & stocks$close >= 1.1 \* stocks$open , ]

1. Show all nasdaq stocks that went up **or down** by at least **10%**

# One possible answer. Note that the parentheses in the answer ARE necessary due to order of
# operations of !, &, |.
stocks[ stocks$exchange == "nasdaq" &
 ( stocks$close >= 1.1 \* stocks$open | stocks$close <= 0.9 \* stocks$open ) , ]

# The following, without the parentheses, is NOT CORRECT:
# If the parentheses weren't there then nasdaq stocks that we up by 10% would
# be shown but stocks that went down by 10% would be shown no matter what exchang e the stock was on.
stocks[ stocks$exchange == "nasdaq" &
 stocks$close >= 1.1 \* stocks$open | stocks$close <= 0.9 \* stocks$open , ]

1. Show all stocks that **had news** and/or fluctuated in price by at least **$10.00**.

# One possible answer - any of the following will cause a stock to be displayed:
# if the stock had news or if the stock went up by at least $10 or if the stock went down by at least $10
stocks [ stocks$news | stocks$close - stocks$open >= 10 | stocks$open - stocks$close >= 10 , ]

1. Show the average opening price of nasdaq stocks that went up

# One possible answer:
mean ( stocks [ stocks$exchange == "nasdaq" & stocks$open < stocks$close , "open" ] )

# Another possible answer - this version takes the open prices as a vector and pulls out the data from that
# vector based on the values of the other columns in the dataframe
mean ( stocks$open [ stocks$exchange == "nasdaq" & stocks$open < stocks$close ] )

1. Show all **tech** and **energy** stocks that went **up or down** by at least 10%

# One possible answer - the parentheses ARE necessary due to order of operations.
stocks[ (stocks$sector == "tech" | stocks$sector == "energy") &
 (stocks$close >= 1.1 \* stocks$open | stocks$close <= 0.9 \* stocks$close) , ]

1. Show all tech and energy stocks on the nasdaq that went up by 10% or that had news today.

# One possible answer - the parentheses ARE necessary due to order of operations.
stocks[ stocks$sector == "nasdaq" & (stocks$sector == "tech" | stocks$sector == "energy") &
 (stocks$close >= 1.1 \* stocks$open | stocks$news) , ]

1. Show all stocks that were "surprising" in that either they

had **no news** and the price went up or down by at least $10.00

or that **had news** but the price remained stable and didn't fluctuate up or down by more than $1.00

Your answer should include a single command that when executed displays BOTH types of "surprising" stocks but no other stocks.

# One possible answer - the parentheses ARE necessary due to order of operations.
stocks[ stocks$news == FALSE & # no news and
 (stocks$close - stocks$open >= 10 | stocks$open - stocks$close >= 10 ) # went up or down by at least $10
 | # OR
 stocks$news == TRUE & # yes - had news and
 abs ( stocks$open - stocks$close) <= 1 # the price remained stable (abs is the absolute value function)
 , ] # don't forget the comma

# Another possible answer - no parentheses necessary since all &'s naturally come before | so it will work as is.
stocks[ stocks$news == FALSE & # no news and
 abs(stocks$open - stocks$close) >= 10 # went up or down by at least $10 - abs is the absolute value
 | # OR
 stocks$news == TRUE & # yes - had news and
 abs ( stocks$open - stocks$close) <= 1 # the price remained stable (abs is the absolute value function)
 , ] # don't forget the comma

1. Show the difference between the average price of nasdaq **tech** stocks closing prices and nasdaq **non-tech** stock closing prices

# one possible answer - this is all one command
mean(stocks[ stocks$exchange == "nasdaq" & stocks$sector == "tech" , "close"]) -
 mean(stocks[ stocks$exchange == "nasdaq" & stocks$sector != "tech" , "close"])

# another possible answer - this is all one command
mean ( stocks$close[ stocks$exchange == "nasdaq" & stocks$sector == "tech" ] ) -
 mean ( stocks$close[ stocks$exchange == "nasdaq" & stocks$sector != "tech" ] ) -

1. Show how many (i.e. a single number) of nasdaq stocks that opened or closed below $1

# one possible answer - the parentheses are necessary
sum( stocks$exchange == "nasdaq" & (stocks$open <= 1 | stocks$close <= 1) )

