Assignment1_Part1.R

Source on Save

```r


pollutantmean <- function(directory, pollutant, id = 1:10) {

        directory <- getwd()
        files <- list.files(directory, pattern="*.csv", full.names=FALSE)
        merged = NULL

        for (i in id) {
                data <- read.csv(file=files[i], header = TRUE)
                merged <- rbind(merged,data)
                sulfate <- subset(merged, sulfate != "NA")
                nitrate <- subset(merged, nitrate != "NA")
        }
        if(pollutant == "sulfate") {
                meanpollutant <- sum(sulfate$sulfate)/nrow(sulfate)
        }
        else if (pollutant == "nitrate") {
                meanpollutant <- sum(nitrate$nitrate)/nrow(nitrate)
        }
        return(meanpollutant)
}

```

Assignment1_Part1.R | Assignment1_Part2.R

Source on Save

```r

complete <- function(directory, id=1:332) {
        directory <- getwd()
        files <- list.files(directory, pattern="*.csv", full.names=FALSE)
        merged = NULL

        for (i in id) {
                data <- read.csv(file=files[i], header = TRUE)
                merged <- rbind(merged,data)
                complete <- subset(merged, sulfate != "NA" & nitrate != "NA")
                counts <- data.frame(table(complete$ID))
                names(counts) <- c("id","nobs")
                }
        {
        return(counts)
        }

}
```

Assignment1_Part1.R | Assignment1_Part2.R | Assignment1_Part3.R

Source on Save

```r
corr <- function(directory, threshold=0) {
        directory <- getwd()
        files <- list.files(directory, pattern="*.csv", full.names=FALSE)
        merged = NULL
        cvect = NULL

        for (i in 1:332) {
                data <- read.csv(file=files[i], header = TRUE)

                        data <- subset(data, sulfate != "NA" & nitrate != "NA")

                        if (nrow(data)>threshold){
                           cvect = c(cvect,cor(data$sulfate, data$nitrate))
                        }


        }
        {

                return(cvect)
        }
}
```

cachematrix.R

Source on Save | Run | Source

```r

## This program caches inverse of matrix and creates inverse if absent

## makeCacheMatrix: This function creates a special "matrix" object that can cache its inverse.

makeCacheMatrix <- function(x = matrix()) {

        m <- NULL
        set <- function(y) {
                x <<- y
                m <<- NULL

        }
        get <- function() x
        setinv <- function(inverse) m <<- inverse
        getinv <- function() m
        list(set=set, get=get, setinv=setinv, getinv=getinv)
}


## cacheSolve: This function computes the inverse of the special "matrix" returned by makeCacheMatr
## If the inverse has already been calculated (and the matrix has not changed), then the cachesolve

cacheSolve <- function(x, ...) {
        ## Return a matrix that is the inverse of 'x'
        m <- x$getinv()

        if (!is.null(m)) {
                message("getting cached data")
                return(m)
        }

        mx.data <- x$get()
        m <- solve(mx.data, ...)
        x$setinv(m)

        return(m)
}
```

rankhospital.R

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

        ocm <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = "Not Available")
        vstates <- unique(c(ocm$State))
        testst = grep(state, vstates)
        testout = grep(outcome, c("heart attack", "heart failure", "pneumonia") )
        #Subset required variables - smaller dataset
        cb <- cbind(ocm[2],ocm[7],ocm[11],ocm[17],ocm[23])

        #Rename variable names for easier handling
        names(cb) <- c("hname","state", "hattack", "hfailure","pneumonia")

        if(!state %in% outsub1[,"state"]) {
                stop('invalid state')
        }
        if(!outcome %in% c("heart attack", "heart failure", "pneumonia")){
                stop('invalid outcome')
        }

