5.5

# Requirements: the chess object from the previous exercise.

Player <- c("dark","dark","dark","dark","dark","light","light","light","light","light")

Piece <- c("king", "queen", "pawn", "pawn", "knight", "bishop", "king", "rook", "pawn", "pawn")

chess <- cbind(Player, Piece)

print(chess)

# Transpose the chess matrix,

# and add a row storing the following values: 3, 5, 2, 2, 7, 4, 6, 5, 2, 1

chessTranspose <- t(chess)

chessTransposeA group of black text

Description automatically generated

A screenshot of a computer code

Description automatically generated

# Name the row "Turn" and transpose the matrix back to its original orientation.

rownames(chessTranspose)[3] <- "Turn"

A white background with black text

Description automatically generated

chess <- t(chessTranspose)

A screenshot of a computer code

Description automatically generated

# Extract the following values from the chess matrix:

#The first piece of the light player

chess[6,"Piece"]

A close up of a text

Description automatically generated

#The Player and Piece columns

chess[,c("Player","Piece")]

A screenshot of a computer code

Description automatically generated

#All the information about the dark player

A close up of text

Description automatically generated

#The Pieces column; try to extract that as a matrix (Hint: lookup the drop = argument)

chess[, "Piece", drop = FALSE]

A screenshot of a computer code

Description automatically generated

#Everything but the Piece column

A screenshot of a computer code

Description automatically generated

#The 1st and 3rd values on the second row

chess[2, c(1, 3)]

chess[2, c("Player", "Turn")]

A close up of a text

Description automatically generated

#Replace the 3rd value on the 7th row with 3 (Hint: works just like creating an object)

A screenshot of a computer code

Description automatically generated

5.6

# Lookup the runif() function.

# Create a 3x4 matrix with 12 random numbers generated

# using the runif() function;

# have the matrix be filled our row-by-row,

# instead of column-by-column.

?runif

A screenshot of a computer

Description automatically generated

randNum <- runif(12,0,12)

mtx <- matrix(randNum, nrow = 3, byrow = T)

A screenshot of a computer

Description automatically generated

# Name the columns of the matrix uno, dos, tres, cuatro, and the rows x, y, z.

A white background with black text

Description automatically generated

# Scale the matrix by 10 and save the result.

A screenshot of a computer

Description automatically generated

# Extract a 2x4 matrix from it and save the result.

A close-up of numbers

Description automatically generated

# Subtract the smaller matrix from the larger one. Can you do that? Why?

No, because it should be same size to subtract  


# Extract a 3x3 matrix from the original matrix and save the result. Try the subtraction again. Can you do that? Why?

A screenshot of a computer code

Description automatically generated

No, because they are still not same size.



# Extract the column called "uno" as a vector from the original matrix and save the result. Try the subtraction again. Can you do that? Why?

A close up of numbers

Description automatically generated

is.vector(uno)



This time, it works. We can do due to recycling of matrix and vector, vector uno will repeat 1 more row to have same size with mtx\_2x4 and then subtracting

A white background with black text

Description automatically generated

# Lookup the rnorm() function. Create a new 3x4 matrix with 12 random values generated using the rnorm() function.

?rnorm

A screenshot of a computer program

Description automatically generated

A close-up of numbers

Description automatically generated

# Perform matrix multiplication (using the \* sign). Can you do that? How is the operation carried out?

A number and numbers on a white background

Description automatically generated

Yes, we can do. I will multiply each element of matrix with this number (3).

# Perform inner matrix multiplication with the two matrixes. Can you do that? Why? Can you think of something to do to make this possible?



No, We can not do inner matrix multiplication with 2 matrix 3x4.

To perform inner matrix multiplication between two matrices A and B, the number of columns of A must be equal to the number of rows of B. If A is a matrix of size m x n (i.e., m rows and n columns) and B is a matrix of size n x p (i.e., n rows and p columns), then the result of inner matrix multiplication between A and B is a matrix C of size m x p.

In this case, we can transpose the matrix and then do inner multiplication.

