```
################################################################################################
####

#3B-1

#Bird.data loading the file Birdstrikes.csv into it

setwd("C:/Users/Michael Propp/Desktop/Information Assignemnt/Ass-2/")


Bird.data <- read.csv("Bird Strikes.csv", header = TRUE , sep = ",",quote = "\"",fill = TRUE)


#Defining the function bird.data.notreported

#bird.data.notreported :Input bird.data.reported

#leastInvInterval takes a particular bird.data.reported and returns

#bird strikes were not reported

#Output : total number of bird strikes were not reported


bird.data.notreported <- function(bird.data.reported){
  bird.reported <- bird.data.reported
  bird.strikes.noreport <- sum(bird.reported$Reported=="")
  cat("The bird striked which were not reported ", "\"")
  print(bird.strikes.noreport)
}


#bird.data.reported(Bird.data)

bird.data.notreported(Bird.data)
```

```
##########################################################################################
##########

#3B-2

#Defining the function Maximumyear.birdstrikes
#Maximumyear.birdstrikes :Input Flight.Date
#Maximumyear.birdstrikes takes a particular Flight.Date and returns
#Which year had the most bird strikes
#Output : The year which had the most bird strikes

Maximumyear.birdstrikes <- function(Flight.Date){
  flightdata<- Flight.Date
  flightdate.data <- as.Date(flightdata$FlightDate,format= "%m/%d/%Y")
  flightdate.data.table <- table(format(flightdate.data,"%Y"))
  flight.sorted <- sort(flightdate.data.table,descending = T)
  cat("The year which had the most bird strikes is","\"")
  print(flight.sorted[12])
}
Maximumyear.birdstrikes(Bird.data)
```

```
################################################################################################
####

#3B- 3 ( As.data.frame)


#Defining the function Bird.strikes.dataframe

#Bird.strikes.dataframe :Input Bird.strikes.maximum

#Bird.strikes.dataframe takes a particular Bird.strikes.maximum

#Bird.data maximum takes Bird.strikes.maximum and converts it into a dataframe

#Output : The class of Bird.data.maximum is printed


Bird.strikes.dataframe <- function(Bird.strikes.maximum){
  Bird.maximum.death <- Bird.strikes.maximum
  Bird.data.maximum <- as.Date(Bird.maximum.death$FlightDate ,format="%m/%d/%Y")
  Bird.data.maximum <- data.frame(table(format(Bird.data.maximum,"%Y")))
  cat("The class of Bird.data.maximum is","\"")
  print(class(Bird.data.maximum))
}
Bird.strikes.dataframe(Bird.data)
```

```
#############################################################################################
#3B-4
#Defining the function Airlines.birdstrikes.data
#Airlines.birdstrikes.data :Input Bird.strikes.maximum
#Airlines.birdstrikes.data takes a particular Airlines.birdstrikes
#airline.data takes the 15th column of airlines.data
#Output : The class of Bird.data.maximum is printed
#Maximum.birdstrikes(airline.data) is calling of data airline.data into maximum.bridstrikes
#which is defined later on
Airlines.birdstrikes.data <- function(Airlines.birdstrikes){
  airlines.data <- Airlines.birdstrikes
  airline.data <- table(airlines.data[15])
  #print(airline.data)
  maximum.birdstrikes(airline.data)
}

#Defining the function maximum.birdstrikes
#maximum.birdstrikes : Input airline data
#airline data converts airline.data into data frame
#airline.data.sorted sorts airline data in order of frequency in decreasing order
#maximum.airline .data.stored takes airline.data.sorted and and prints the second maximum value

maximum.birdstrikes<- function(airline.data){
  airline.data <- as.data.frame(airline.data)
  View(airline.data)
  airline.data.sorted <- airline.data[with(airline.data, order(-Freq)), ]
  maximum.airline.data.sorted <- airline.data.sorted[2,]
  cat("The airline which is not unknown and has the maximum bird strikes is Military","\"")
  print(maximum.airline.data.sorted[1,2])
```

}

Airlines.birdstrikes.data(Bird.data)

###

#Assignment 3B – 5th answer

#Complexity is a way to measure time and space required for an algorithm or program to execute.

