**Given a invertible matrix, the following two functions will calculate the inverse matrix or retrieve the inverse matrix from the cache.**

Function “makeCacheMatrix” creates a special “matrix” object that can cache its inverse. makeCacheMatrix contains 4 functions: set, get, setmean, getmean.

(1)get is a function that returns the vector x stored in the main function.  
(2)set is a function that changes the vector stored in the main function.  
(3)setmean and getmean are functions very similar to set and get.  
(4)They don’t calculate the mean, they simply store the value of the input in a variable m.  
(5)into the main function makeVector (setmean) and return it (getmean).

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

m <- NULL

set <- function(y) {

x <<- y

m <<- NULL

}

get <- function() x

setinverse <- function(solve) m <<- solve

getinverse <- function() m

list(set = set, get = get,

setinverse = setinverse,

getinverse = getinverse)

}

Function “cacheSolve” computes the inverse of the special “matrix” (which is the input of cachemean) returned by makeCacheMatrix above. If the inverse has already been calculated (and the matrix has not changed), then the cachesolve should retrieve the inverse from the cache. If the inverse has not been calculated, data gets the matrix stored with makeCacheMatrix, m calculates the inverse, and x$setmean(m) stores it in the object m in makeCacheMatrix.

cacheSolve <- function(x, ...) {

m <- x$getinverse()

if(!is.null(m)) {

message("getting cached data")

return(m)

}

data <- x$get()

m <- solve(data, ...)

x$setinverse(m)

m

}