A number and percentages on a white background

Description automatically generated

5.7

# Create a 5x5 matrix with the rnorm() function,

# and a 5x5 matrix with runif(). Create each in a single line of code (Hint: nest the operations)

matrix1 <- matrix(rnorm(25), nrow = 5)

matrix2 <- matrix(runif(25), nrow = 5)

matrix1

matrix2

A screenshot of a computer code

Description automatically generated

# For the two matrices, get the following information; for the first four, save the new values as columns in their corresponding matrixes:

#Column averages

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

#Row averages

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

#Column sums

A screenshot of a computer

Description automatically generated

A screenshot of a computer

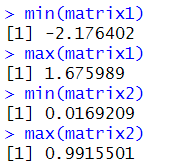
Description automatically generated

#Row sums

A screenshot of a computer

Description automatically generated

#Minimum and maximum value in the matrix



#Minimum and maximum value for the 3rd column in each matrix

A screenshot of a computer code

Description automatically generated

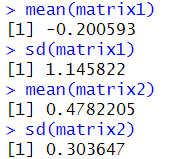
#The means and standard deviations for each matrix

# (compare the two values;

# if interested in the mathematics side of things,

# recreate the matrices a couple of times,

# and compare the results; can you explain what is happening?)



Matrix1 create by rnorm() - The Normal Distribution

🡪value of mean is near 0 and sd is near 1 – because we use default rnorm() and it already set to that numbers

A screenshot of a computer program

Description automatically generated

Similarly for matrix2, create by runif() - The Uniform Distribution

* Value of mean near 0.5, and sd is round 0.25 – 0.3. because I use default runif() with min =0, max =1, and mean of Uniform distribution is (0+1)/2 = 0.5 and sd has formular: (1 - 0) / sqrt(12) ≈ 0.2887

A screenshot of a computer

Description automatically generated

matrix11 <- matrix(rnorm(25), nrow = 5)

matrix22 <- matrix(runif(25), nrow = 5)

mean(matrix11)

sd(matrix11)

mean(matrix22)

sd(matrix22)

5.9

# Requirements: the chess matrix from Exercise 11.

Player <- c("dark","dark","dark","dark","dark","light","light","light","light","light")

Piece <- c("king", "queen", "pawn", "pawn", "knight", "bishop", "king", "rook", "pawn", "pawn")

chess <- cbind(Player, Piece)

print(chess)

# Save the Piece column of the matrix as a vector.

piece <- chess[,"Piece"]

piece

A black text on a white background

Description automatically generated

# Create a factor from the vector.

piece\_factor <- factor(piece)

piece\_factor

# Organize the levels in the following way but do not order them:

# King, Queen, Rook, Bishop, Knight, Pawn.

piece\_factor <- factor(piece,

levels=c("king", "queen", "rook", "bishop", "knight", "pawn"))

piece\_factor

A white background with black text

Description automatically generated

# Rename the levels with just their initial letters.

# Order the levels in the way specified above.

piece\_factor2 <- factor(piece, order = T,

levels=c("king", "queen", "rook", "bishop", "knight", "pawn"),

labels = c("K","Q","R","B","K","P"))

str(piece\_factor2)

A math equations and symbols

Description automatically generated

# create a list which prints like this:

# [[1]]

# [1] 1 3 5 7 9 11

#

# [[2]]

# [[2]][[1]]

# [1] "Happy Birthday"

#

# [[2]][[2]]

# [1] "Archery"

list <- list(c(1,3,5,7,9,11),list("Happy Birthday","Archery"))

A white background with blue text

Description automatically generated

# extract the numbers as a vector

vector1 <- list[[1]]



# extract the phrase Happy Birthday as a vector

hb <- list[[2]][[1]]

A close up of a text

Description automatically generated

# extract the second item of the second list as a list

list[[2]][2]

A black and blue text

Description automatically generated

# extract the second list as a list

A screenshot of a computer screen

Description automatically generated

# extract the numbers item as a list

A number on a white background

Description automatically generated

# add 2 to each element in the numbers item



# name the items in the list as "Numbers" and "Phrases"

list <- list(Number = c(1,3,5,7,9,11),

Phrases = list("Happy Birthday","Archery"))

A white background with blue text

Description automatically generated

# you can use the $ to extract named items of a list



# if you extract the numbers item from newList with the $, what other extraction method is this equivalent to?



# use the dollar sign to repeat the addition from above (add 2 to each element in the numbers list)



# add a new item called "Brands" to the list. It should contain the brands Kellogs, Nike, iPhone

A computer code with text

Description automatically generated

# use either brackets or the dollar sign to do that

A screenshot of a computer code

Description automatically generated

# remove the iPhone from the Brands item

A screenshot of a computer code

Description automatically generated

# remove the Brands item from the list

A screenshot of a computer code

Description automatically generated