#It is a time versus input size n. The time required to perform a particular function depends on various factors like speed of processor, disk speed, brand of compiler, type of CPU, Operating system, RAM etc.

#Calculation time

#Functions used in assignment 3B vs Graphs drawn based on them

#For “bird.data.notreported” function (First function)

By seeing the regression line we can conclude that,

#Calculation Time by function “bird.data.notreported” increases linearly with size of input.

#So, the time complexity for this function is O(n)

#For “Maximumyear.birdstrikes” function(Second function)

#By seeing the regression line we can conclude that,

#Calculation Time by function “Maximumyear.birdstrikes” increases linearly with size of input. There is a slightest low increase compare to the linear. But that can be consider as a linear increase.

#So, the time complexity for this function is O(n)

#For “Bird.strikes.dataframe” function

#By seeing the regression line we can conclude that,

#Calculation Time by function “Bird.strikes.dataframe” increases linearly with size of input. There is a slightest low increase compare to the linear. But that can be consider as a linear increase.

#So, the time complexity for this function is O(n)

#For “Airlines.birdstrikes.data” function

#By seeing the regression line we can conclude that,

#Calculation Time by function “Airlines.birdstrikes.data” increases linearly with size of input.

#So, the time complexity for this function is O(n)

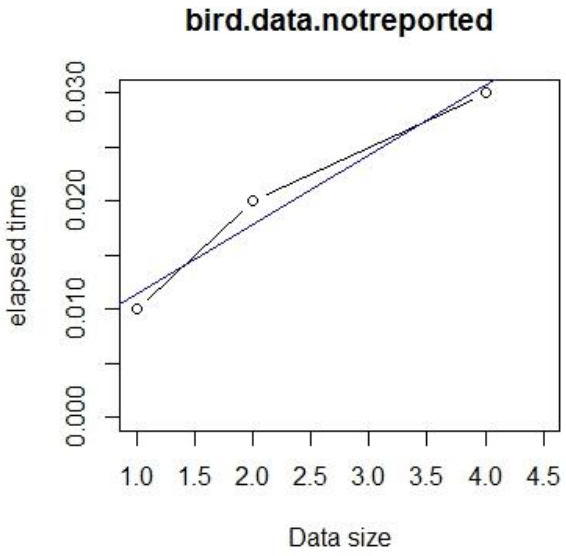
#By observing all the 4 functions and by visualization with their respective graphs with addition of their regression line we can say that they are linear with each other.

#Even the dataset were 2x, 10x, 100x, 1000x bigger than now then only the function will follow the same kind of the linearity and the complexity will remain O(n). More or less

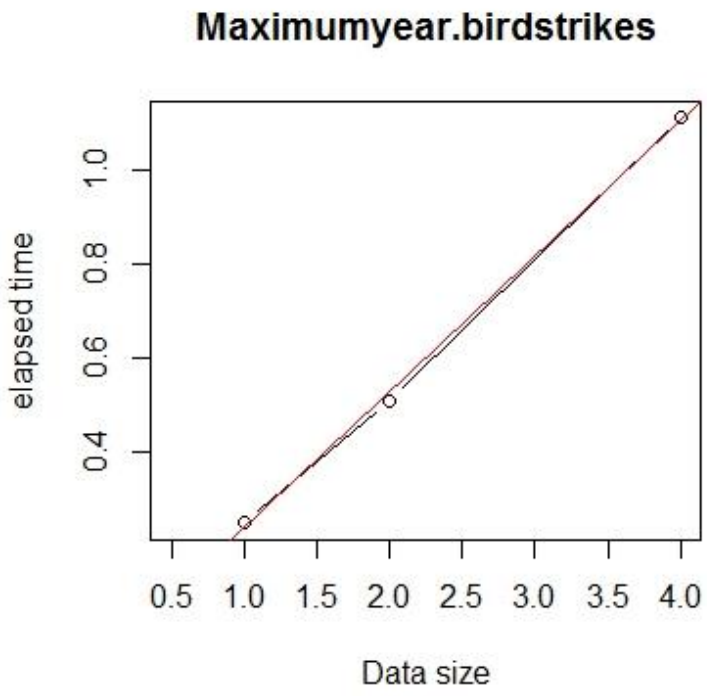
#On various trials I have noticed that, more or less the values are reported to be linear, we can assume that the system has shown its effects when the data has been increased by a significant amount like 10X or 20X (did not try the 50X or 100X)

