Assignment03

Derek Olson
11/11/2019

#This section measures the time it takes to the sort function we wrote to sort increasing sizes of words
##
##################################

#Read in the data

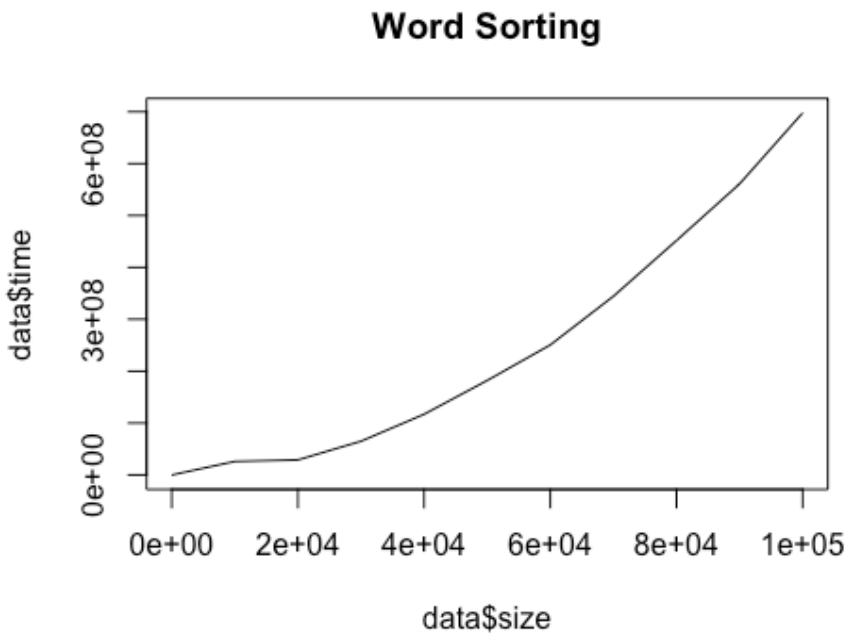
```
library(readr)
data <- read_csv("DerekOlson/CS6012/assignment03/word_sorting.csv", col_types = cols(X3 = col_skip(), X4 = col_skip(), X5 = col_skip(), X6 = col_skip(), X7 = col_skip()))
```

#Add column names

```
colnames(data)<-c("size", "time")
```

#Plot the word sorting times

```
plot(data$size, data$time, type="l")
title("Word Sorting")
```

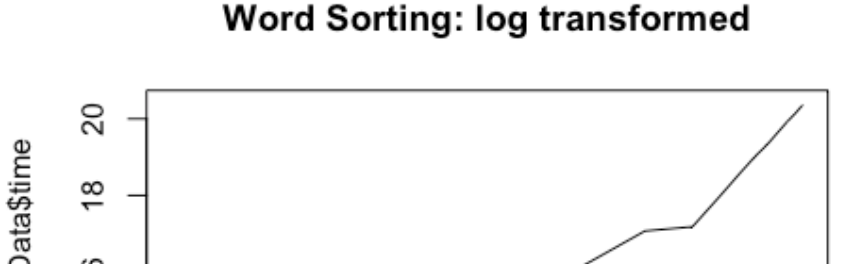


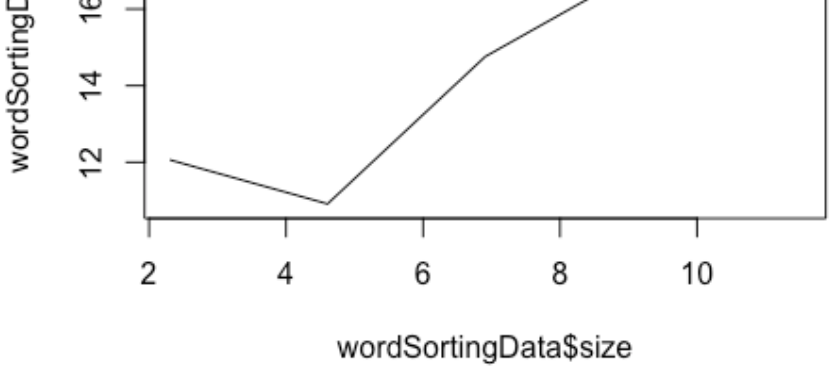
#Log transfrom the data

```
wordSortingData = log(data)
```

#Plot the log transformed data

```
plot(wordSortingData$size, wordSortingData$time, type = "l")
title("Word Sorting: log transformed")
```





