1) library(xtable)
library(datasets)
data<-USJudgeRatings[1:5,1:5]
print(xtable(data),type="html")
?print
data[data$CONT<7.3,c("INTG","DMNR")]

data_2<- data[(data$x1>0)&(data$y1<0)&(data$gender=="M"),]
data_2

setwd("_")
data_coefs<-cbind(model_1$coefficients,model_2$coefficients)
data_coefs<-as.data.frame(data_coefs)
write.csv2(data_coefs,"data2.csv",row.names=FALSE)

2) setwd("_")
list.files()
list_data <- vector('list',100)

list_data[[1]] <- read.csv('data_1.csv')
list_data[[1]]
i=1

for (i in 1:100){
  list_data[[i]] <- read.csv(list.files()[i])
}

matrix_results <- matrix (nrow= length(list.files()),ncol = 2)

lm(y~x, list_data[[1]])$coefficients

for (i in 1:length(list.files())){
  matrix_results[i,] <- lm(y~x, list_data[[i]])$coefficients
}
matrix_results[1,2]

list_reg <- vector('list', length(list.files()))

for (i in 1:length(list.files())){
  list_reg[[i]] <- lm(y~x, list_data[[i]])$coefficients
}
list_reg

do.call(rbind, list_reg)

data_1<-data.frame(x=rnorm(1000,0,1),

```
            y=rnorm(1000,1,2),
            z=rnorm(1000,2,3),
            cat_1=sample(c("A","B","C"),1000,replace=TRUE),
            cat_2=sample(c("A","B","C"),1000,replace=TRUE))
vec_1<-c(mean(data_1$x),mean(data_1$y),mean(data_1$z))
vec_2<-c(var(data_1$x),var(data_1$y),var(data_1$z))
vec_1
vec_2
summary(data_1$x)
summary(data_1$y)
summary(data_1$z)
```

```
hist(data_1$x,breaks=seq(floor(min(data_1$x)), #minimum rounded down
                 ceiling(max(data_1$x)),  #maximum rounded up
                 0.5 ),
   xlim=c(-5,10),ylim=c(-1,200),
   xlab="My variable",ylab="Count", main="Some histogram"
)
```

```
data_1$cat_1 %in% c("A") #specifies whether an element is in the data frame (if this %in% there)
which(data_1$cat_1 %in% c("A")) #logical variable under which
```

