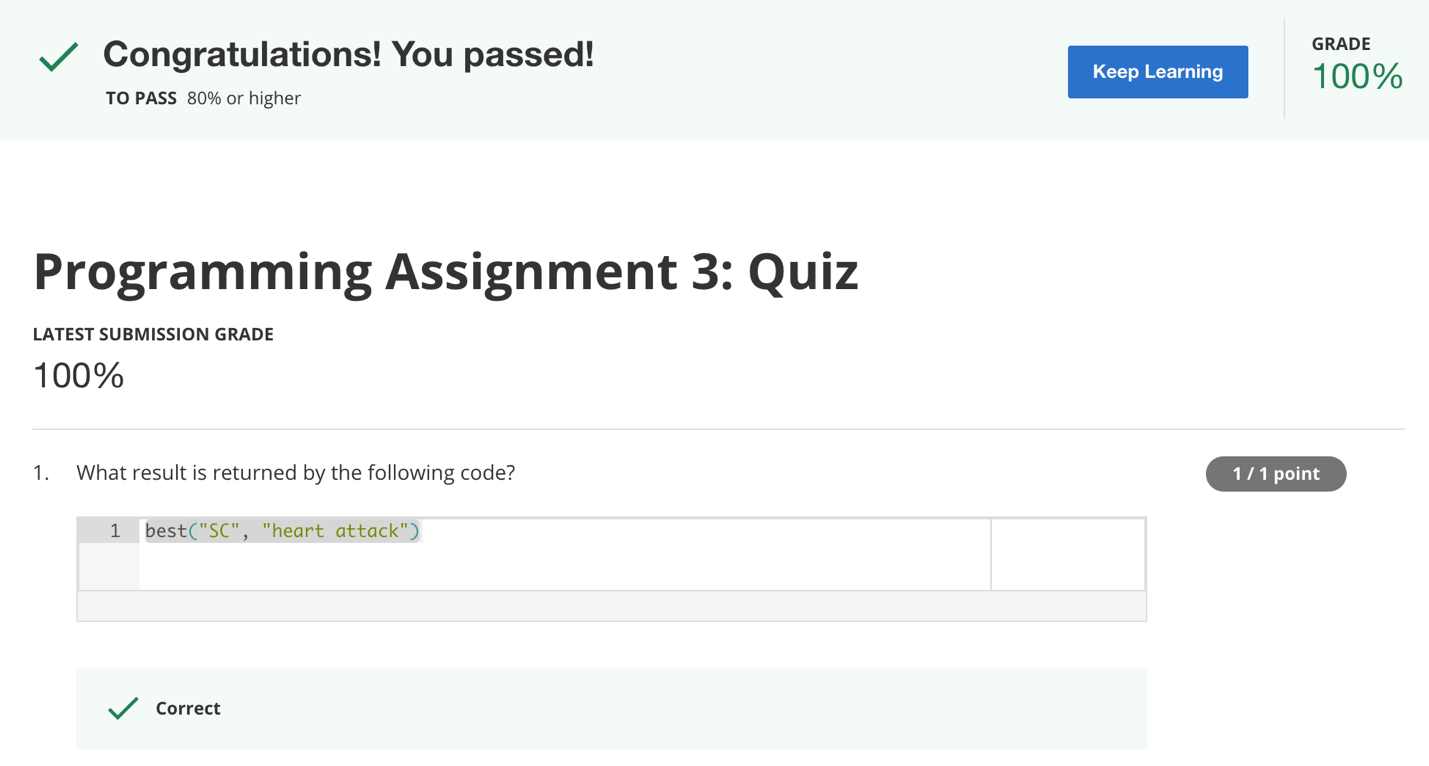