#Also the efficiency of the function also shows it effects on larger data sets, comparing it to a very normal code which doesn't have include usage of functions or loops we can see that it takes more time to run a larger data set.

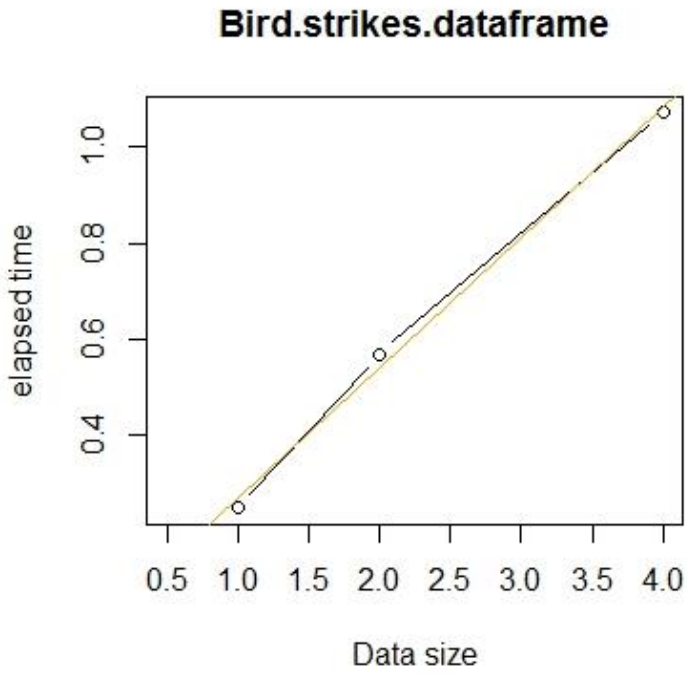
Function-1

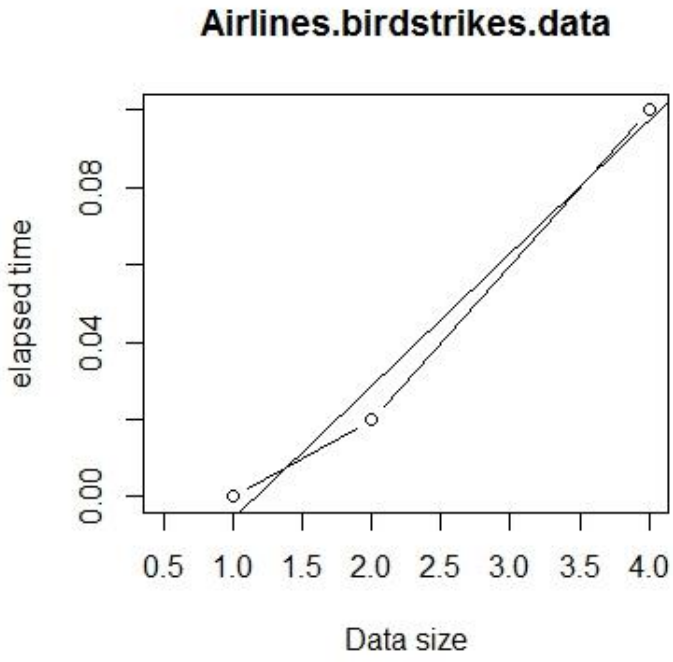


Function-2


Maximumyear.birdstrikes
elapsed time
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size

