R Programming Week 4 Assignment

Ken Wood

7/10/2020

```
rm(list=ls(all=TRUE))
```

```
best <- function(state, outcome) {

  ## Read outcome data
  df <- read.csv("outcome-of-care-measures.csv", na.strings="Not Available",stringsAsFactors=FALSE,   c

## Check that state and outcome arguments are valid
  if ((state %in% df$State == TRUE) && (outcome %in% c("heart attack","heart failure","pneumonia") == TR
  outcomes <- c("heart attack"=11, "heart failure"=17, "pneumonia"=23)

  # Get df columns 2, 7, and the numerical equivalent of outcome according to the 'outcomes' vector.
  my_df <- df[,c(2,7,outcomes[outcome])]

  # Set the column names for the subsetted dataframe, 'my_df'
  names(my_df) <- c("Hospital", "State", "Mortality_Rate")

  # Filter the 'my_df' rows on state
  my_df <- my_df[my_df$State==state,]

  # Eliminate any rows with NA values
  my_df <- my_df[complete.cases(my_df),]

  # Convert the 'Mortality_Rate' column to a numeric variable type
  my_df$Mortality_Rate <- as.numeric(unlist(my_df$Mortality_Rate))

  # Sort 'my_df' first by "Mortality Rate" ascending, then by "Hospital"
  my_df <- my_df[with (my_df, order(Mortality_Rate,Hospital)),]

  ## Return hospital name in that state with lowest 30-day death rate
  result <- my_df[1,1]

    # Error handling for incorrect inputs.

    } else {
      ifelse(state %in% df$State == FALSE,stop("invalid state"),stop("invalid outcome"))

      # Alternate code to generate error messages for invalid inputs

      #if (state %in% df$State == FALSE) {   # generate error message for invalid state input
        #stop("invalid state")
      #} else {
```

```
       # stop("invalid outcome")             # generate error message for invalid outcome input
     # }
    }
result
}
```

```
best("AK","heart failure")
```

```
## [1] "SOUTH PENINSULA HOSPITAL"
```

```
best("SC","heart attack")
```

```
## [1] "MUSC MEDICAL CENTER"
```

```
best("NY","pneumonia")
```

```
## [1] "MAIMONIDES MEDICAL CENTER"
```

```
best("AK","pneumonia")
```

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

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

  ## Read outcome data
  df <- read.csv("outcome-of-care-measures.csv", na.strings="Not Available",stringsAsFactors=FALSE,   c

## Check that state and outcome arguments are valid
  if ((state %in% df$State == TRUE) && (outcome %in% c("heart attack","heart failure","pneumonia") == TR
  outcomes <- c("heart attack"=11, "heart failure"=17, "pneumonia"=23)

  # Get df columns 2, 7, and the numerical equivalent of outcome according to the 'outcomes' vector.
  my_df <- df[,c(2,7,outcomes[outcome])]

  # Set the column names for the subsetted dataframe, 'my_df'
  names(my_df) <- c("Hospital", "State", "Mortality_Rate")

  # Filter the 'my_df' rows on state
  my_df <- my_df[my_df$State==state,]

  # Eliminate any rows with NA values
  my_df <- my_df[complete.cases(my_df),]

  # Convert the 'Mortality_Rate' column to a numeric variable type
  my_df$Mortality_Rate <- as.numeric(unlist(my_df$Mortality_Rate))

  # Sort 'my_df' first by "Mortality Rate" ascending, then by "Hospital"
  my_df <- my_df[with (my_df, order(Mortality_Rate,Hospital)),]

  # Add rank column to 'my_df'.  In case of tie in 'Mortality_Value' numbers, first value wins.
  my_df$Rank <- rank(my_df$Mortality_Rate,ties.method= "first")

  # Develop logic if user enters "best" or "worst" for 'num' argument.
  rownum <- integer()

  if (num =="best") {
    rownum = 1
```

```
    } else {
      if (num == "worst") {
        rownum = nrow(my_df)
        } else {
          rownum=num
        }
}

  ## Return hospital name in that state with lowest 30-day death rate
  result <- my_df[rownum,1]

  # Error handling for incorrect inputs.

    } else {
      ifelse(state %in% df$State == FALSE,stop("invalid state"),stop("invalid outcome"))

      # Alternate code to generate error messages for invalid inputs

      #if (state %in% df$State == FALSE) {   # generate error message for invalid state input
        #stop("invalid state")
      #} else {
       # stop("invalid outcome")             # generate error message for invalid outcome input
     # }
    }
result
}
```

```
rankhospital("NC","heart attack","worst")
```

```
## [1] "WAYNE MEMORIAL HOSPITAL"
```

```
rankhospital("WA","heart attack",7)
```

```
## [1] "YAKIMA VALLEY MEMORIAL HOSPITAL"
```

```
rankhospital("TX","pneumonia",10)
```

```
## [1] "SETON SMITHVILLE REGIONAL HOSPITAL"
```

```
rankhospital("NY","heart attack",7)
```

```
## [1] "BELLEVUE HOSPITAL CENTER"
```

```
rankall <- function(outcome, num = "best") {
    ## Read outcome data
    ## Check that state and outcome are valid
    ## For each state, find the hospital of the given rank
    ## Return a data frame with the hospital names and the
    ## (abbreviated) state name

    #Make list of possible values of outcome and their index
    possible.outcomes <- list("heart attack" = 11, "heart failure" = 17, "pneumonia" = 23)
    outcome.col <- possible.outcomes[[outcome]]

    #Stop if outcome was not in possible.outcomes
    if (is.null(outcome.col))
        stop("invalid outcome")
```