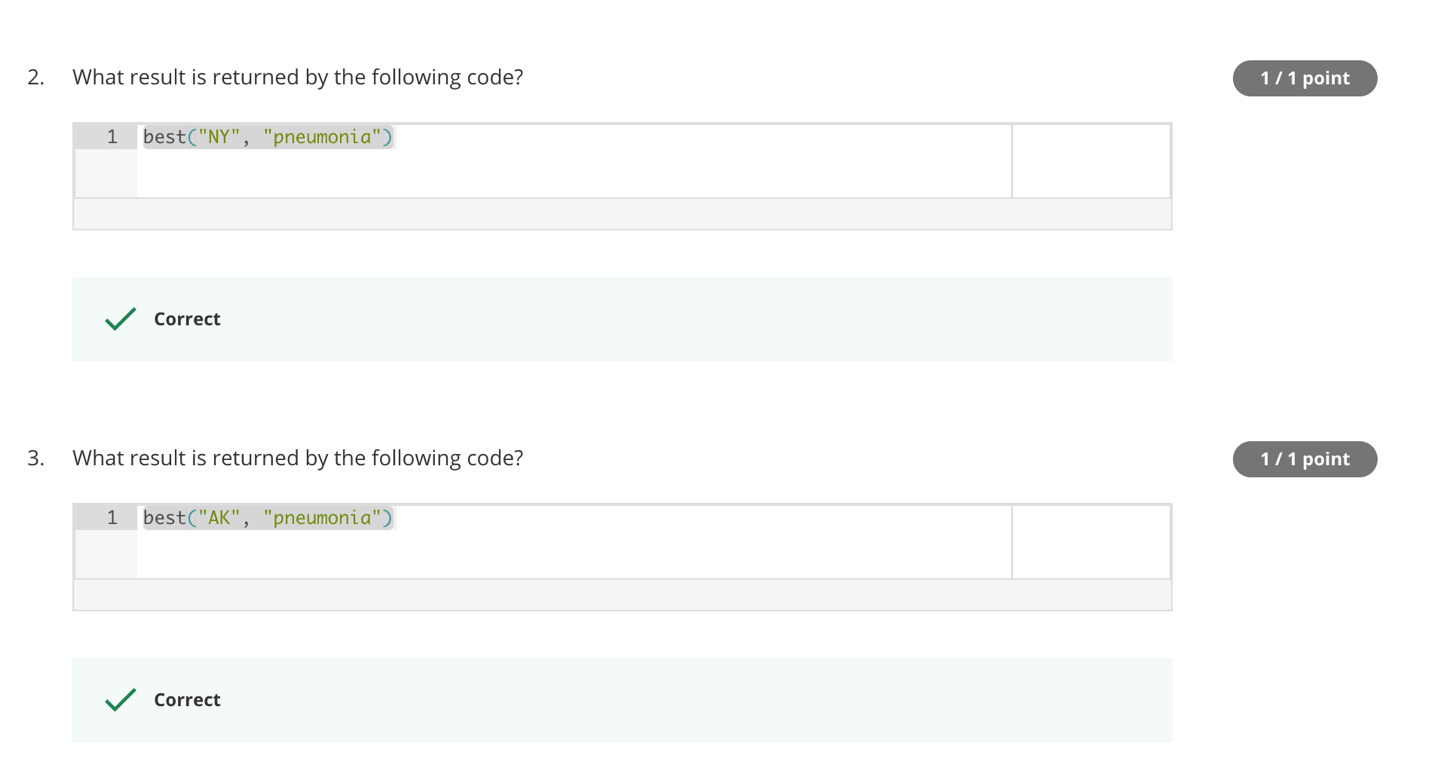
Programming Assignment 3 Quiz