Function-3


Bird.strikes.dataframe
elapsed time
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size

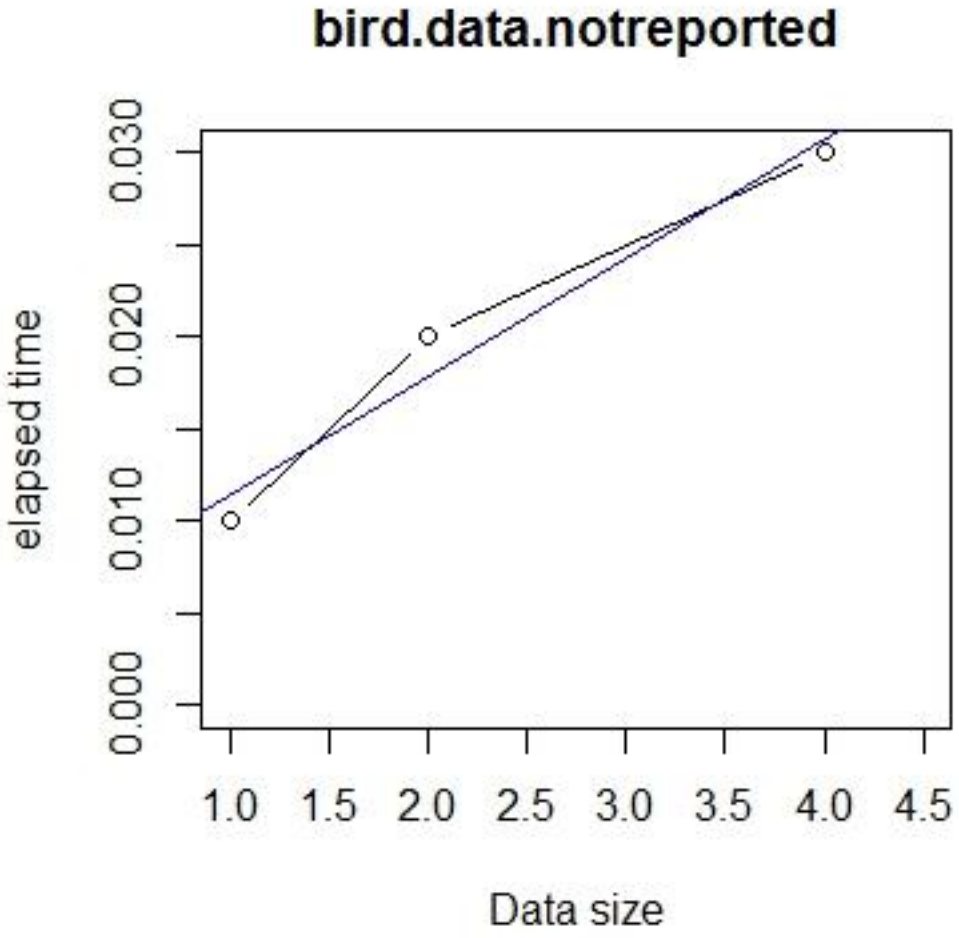

Airlines.birdstrikes.data
elapsed time
0.00
0.04
0.08
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size