1. Show all nasdaq stocks that opened or closed below $1 or that opened or closed above $1000

# one possible answer - the parentheses are necessary
stocks [ stocks$exchange == "nasdaq" &
 ( stocks$open < 1 | stocks$closed < 1 | stocks$open > 1000 | stocks$close > 1000 ) , ]

1. Show **tech** stocks on the nasdaq that opened or closed below $1 or that opened or closed above $1000

# one possible answer - the parentheses are necessary
stocks [ stocks$exchange == "nasdaq" & stocks$sector == "tech" &
 ( stocks$open < 1 | stocks$closed < 1 | stocks$open > 1000 | stocks$close > 1000 ) , ]

1. Show **tech** and **fin** stocks on the nasdaq that opened or closed below $1 or that opened or closed above $100

# one possible answer - all the parentheses are necessary
stocks [ stocks$exchange == "nasdaq" & ( stocks$sector == "tech" | stocks$sector == "fin" ) &
 ( stocks$open < 1 | stocks$closed < 1 | stocks$open > 1000 | stocks$close > 1000 ) , ]

# another possible answer - all the parentheses are necessary
stocks [ stocks$exchange == "nasdaq" & stocks$sector %in% c( "tech" , "fin" ) &
 ( stocks$open < 1 | stocks$closed < 1 | stocks$open > 1000 | stocks$close > 1000 ) , ]

1. Show **tech** stocks on the nasdaq that

had news come out today

 and also
opened or closed below $1 or opened or closed above $1000.

You should provide a single answer that includes a command that returns all the data for stocks that meet the specified conditions but no other stock data .

# one possible answer - the parentheses are necessary
stocks [ stocks$sector == "tech" & stocks$sector == "nasdaq" & stocks$news &
 ( stocks$open < 1 | stocks$close < 1 | stocks$open > 1000 | stocks$close > 1000 ) , ]

1. Show **tech** stocks on the nasdaq that opened or closed below $1 or that opened or closed above $1000.
In addition, show **any** stocks that had some news come out today.

# one possible answer - the parentheses are necessary - the spacing is just to make it easier to see where the
# parentheses match up.
stocks [ ( stocks$sector == "tech" & stocks$sector == "nasdaq" & ( stocks$open < 1 | stocks$close < 1 |
 stocks$open > 1000 | stocks$close > 1000 )
 ) | stocks$news , ]

1. Answer both parts - provide separate answers for parts (a) and (b)
	1. Did the nasdaq go up or down? Write a command that results in TRUE if the average closing price of nasdaq stocks is above the average opening price of nasdaq stocks.

# one possible answer
mean ( stocks$close [ stocks$exchange == "nasdaq" ] ) > mean ( stocks$open [ stocks$exchange == "nasdaq" ] )

# another possible answer
mean ( stocks [ stocks$exchange == "nasdaq" , "close"] ) > mean ( stocks [ stocks$exchange == "nasdaq", "open" ] )

* 1. Write a command that shows those stocks that moved in the opposite direction to the nasdaq as a whole.

	The answer to part (b) should be a single answer that works whether or not the nasdaq went up or down. If the nasdaq went up then show the stocks that went down. If the nasdaq went down, then show the stocks that went up. If you run the same exact command with data from different days, your R code should give the correct answer no matter whether the nasdaq went up or down that day.

# one possible answer - no parentheses are necessary - except for the mean() function calls
stocks [
 mean ( stocks$close [ stocks$exchange == "nasdaq" ] ) > # These two lines have the answer from part (a)
 mean ( stocks$open [ stocks$exchange == "nasdaq" ] ) # If this evaluates to TRUE then the nasdaq went UP.
 & stocks$close < stocks$open # and these stocks went DOWN
 |
 mean ( stocks$close [ stocks$exchange == "nasdaq" ] ) < # These two lines have the answer from part (a) but
 mean ( stocks$open [ stocks$exchange == "nasdaq" ] ) # with a "<" instead of ">" i.e. if this evaluates to TRUE
 # then the nasdaq went DOWN.
 & stocks$close > stocks$open # and these stocks went UP
 , ] # don't forget the comma and the closing bracket.