        if(testout == "1"){
                cb1 <- cb[, c(1,2,3)]
                cb1 <- subset(cb1, state == vstates[testst])
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                nc <- nrow(cb1)
                if(num == "best"){
                        ret <- best(state, outcome)
                }
                else if(num == "worst"){
                         num = nc
                         ord <- cb1[order(cb1$hattack, cb1$hname),]
                         rank <- rank(ord$hattack, na.last=TRUE, ties.method = "first")
                         srt <- cbind(ord,rank)
                         out <- srt[order(srt$rank, srt$hname), ]
                         ret <- out[out$rank == num, 1]
                }
                else if(num > nc){
                         stop(print("NA"))
                }
                else if(num > 0){
                         ord <- cb1[order(cb1$hattack, cb1$hname),]
                         rank <- rank(ord$hattack, na.last=TRUE, ties.method = "first")
                         srt <- cbind(ord,rank)
                         out <- srt[order(srt$rank, srt$hname), ]
                         ret <- out[out$rank == num, 1]
                }
        }
        if(testout == "2"){
                cb1 <- cb[, c(1,2,4)]
                cb1 <- subset(cb1, state == vstates[testst])
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                nc <- nrow(cb1)
                if(num == "best"){
                        ret <- best(state, outcome)
                }
                else if(num == "worst"){
                        num = nc
                        ord <- cb1[order(cb1$hfailure, cb1$hname),]
                        rank <- rank(ord$hfailure, na.last=TRUE, ties.method = "first")
                        srt <- cbind(ord,rank)
                        out <- srt[order(srt$rank, srt$hname), ]
                        ret <- out[out$rank == num, 1]
                }
                else if(num > nc){
                        stop(print("NA"))
                }
                else if(num > 0){
                        ord <- cb1[order(cb1$hfailure, cb1$hname),]
                        rank <- rank(ord$hfailure, na.last=TRUE, ties.method = "first")
                        srt <- cbind(ord,rank)
                        out <- srt[order(srt$rank, srt$hname), ]
                        ret <- out[out$rank == num, 1]
                }
        }
        if(testout == "3"){
                cb1 <- cb[, c(1,2,5)]
                cb1 <- subset(cb1, state == vstates[testst])
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                nc <- nrow(cb1)
                if(num == "best"){
                        ret <- best(state, outcome)
                }
                else if(num == "worst"){
                        num = nc
                        ord <- cb1[order(cb1$pneumonia, cb1$hname),]
                        rank <- rank(ord$pneumonia, na.last=TRUE, ties.method = "first")
                        srt <- cbind(ord,rank)
                        out <- srt[order(srt$rank, srt$hname), ]
                        ret <- out[out$rank == num, 1]
                }
                else if(num > nc){
                        stop(print("NA"))
                }
                else if(num > 0){
                        ord <- cb1[order(cb1$pneumonia, cb1$hname),]
                        rank <- rank(ord$pneumonia, na.last=TRUE, ties.method = "first")
                        srt <- cbind(ord,rank)
                        out <- srt[order(srt$rank, srt$hname), ]
                        ret <- out[out$rank == num, 1]
                }
        }
        return(ret)
}
```

rankall.R

```r
rankall <- function(outcome, num="best") {
        library(data.table)
        ocm <- read.csv("outcome-of-care-measures.csv", colClasses = "character", na.strings = "Not Available")
        testout = grep(outcome, c("heart attack", "heart failure", "pneumonia") )

        cb <- cbind(ocm[2],ocm[7],ocm[11],ocm[17],ocm[23]) #Subset with required variables
        names(cb) <- c("hname","state", "hattack", "hfailure","pneumonia") #Rename variable names

        state <- unique(cb$state)
        states <- data.frame(state, state)

        if(!outcome %in% c("heart attack", "heart failure", "pneumonia")){
                stop('invalid outcome')
        }

        if(testout == "1"){
                cb1 <- cb[, c(1,2,3)]
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                cb1 <- cb1[order(cb1$state, cb1$hname),]
                cb2 <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(hattack,ties.method = "first")))

                if(num == "best"){
                        num = 1
                        cb3 <- cb2[cb2$Order == num, ]
                }
                else if(num == "worst"){
                        num = 1
                        cbw <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(-hattack,ties.method = "first")))
                        cb3 <- cbw[cbw$Order == num, ]
                }
                else if(num > 0){
                        cb3 <- cb2[cb2$Order == num, ]
                }

        }
        if(testout == "2"){
                cb1 <- cb[, c(1,2,4)]
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                cb1 <- cb1[order(cb1$state, cb1$hname),]
                cb2 <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(hfailure,ties.method = "first")))

                if(num == "best"){
                        num = 1
                        cb3 <- cb2[cb2$Order == num, ]
                }
                else if(num == "worst"){
                        num = 1
                        cbw <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(-hfailure,ties.method = "first")))
                        cb3 <- cbw[cbw$Order == num, ]
                }
                else if(num > 0){
                        cb3 <- cb2[cb2$Order == num, ]
                }

        }
        if(testout == "3"){
                cb1 <- cb[, c(1,2,5)]
                cb1[,3] <- suppresswarnings(as.numeric(cb1[,3]))
                cb1 <- na.omit(cb1)
                cb1 <- cb1[order(cb1$state, cb1$hname),]
                cb2 <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(pneumonia,ties.method = "first")))

                if(num == "best"){
                        num = 1
                        cb3 <- cb2[cb2$Order == num, ]
                }
                else if(num == "worst"){
                        num = 1
                        cbw <- do.call(rbind,lapply(split(cb1,cb1$state),transform, Order = rank(-pneumonia,ties.method = "first")))
                        cb3 <- cbw[cbw$Order == num, ]
                }
                else if(num > 0){
                        cb3 <- cb2[cb2$Order == num, ]
                }

        }
        cb4 <- merge(x=cb3, y=states, by = "state", all = TRUE)
        cb5 <- cb4[order(cb4$state, cb4$hname, cb4$Order),]

        return(cb5[,2:1])
}
```