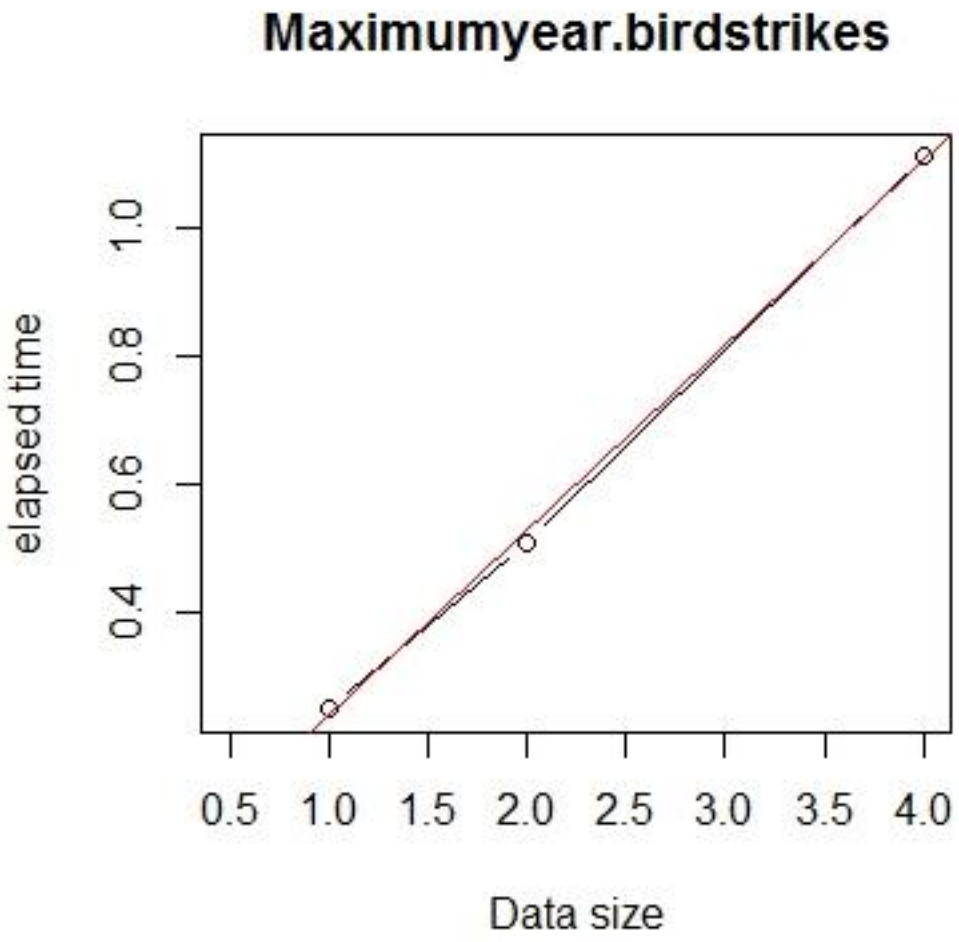
Function-4

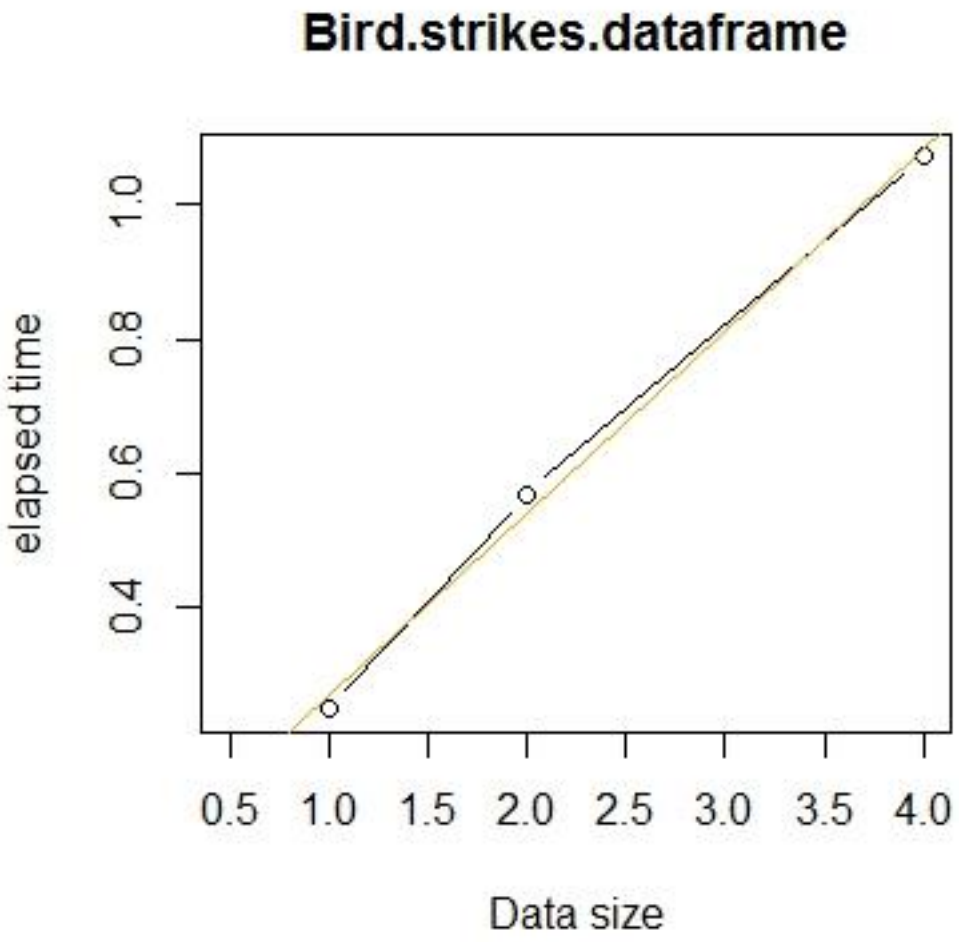
```
################################################################################################
########################
#3B-6
#Defining the 4 functions used
#double : Making a data frame which consists of 2 data sets of Bird.data, with rbind function
#Quad : Making a data frame which consists of 2 data sets of double data, with rbid function
#dataflies <- Listing the 3 dataframes
#time : Creating an empty matrix
#norw is equal to the number of functions we are using which is 4
#ncol is equal to the length of the datafile which is 3
#writing a for loop
# it runs each function for 3 times each time taking the a particular data file
# from the list of datafiles and
#puts them into each column and row in the matrix
# View time gives us the output of the matrix
# data size is creating a vector of values 1,2 and 4
#function 1 is drawing a plot taking the first row of matrix time with vector data size
#function 2 is drawing a plot taking the second row of matrix time with vector data size
#function 3 is drawing a plot taking the third row of matrix time with vector data size
#function 4 is drawing a plot taking the fourth  row of matrix time with vector data size

bird.data.notreported()
Maximumyear.birdstrikes()
Bird.strikes.dataframe()
Airlines.birdstrikes.data()

double<-rbind(Bird.data,Bird.data)
quad<-rbind(double,double)
```