```
    #Read the csv
    raw_df <- read.csv("outcome-of-care-measures.csv", colClasses = "character")

    #Convert the desired column to numeric
    raw_df[, outcome.col] <- suppressWarnings(sapply(raw_df[, outcome.col], as.numeric))

    #Make data.frame for all states
    working_df <- subset(raw_df, select = c(outcome.col,2, 7))

    # Set the column names for the subsetted dataframe, 'working_df'
    names(working_df) <- c("Mortality_Rate","Hospital", "State")

    # Eliminate any rows with NA values
    working_df <- working_df[complete.cases(working_df),]

    # Split the 'working_df' into dataframes grouped by State
    split_df <- split(working_df, working_df$State)

    staterank <- function(working_df) {
        #Make list of positions
        rank.list <- order(working_df$Mortality_Rate,working_df$Hospital, na.last = NA)

        #Check validity of num argument and assign numeric value
        if (num == "best")
            num <- 1
        else if (num == "worst")
            num <- length(rank.list)
        else if (!is.numeric(num))
            stop("Unrecognised num argument")

        working_df[rank.list[num],2]
    }

    ranked.states <- data.frame(sapply(split_df, staterank))
    ranked.states <- data.frame(ranked.states, row.names(ranked.states))
    names(ranked.states) <- c("Hospital", "State")
    ranked.states
}
```

```
rankall("heart attack","best")
```

```
##                                   Hospital State
## AK         PROVIDENCE ALASKA MEDICAL CENTER    AK
## AL                 CRESTWOOD MEDICAL CENTER    AL
## AR                  ARKANSAS HEART HOSPITAL    AR
## AZ                     MAYO CLINIC HOSPITAL    AZ
## CA        GLENDALE ADVENTIST MEDICAL CENTER    CA
## CO     ST MARYS HOSPITAL AND MEDICAL CENTER    CO
## CT                       WATERBURY HOSPITAL    CT
## DC                      PROVIDENCE HOSPITAL    DC
## DE        BAYHEALTH - KENT GENERAL HOSPITAL    DE
## FL               MOUNT SINAI MEDICAL CENTER    FL
## GA                 STEPHENS COUNTY HOSPITAL    GA
## GU         GUAM MEMORIAL HOSPITAL AUTHORITY    GU
```

```
## HI                      HILO MEDICAL CENTER    HI
## IA              MARY GREELEY MEDICAL CENTER    IA
## ID                  PORTNEUF MEDICAL CENTER    ID
## IL                    SAINT JOSEPH HOSPITAL    IL
## IN   ST VINCENT HEART CENTER OF INDIANA LLC    IN
## KS                    KANSAS HEART HOSPITAL    KS
## KY        ST ELIZABETH MEDICAL CENTER NORTH    KY
## LA                ST FRANCIS MEDICAL CENTER    LA
## MA     BETH ISRAEL DEACONESS MEDICAL CENTER    MA
## MD              JOHNS HOPKINS HOSPITAL, THE    MD
## ME                            YORK HOSPITAL    ME
## MI                    MUNSON MEDICAL CENTER    MI
## MN                        ST MARYS HOSPITAL    MN
## MO                    BOONE HOSPITAL CENTER    MO
## MS                    WESLEY MEDICAL CENTER    MS
## MT                    BENEFIS HOSPITALS INC    MT
## NC       CAROLINAS MEDICAL CENTER-NORTHEAST    NC
## ND             SANFORD MEDICAL CENTER FARGO    ND
## NE           FAITH REGIONAL HEALTH SERVICES    NE
## NH                  CATHOLIC MEDICAL CENTER    NH
## NJ             EAST ORANGE GENERAL HOSPITAL    NJ
## NM                      ST VINCENT HOSPITAL    NM
## NV      SUNRISE HOSPITAL AND MEDICAL CENTER    NV
## NY                    NYU HOSPITALS CENTER    NY
## OH                    JEWISH HOSPITAL, LLC    OH
## OK            OKLAHOMA HEART HOSPITAL SOUTH    OK
## OR               PORTLAND VA MEDICAL CENTER    OR
## PA                     DOYLESTOWN HOSPITAL    PA
## PR        HOSPITAL DR CAYETANO COLL Y TOSTE    PR
## RI                          MIRIAM HOSPITAL    RI
## SC                    MUSC MEDICAL CENTER    SC
## SD AVERA HEART HOSPITAL OF SOUTH DAKOTA LLC    SD
## TN    METHODIST MEDICAL CENTER OF OAK RIDGE    TN
## TX         CYPRESS FAIRBANKS MEDICAL CENTER    TX
## UT            DIXIE REGIONAL MEDICAL CENTER    UT
## VA       CHESAPEAKE REGIONAL MEDICAL CENTER    VA
## VI        ROY LESTER SCHNEIDER HOSPITAL,THE    VI
## VT       FLETCHER ALLEN HOSPITAL OF VERMONT    VT
## WA   PROVIDENCE SACRED HEART MEDICAL CENTER    WA
## WI                    BELLIN MEMORIAL HSPTL    WI
## WV       MONONGALIA COUNTY GENERAL HOSPITAL    WV
## WY                  WYOMING MEDICAL CENTER    WY
```

```
r <- rankall("heart attack", 4)
as.character(subset(r, State == "HI")$Hospital)
```

```
## [1] "CASTLE MEDICAL CENTER"
```

```
r <- rankall("pneumonia", "worst")
as.character(subset(r, State == "NJ")$Hospital)
```

```
## [1] "BERGEN REGIONAL MEDICAL CENTER"
```

```
r <- rankall("heart failure", 10)
as.character(subset(r, State == "NV")$Hospital)
```

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