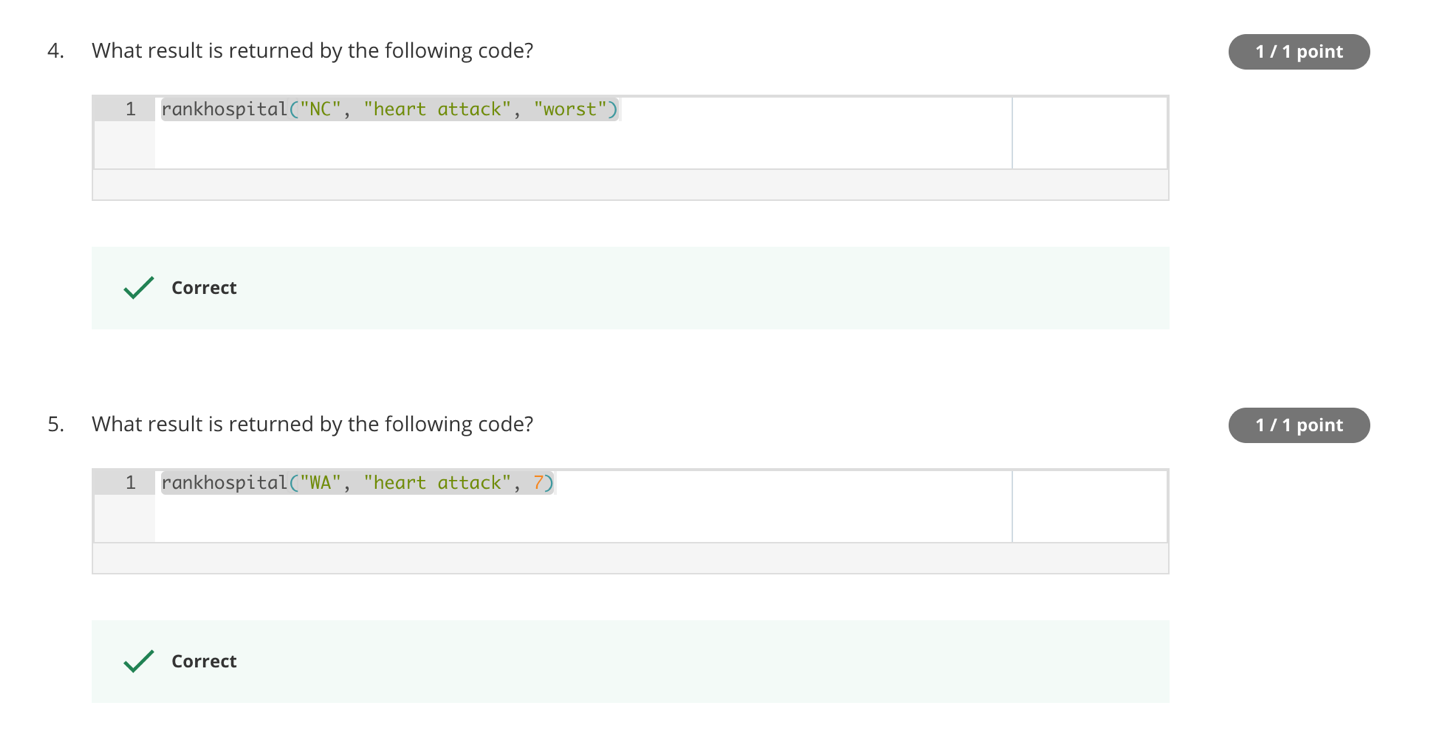


Q1 - best("SC", "heart attack")

Q2 - best("NY", "pneumonia")

Q3 - rankhospital("NC", "heart attack", "worst")