1. Show tech stocks whose move was 10% more than the average move for all tech stocks.

NOTE: Correction - this was not clear - I should have said "... and in the same direction (up or down) of the average move for all tech stocks", i.e. the same as the first paragraph in #23 below

> # one possible answer - for clarity, I broke this answer up into several commands

> # the following will be a vector of +1's and -1's , +1 if a stock went up and -1 if a stock went down
> directionOfStockMove <- (stocks$close - stocks$open) / abs(stocks$close - stocks$open)
> avgOpenForTech <- mean ( stocks$open [ stocks$sector == "tech" ] )
> avgCloseForTech <- mean ( stocks$close [ stocks$sector == "tech" ] )

> # the following will be a single +1 if tech went up and -1 if tech went down
> directionForTech <- (avgCloseForTech - avgOpenForTech) / abs(avgCloseForTech - avgOpenForTech)

> # the following will be a logical vector, TRUE if a stock moved in the same direction as the tech sector, FALSE otherwise
> sameDirectionAsTech <- directionOfStockMove == directionForTech

> moveForStock <- stocks$close - stocks$open # this vector contains one number for each stock
> moveForTech <- avgCloseForTech - avgOpenForTech # this is a single number

> # the following is the final answer. It shows tech stocks that moved in the same direction as the
> # tech sector and whose move was at least 10 percent more than the move for the tech sector
> stocks [ stocks$sector == "tech" & sameDirectionAsTech & moveForStock >= 1.1 \* moveForTech , ]

1. Show tech stocks whose move was

10% more than the average move for all tech stocks and in the same direction (up or down) of the average move for all tech stocks

or 10% more than the average move for all stocks (regardless of the exchange/sector) and in the same direction as the average move for all stocks

You should provide a single answer that includes a command that returns all the data for stocks that meet the specified conditions but no other stock data .

> # one possible answer - for clarity, I broke this answer up into several commands

> #########################################################
> # THE FOLLOWING IS COPIED FROM THE ANSWER TO THE PREVIOUS QUESTION
> #########################################################

> # the following will be a vector of +1's and -1's , +1 if a stock went up and -1 if a stock went down
> directionOfStockMove <- (stocks$close - stocks$open) / abs(stocks$close - stocks$open)
> avgOpenForTech <- mean ( stocks$open [ stocks$sector == "tech" ] )
> avgCloseForTech <- mean ( stocks$close [ stocks$sector == "tech" ] )

> # the following will be a single +1 if tech went up and -1 if tech went down
> directionForTech <- (avgCloseForTech - avgOpenForTech) / abs(avgCloseForTech - avgOpenForTech)

> # the following will be a logical vector, TRUE if a stock moved in the same direction as the tech sector, FALSE otherwise
> sameDirectionAsTech <- directionOfStockMove == directionForTech

> moveForStock <- stocks$close - stocks$open # this vector contains one number for each stock
> moveForTech <- avgCloseForTech - avgOpenForTech # this is a single number

> #######################################################################################
> # UP TO HERE WAS THE SAME AS FOR THE PREVIOUS QUESTION - THE REST FROM HERE DOWN IS NEW
> #######################################################################################

> avgOpenForAllStocks <- mean ( stocks$open )
> avgCloseForAllStocks <- mean ( stocks$close )

> # the following will be a single +1 if stocks on average went up and -1 if stocks on average went down
> directionForMarket <- (avgCloseForAllStocks - avgOpenForAllStocks) /
 abs(avgCloseForAllStocks - avgOpenForAllStocks)

> # the following will be a logical vector, TRUE if a stock moved in the same direction as the tech sector, FALSE otherwise
> sameDirectionAsMarket <- directionOfStockMove == directionForMarket

> # the following is the final answer
> stocks [ stocks$sector == "tech" & # show tech stocks
 ( # (the rest of the logic is the other side of the & on the previous line)
 ( sameDirectionAsTech & moveForStock >= 1.1 \* moveForTech ) # moved same dir as tech and 10% more
 | # or
 ( sameDirectionAsMarket & moveForStock >= 1.1 \* moveForMarket ) # mved in same dir as Mkt and 10% more
 )
 , ]