3)

```
# 1. Create vectors to store the trajectory.
x<-vector()
y<-vector()
# 2. Set starting conditions.
x[1]<-0
y[1]<-0
i<-1
diff<-1 # placeholder, as 1 > 10^(-16)
# Iterate untl the system converges (or times run out)
while (diff > 10^(-16)) {
  # a) mind the trajectory
  i<-i+1
  # b) calculate x and y
  x[i] <- x[i-1]*y[i-1]+x[i-1]+0.07
  y[i] <- x[i-1]^2+ y[i-1]^2+y[i-1]-0.41
  # c) calculate diff
  diff<- max(
    abs(x[i]-x[i-1]), # change in x
    abs(y[i]-y[i-1]))   # change in y
```

```
  # check for time out
  if (i >1000) { break }
}
length(x)
plot(y=x,x = 1:41, type='l')



4)   #1
data_1<-fread('data_1.csv')
data_2<-fread('data_2.csv')
head(data_1)
unique(data_1[is.na(as.numeric(wage)), wage])
unique(data_2[is.na(as.numeric(hours)), hours])

#2 Convert wage & hours to numeric
data_1[, wage := as.numeric(wage)]
data_2[, hours := as.numeric(hours)]



# 2
vec<-c(1,0,0,1,0,0,0,1,1,0)

f2_zeros_index<-function(x) {
  aux<-rle(x)
  max_length<-max(aux$lengths[aux$values == 0])
  location<- min(which(aux$values == 0 & aux$lengths == max_length))
  if (location==1) {
    return(1)
  }
  else {return(sum(aux$lenths[1:(location-1)])+1)}
}

f2_zeros_index(vec)



# 3
f3_simulation <-function(n,x_mean,x_sd, eps_mean,eps_sd,alpha,beta,n_sim,...) {
# Create a matrix (n_sim x 2) to store the coefficients.
  coeff_mat <-matrix(nrow=n_sim, ncol=2)
  dim(matrix)
# Run n_sim simulations:
  for (i in 1:n_sim){
# a) generate x, eps and y
    x <- rnorm(n,x_mean,x_sd)
    eps <- rnorm(n,eps_mean,eps_sd)
    y <- alpha+beta*x+eps
# b) estimate the model
    res<- lm(y~x)
```

```
# c) store the coeffs in the matrix
   coeff_mat[i, ]<-res$coefficients
 }
 # Plot the coeffs distribution
 par(mfrow=c(1,2))  # the way we plot to graphs in an output
 hist(coeff_mat[, 1],xlab="alpha",main="Hist of alpha",...)
 hist(coeff_mat[, 2],xlab="beta",main="Hist of betas",...)
}

f3_simulation(100,10,3,0,2,3,-2,200,breaks=5)



5)  # Problem set 5
library(XML)
library(stringr)
library(data.table)

#1
files_to_loop<-list.files("./TWITTER_DATA/")
files_to_loop[1:6]
unique(do.call(rbind, str_split(files_to_loop, "\\."))[, 1]) # remember to use \ to . and \ to \ itself

#2
economists <- data.table(
 people =  do.call(rbind, str_split(files_to_loop, "\\."))[, 1],
 file_names = files_to_loop)
economists[, .N, by = c("people")]   #.N - number of rows; specify by "by" - row number of a group;
witout by - rows in the whole data
economists

#3
#var 1.
dinapomeranz<- economists[people=="dinapomeranz", file_names]
dinapomeranz

#4
tweets<- readHTMLTable("./TWITTER_DATA/dinapomeranz.100.html")
str(tweets[1:6])

#5
# identical(tweets[[2]], NULL)
tweets <- tweets[!sapply(tweets, identical, NULL)]
tweets <- tweets[-1] # Run this just once
tweets

#6
retweets<-tweets[sapply(tweets, dim)[1, ] == 3]
f_extract <- function(df, r, c) { return(df[r, c]) }
# f_extract(tweets[[1]], 1, 2)
```

```
people <- sapply(retweets, f_extract, 1, 2)
dates <- sapply(retweets, f_extract, 1, 3) # replace 2 column with third for dates
data_retweets<- data.table(people=people, dates=dates)
data_retweets[,
  people := str_trim(gsub("@(.*)|\\n","",people))
]
data_retweets

#7
hashtags <- list("vector")
for (i in 1:length(tweets)) {
  # Collapse all info from tweet to a string
  tweet <- paste(unlist(tweets[[i]]), collapse = " ")
  # Match a regular expression
  hashtags[[i]] <- str_match_all(tweet, "#[A-Za-z0-9]+") #+ - character could be repeated
}
hashtags<-unlist(hashtags) # unlist - make from a list one vector
hashtags

#8
tnames<- list("vector")
for (i in 1:length(tweets)) {
  # Collapse all info from tweet to a string
  ntweet <- paste(unlist(tweets[[i]]), collapse = " ")
  # Match a regular expression
  tnames[[i]] <- str_match_all(ntweet, "@[A-Za-z0-9_]+") #+ - character could be repeated
}
tnames<-unlist(tnames)
tnames
```

```
6)  library(ggplot2)
library(data.table)
#Class code
#Use in built functions to create the plot you want:
ggplot(data = diamonds) +
  geom_bar(mapping = aes(x = cut))

#Create the table to plot:
#Notice the identity argument which you are using in this plot.
table_to_plot<-as.data.table(diamonds)
table_to_plot<-table_to_plot[,length(carat),by=c("cut")]
setnames(table_to_plot,c("cut","count"))
ggplot(table_to_plot)+geom_bar(mapping=aes(x=cut,y=count),stat="identity")
```

```
library(ggplot2)
```

```
# Class code
#
#

# engine displacement, in litres
# highway miles per gallon

#var 1
(ggplot(data=mpg)
  +geom_point(mapping=aes(x=displ,y=hwy))
  +geom_smooth(mapping=aes(x=displ,y=hwy)))

#var 2 - разница в mapping
(ggplot(data=mpg, mapping=aes(x=displ,y=hwy))
  +geom_point(mapping=aes(color=class)) # color class нельзя добавлять в общее условие
  +geom_smooth() # иначе построение идет по каждому классу из-за geom_smooth
  +xlab("engine displacement, in litres")
  +ylab("highway miles per gallon")
  +ggtitle("Dependency on fuel cons. on engine volume")
  +theme_bw()
  )# mpg - built-in source

#var 3 - more sutable for black/white printing
(ggplot(data=mpg, mapping=aes(x=displ,y=hwy))
  +geom_point()
  +xlab("engine displacement, in litres")
  +ylab("highway miles per gallon")
  +ggtitle("Dependency on fuel cons. on engine volume")
  +theme_bw()
  +facet_wrap(~ class, nrow=2)
  )


# Another example
(ggplot(data=mpg)
  +aes(x=class)
  + geom_bar()
)

7)   library(ISLR)
model_1 <- lm(mpg~horsepower, data= Auto)
summary(model_1)

predict(model_1, data.frame(horsepower=c(98)), interval='confidence')
predict(model_1, data.frame(horsepower=c(98)), interval='prediction')
# conf уже pred, так как s.e. разные

plot(Auto$horsepower,Auto$mpg)
abline(model_1, col="red") # add a line
```