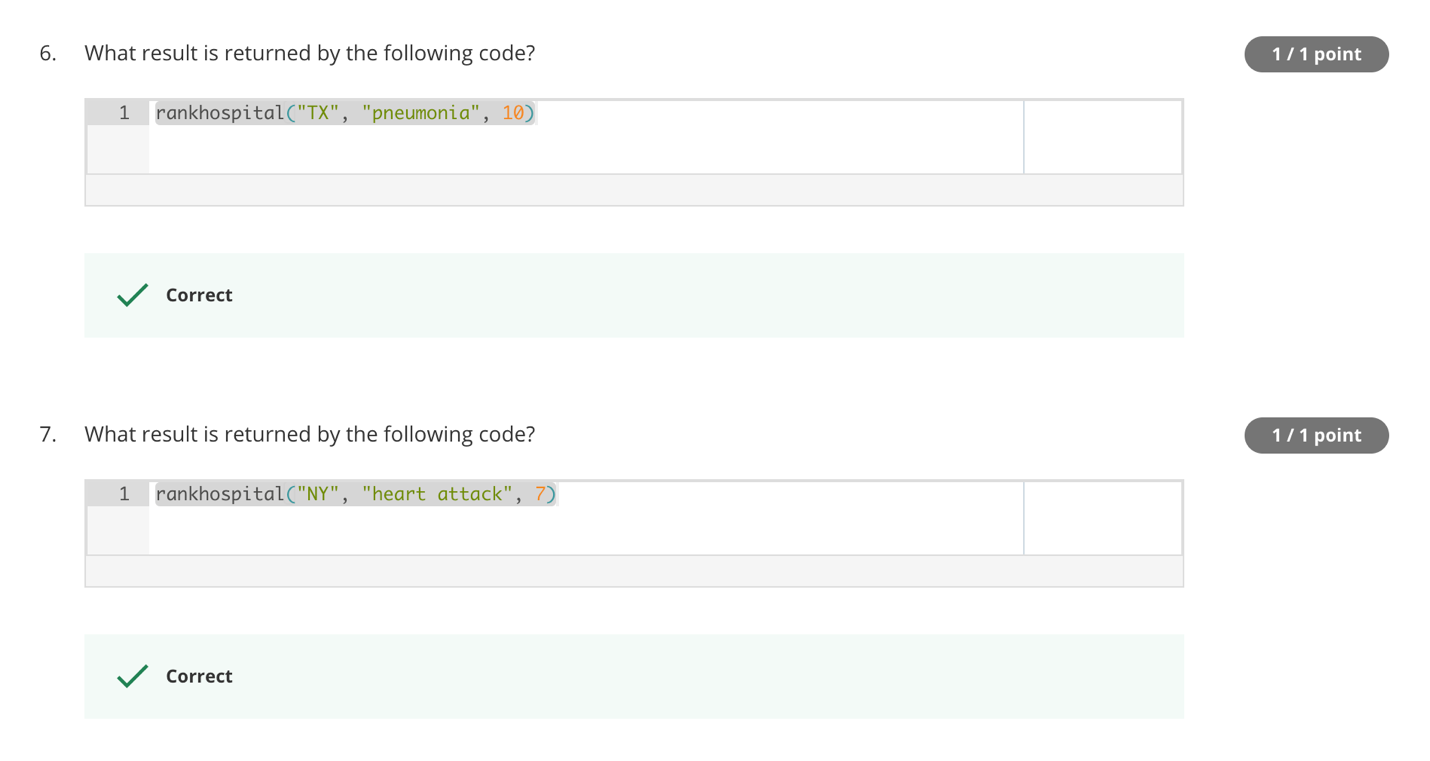


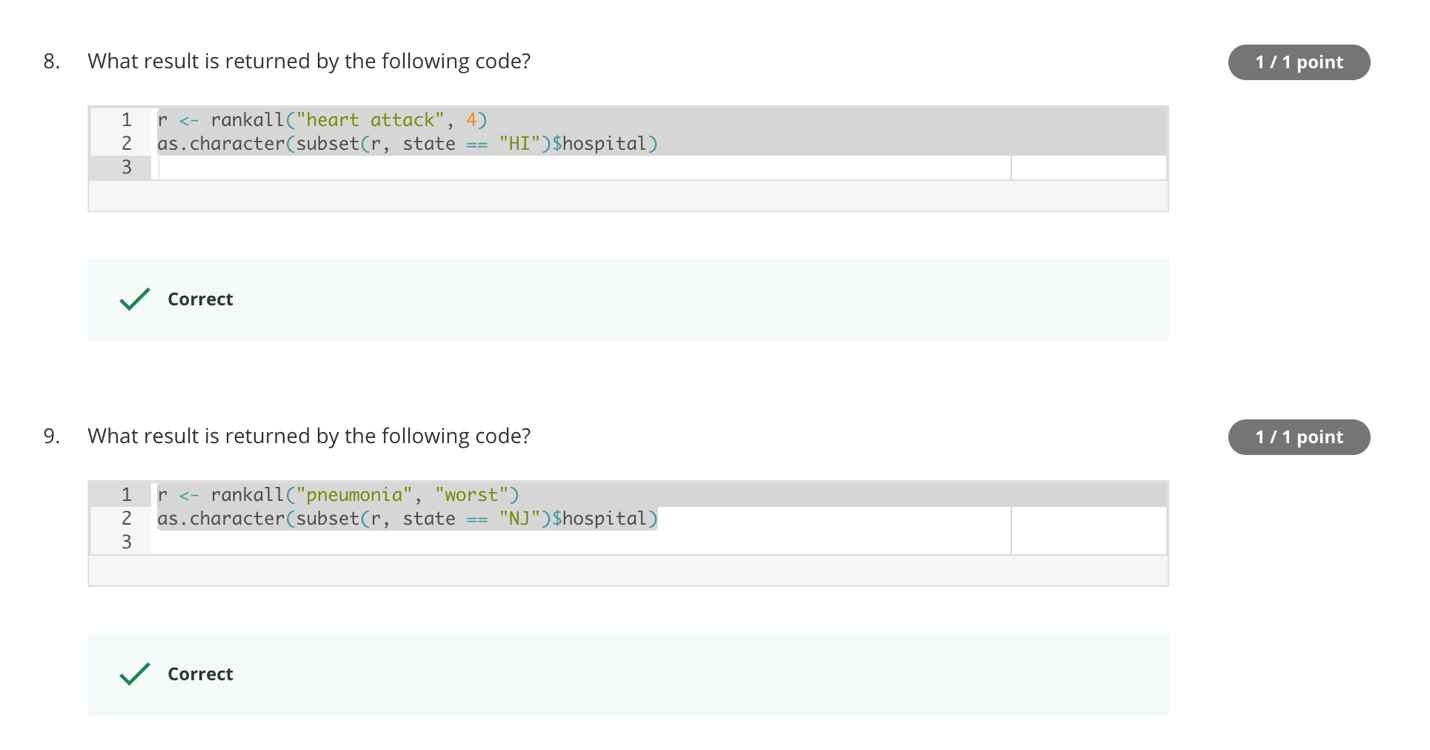
Q4 - rankhospital("NC", "heart