3. How many rows in the data?4. How many columns in the data?5. What are the column names?

6. Do a statistical summary of the data7. Average miles per gallon (mpg) of all carsmean(miles per gallon of all cars)

8. Average mpg of automatic transmission carsmean(mpg of automatic transmission cars)

9. Average mpg of manual transmission cars10. Average Displacement of cars with 4 gears11. Average Horse power of cars with 3 carb12. Average mpg of automatic cars with 4 gearsWhat is gearwise average mpg (avg mpg of for 3, 4 and 5 gear cars). Save result in a named vector

# Practice on mtcars dataView(mtcars)# 1. How many rows in the data?dim(mtcars)nrow(mtcars)# 2. How many columns in the data?ncol(mtcars)# 3. What are the column names?colnames(mtcars)# 4. Do a statistical summary of the datasummary(mtcars)attach(mtcars)# 5. Average miles per gallon (mpg) of all carsmean(mpg)mean(mtcars$mpg)# 6. Average mpg of automatic transmission carsfilter\_condn = am == 0mean(mtcars[filter\_condn,"mpg"])# 7. Average mpg of manual transmission carsfilter\_condn = am == 1mean(mtcars[filter\_condn,"mpg"])# 8. Average Displacement of cars with 4 gearsfilter\_condn = gear == 4mean(mtcars[filter\_condn,"disp"])# 9. Average Horse power of cars with 3 carbfilter\_condn = carb == 3mean(mtcars[filter\_condn,"hp"])# 10. Average mpg of automatic cars with 4 gearsfilter\_condn1 = am == 0filter\_condn2 = gear == 4mean(mtcars[filter\_condn1 & filter\_condn2,"mpg"])

# using apply group of functtions# apply (row wise or column wise operation)# by (filtering and repeating an operation on each filter output; can be done on multiple columns); aggregate# lapply (columns wise or list element wise operation)# sapply (columns wise or list element wise operation. output is simplified as a vector/matrix)# tapply (filtering and repeating on each filter output; can work only on one filtered output)# eapply (repeats on elements in environment)# mapply (repats function in a vectorized wat)# rapply (recursive)m <- matrix(c(1:10,11:20), nrow=5, ncol=4)mean(m[,1])mean(m[,2])mean(m[,3])mean(m[,4])mean(m[1,])mean(m[2,])mean(m[3,])mean(m[4,])mean(m[5,])# compute average for all the columns apply(m,2,mean)apply(m,2,median)apply(m,2,sum)apply(m,2,sd)# compute average for all the rowsapply(m,1,mean)apply(m,1,median)by(mtcars$mpg,mtcars$gear,mean)# cylinder wise average hpby(mtcars$hp,mtcars$cyl,mean)#compute the aveerage of all columns by cylnder typeby(mtcars[,-2], mtcars$cyl, colMeans)#by function for categorical variablesby(mtcars[,c(1,5)], factor(mtcars$cyl), colMeans)

mylist<- as.list(mtcars)lapply(mylist, median)# sapply - similar to vapply# sapply takes input as list/matrix and output is a vectorsapply(mylist, median)# vapplysapply(mylist,fivenum)vapply(mylist, fivenum, c(min=0,'Q1'=0,median =0, 'Q3'=0, max=0))# replicate functionreplicate(10, rnorm(5, 100, 2), simplify=F)replicate(10, rnorm(5, 100, 2), simplify=T)#mapplyrep(5,10) # repeat 5 10 timesrep(6,10)rep(7,10)rep(8,10)mapply(rep, 5:8, 10)mapply(rep, times = 1:4, x = 4:1)l1 <- list(mtcars[,1:4])str(l1)l2 <- list(mtcars[,3:8])str(l2)# sum of corresponding elements captured in L1 and L2mapply(sum, x=l1,y=l2)# rapplyrapply(mtcars[,1:3], log2, how='list')#tapply functiontapply(mtcars$mpg, mtcars$cyl, mean)