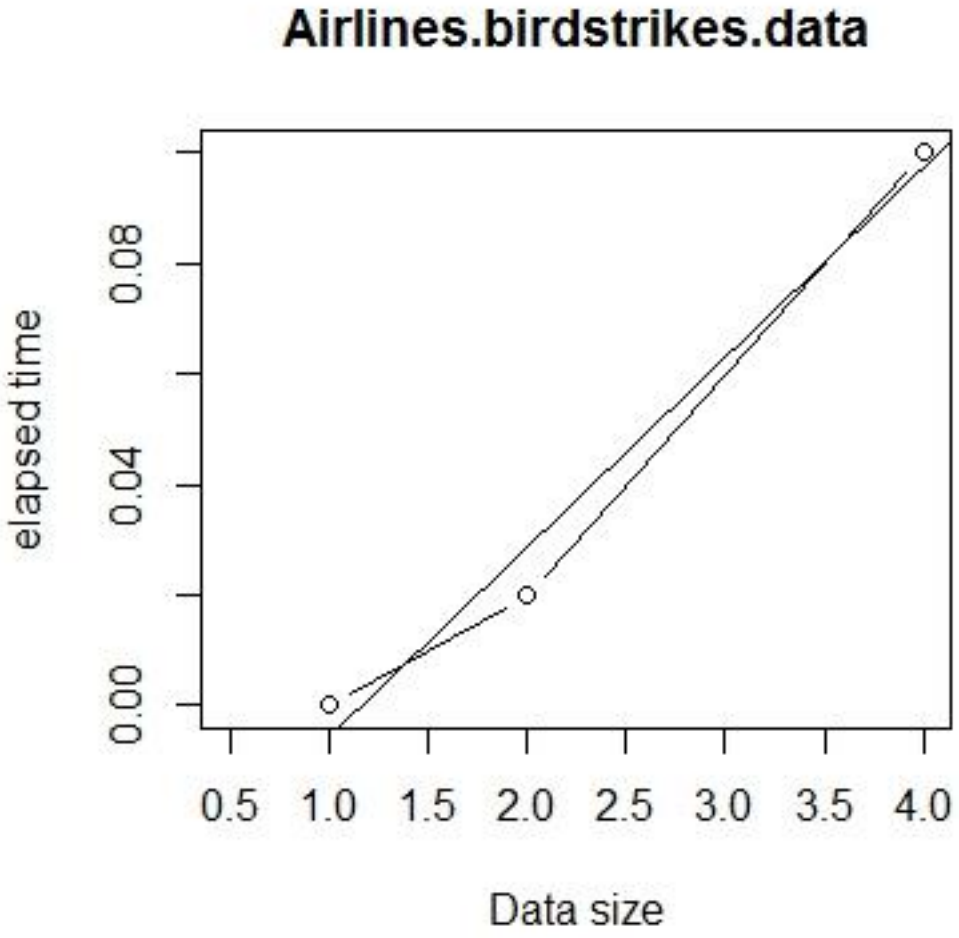
```
datafile<-list(Bird.data,double,quad)

time<-matrix(nrow=4,ncol =length(datafile))

for(i in 1: length(datafile)){
  time[1,i] <-system.time(bird.data.notreported(datafile[[i]]))[3]
  time[2,i] <-system.time(Maximumyear.birdstrikes(datafile[[i]]))[3]
  time[3,i] <-system.time(Bird.strikes.dataframe(datafile[[i]]))[3]
  time[4,i] <-system.time(Airlines.birdstrikes.data(datafile[[i]]))[3]
  print(time)
}

View(time)

data_size = c(1,2,4)

first.function<-plot(data_size,time[1,],type="b",xlim = c(1,4.5), ylim = c(0.00001,0.03), xlab = "Data size",
ylab = "elapsed time",main = "bird.data.notreported" )

abline(lm(time[1,]~data_size), col="blue")


second.function<-plot(data_size,time[2,],type="b",xlim = c(0.5,4.0), ylim = c(0.25,1.11), xlab = "Data
size", ylab = "elapsed time",main = "Maximumyear.birdstrikes")
abline(lm(time[2,]~data_size), col="red")

third.function<-plot(data_size,time[3,],type="b",xlim = c(0.5,4.0),
                     ylim = c(0.25,1.07), xlab = "Data size", ylab = "elapsed time",main = "Bird.strikes.dataframe" )
abline(lm(time[3,]~data_size), col="orange")
```

```
fourth.function<-plot(data_size,time[4,],type="b",xlim = c(0.5,4.0), ylim = c(0.00,0.1),
                      xlab = "Data size", ylab = "elapsed time", main = "Airlines.birdstrikes.data" )
abline(lm(time[4,]~data_size), col="black")
```


bird.data.notreported
elapsed time
0.000
0.010
0.020
0.030
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
Data size


Maximumyear.birdstrikes
elapsed time
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size


Bird.strikes.dataframe
elapsed time
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size


Airlines.birdstrikes.data
elapsed time
0.00
0.04
0.08
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Data size