attack", "worst")

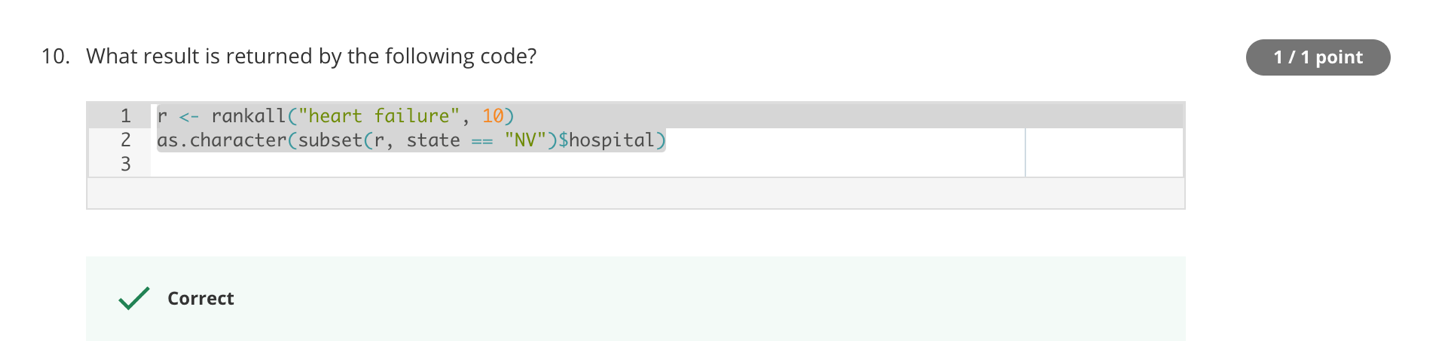
Q5 - rankhospital("WA", "heart attack", 7)

Q6 - rankhospital("TX", "pneumonia", 10)

Q7 - rankhospital("NY", "heart attack", 7)







Q8 - r <- rankall("heart attack", 4)

as.character(subset(r, state == "HI")$hospital)

Q9 - r <- rankall("pneumonia", "worst")

as.character(subset(r, state == "NJ")$hospital)

Q10 - r <- rankall("heart failure", 10)

as.character(subset(r, state == "NV")$hospital)

# best.R

# Write a function called best that take two arguments: the 2-character

# abbreviated name of a state and an outcome name. The function reads

# the outcome-of-care-measures.csv file and returns a character vector

# with the name of the hospital that has the best (i.e. lowest) 30-day

# mortality for the specified outcome in that state. The hospital name

# is the name provided in the Hospital.Name variable. The outcomes can

# be one of “heart attack”, “heart failure”, or “pneumonia”. Hospitals

# that do not have data on a particular outcome should be excluded

# from the set of hospitals when deciding the rankings.

# The function should use the following template.

best <- function(state, outcome) {

## This function will do the following

## 1) Read outcome data

## 2) Check that state and outcome are valid

## 3) Return hospital name in that state with lowest 30-day death rate

## 1) Read outcome data

outcome\_data <- read.csv("outcome-of-care-measures.csv",

colClasses = "character")

## 2) Check that state and outcome are valid

states <- unique(outcome\_data$State)

## Validate the input parameter 'state'

if (is.na(match(state, states))) {

return ("invalid state")

}

# [2] "Hospital.Name"

# [7] "State"

# [11] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Attack"

# [17] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure"

# [23] "Hospital.30.Day.Death..Mortality..Rates.from.Pneumonia"

## common column: State, Hospital.Name

cols <- c(2, 7)

## Create a mapping of valid 'outcome' parameter to column index

outcome\_map <- new.env(hash = TRUE)

outcome\_map[["heart attack"]] <- 11

outcome\_map[["heart failure"]] <- 17

outcome\_map[["pneumonia"]] <- 23

## Validate the 'outcome' parameter

index <- outcome\_map[[outcome]]

if (is.null(index)) {

return(" invalid outcome")

}

## Create query column indexes

cols <- c(cols, index)

## Query

## 1) select subset by given column indexes 'cols'

## 2) select records that have state match 'state' parameter

df <- subset(outcome\_data[cols], State == state)

## Change the outcome colunm name to "Rate"

names(df)[3] <- "Rate"

## Creates or coerces objects of type 'numeric'

df[, length(cols)] <- as.numeric(df[, length(cols)])

## Filter out the NA on the 'outcome' parameter

## df <- subset(df, !is.na(df[, 3])) or

df <- subset(df, !is.na(df$Rate))

## Sort df by 'outcome', State, which is c(3,2)

order.by = order(df[,3], df[,1])

df <- df[order.by,]

## 3) Return hospital name in that state with lowest 30-day death rate

df[1,1]

}

## Testing

## source("best.R")

## 1) best("SC", "heart attack")

## [1] "MUSC MEDICAL CENTER"

## 2) best("NY", "pneumonia")

## [1] "MAIMONIDES MEDICAL CENTER"

## 3) best("AK", "pneumonia")

## [1] "YUKON KUSKOKWIM DELTA REG HOSPITAL"

# rankhospital.R

# Write a function called rankhospital that takes three arguments:

# the 2-character abbreviated name of astate (state), an outcome (outcome),

# and the ranking of a hospital in that state for that outcome (num).

# The function reads the outcome-of-care-measures.csv file and returns a

# character vector with the name of the hospital that has the ranking

# specified by the num argument.

# For example, the call

# rankhospital("MD", "heart failure", 5)

# would return a character vector containing the name of the hospital with

# the 5th lowest 30-day death rate for heart failure. The num argument can

# take values “best”, “worst”, or an integer indicating the ranking

# (smaller numbers are better). If the number given by num is larger than

# the number of hospitals in that state, then the function should return NA.

# Hospitals that do not have data on a particular outcome should

# be excluded from the set of hospitals when deciding the rankings.

rankhospital <- function(state, outcome, num = "best") {

## 1) Read outcome data

## 2) Check that state and outcome are valid

## 3) Return hospital name in that state with the given rank

## 30-day death rate

## 1) Read outcome data

outcome\_data <- read.csv("outcome-of-care-measures.csv",

colClasses = "character")

## 2) Check that state and outcome are valid

states <- unique(outcome\_data$State)

## Validate the input parameter 'state'

if (is.na(match(state, states))) {

# mesg <- sprintf('Error in best("%s", "%s") : invalid state', state, outcome)

return ("invalid state")

}

# [2] "Hospital.Name"

# [7] "State"

# [11] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Attack"

# [17] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure"

# [23] "Hospital.30.Day.Death..Mortality..Rates.from.Pneumonia"

## common column: State, Hospital.Name

cols <- c(2, 7)

## Create a mapping of valid 'outcome' parameter to column index

outcome\_map <- new.env(hash = TRUE)

outcome\_map[["heart attack"]] <- 11

outcome\_map[["heart failure"]] <- 17

outcome\_map[["pneumonia"]] <- 23

## Validate the 'outcome' parameter

index <- outcome\_map[[outcome]]

if (is.null(index)) {

return(" invalid outcome")

}

## Create query column indexes

cols <- c(cols, index)

## Query

## 1) select subset by given column indexes 'cols'

## 2) select records that have state match 'state' parameter

df <- subset(outcome\_data[cols], State == state)

## Change the outcome colunm name to "Rate"

names(df)[3] <- "Rate"

## Creates or coerces objects of type 'numeric'

df[, length(cols)] <- as.numeric(df[, length(cols)])

## Filter out the NA on the 'outcome' parameter

## df <- subset(df, !is.na(df[, 3])) or

df <- subset(df, !is.na(df$Rate))

## Sort df by 'outcome', State, which is c(3,2)

order.by = order(df[,3], df[,1])

df <- df[order.by,]

df$Rank <- 1:nrow(df)

## 3) Return hospital name in that state with the given rank

## 30-day death rate

if (num == "best") {

df[1, 1]

} else if (num == "worst") {

df[nrow(df), 1]

} else {

nth <- as.integer(num)

df[nth, 1]

}

}

## Testing

## source("rankhospital.R")

## 1) rankhospital("NC", "heart attack", "worst")

## [1] "WAYNE MEMORIAL HOSPITAL"

## 2) rankhospital("WA", "heart attack", 7)

## [1] "YAKIMA VALLEY MEMORIAL HOSPITAL"

## 3) rankhospital("TX", "pneumonia", 10)

## [1] "SETON SMITHVILLE REGIONAL HOSPITAL"

## 4) rankhospital("NY", "heart attack", 7)

## [1] "BELLEVUE HOSPITAL CENTER"

# source("rankall.R")

# Write a function called rankall that takes two arguments: an outcome

# name (outcome) and a hospital ranking (num). The function reads the

# outcome-of-care-measures.csv file and returns a 2-column data frame

# containing the hospital in each state that has the ranking specified

# in num. For example the function call rankall("heart attack", "best")

# would return a data frame containing the names of the hospitals that

# are the best in their respective states for 30-day heart attack death

# rates. The function should return a value for every state (some may be NA).

# The first column in the data frame is named hospital, which contains

# the hospital name, and the second column is named state, which contains

# the 2-character abbreviation for the state name. Hospitals that do not

# have data on a particular outcome should be excluded from the set of

# hospitals when deciding the rankings. Handling ties. The rankall function

# should handle ties in the 30-day mortality rates in the same way

# that the rankhospital function handles ties.

# The function should use the following template.

# 'num' can be "best", "worst" or an integer

rankall <- function(outcome, num = "best") {

## 1) Read outcome data

## 2) Check that state and outcome are valid

## 3) For each state, find the hospital of the given rank

## 4) Return a data frame with the hospital names and the

## (abbreviated) state name

## 1) Read outcome data

outcome\_data <- read.csv("outcome-of-care-measures.csv",

colClasses = "character")

## 2) Check that state and outcome are valid

states <- sort(unique(outcome\_data$State))

## Validate the input parameter 'state'

if (is.na(match(state, states))) {

return ("invalid state")

}

# [2] "Hospital.Name"

# [7] "State"

# [11] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Attack"

# [17] "Hospital.30.Day.Death..Mortality..Rates.from.Heart.Failure"

# [23] "Hospital.30.Day.Death..Mortality..Rates.from.Pneumonia"

## common column: State, Hospital.Name

cols <- c(2, 7)

## Create a mapping of valid 'outcome' parameter to column index

outcome\_map <- new.env(hash = TRUE)

outcome\_map[["heart attack"]] <- 11

outcome\_map[["heart failure"]] <- 17

outcome\_map[["pneumonia"]] <- 23

## Validate the 'outcome' parameter

index <- outcome\_map[[outcome]]

if (is.null(index)) {

return(" invalid outcome")

}

## Create query column indexes, i.e. Hospital.Name, State, ..Rates...

cols <- c(cols, index)

## Query

## 1) select subset by given column indexes 'cols'

## 2) select records that have state match 'state' parameter

df <- outcome\_data[cols]

## Change the 'outcome' colunm name to "Rate"

names(df)[3] <- "Rate"

## Creates or coerces objects of type 'numeric'

df[, length(cols)] <- as.numeric(df[, length(cols)])

## Filter out the NA on the 'outcome' parameter

## df <- subset(df, !is.na(df[, 3])) or

df <- subset(df, !is.na(df$Rate))

## Sort df by State, Rate, Hospital.Name

order.by = order(df[,2], df[,3], df[,1])

df <- df[order.by, ]

## Processing the each state and select the hospital by 'num'

## First create a data frame storing result of the query

ans <- data.frame(matrix(nrow = length(states), ncol = 2))

rownames(ans) <- states

colnames(ans) <- c("hospital", "state")

ans[, 2] <- states

# Loop through each of the states, 52 + Puerto Rico + Guam

for (state in states) {

df\_state <- subset(df, df$State == state)

if (num == "best") {

ans[state, 1] <- df\_state[1, 1]

} else if (num == "worst") {

ans[state, 1] <- df\_state[nrow(df\_state), 1]

} else {

nth <- as.integer(num)

ans[state, 1] <- df\_state[nth, 1]

}

}

ans

}

## Testing

## source("rankall.R")

## 1) r <- rankall("heart attack", 4)

## as.character(subset(r, state == "HI")$hospital)

## [1] "CASTLE MEDICAL CENTER"

## 2) r <- rankall("pneumonia", "worst")

## as.character(subset(r, state == "NJ")$hospital)

## [1] "BERGEN REGIONAL MEDICAL CENTER"

## 3) r <- rankall("heart failure", 10)

## as.character(subset(r, state == "NV")$hospital)

## [1] "RENOWN SOUTH MEADOWS MEDICAL CENTER"