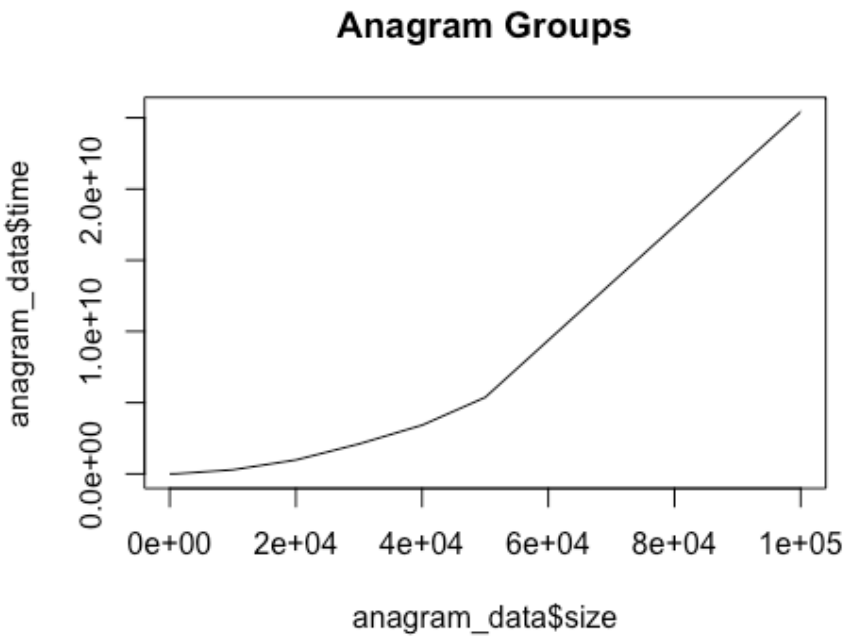

```
##############################################################################
########################################## ##This section measures the time it takes our method to get the largest anagram group with increasing array sizes
##############################################################################
##########################################
```

#Read in the data and rename the columns

```
anagram_data <- read_csv("DerekOlson/CS6012/assignment03/anagrams.csv", col_types = cols(X3 = col_skip(), X4 = col_skip(), X5 = col_skip(), X6 = col_skip(), X7 = col_skip()))
colnames(anagram_data)<-c("size", "time")
```

#Plot the times to get the largest anagram groups

```
plot(anagram_data$size, anagram_data$time, type="l")
title("Anagram Groups")
```

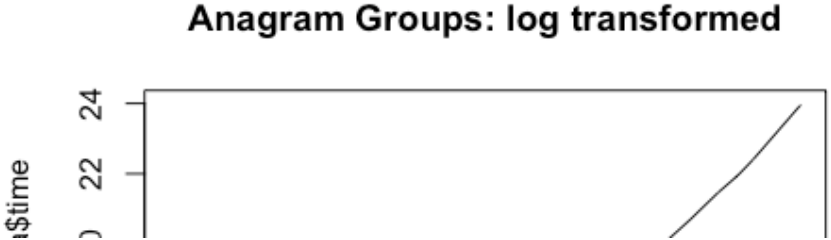


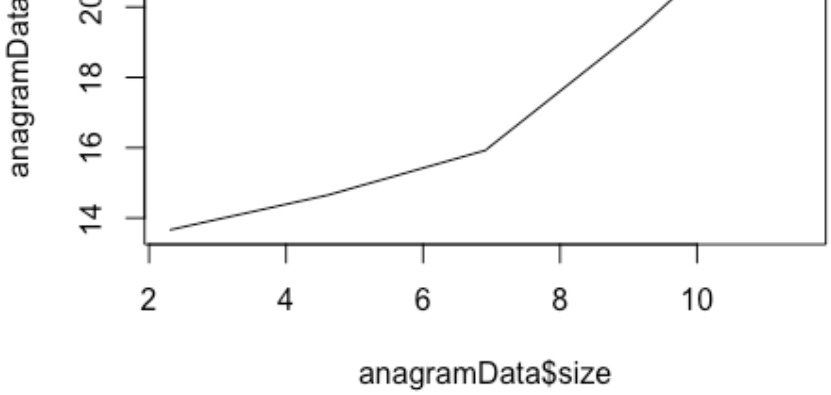
#Log transform the data

```
anagramData = log(anagram_data)
```

#Plot the log transformed data

```
plot(anagramData$size, anagramData$time, type="l")
title("Anagram Groups: log transformed")
```





 # -What is the Big-O behavior and why? Be sure to define N.
 # The Big-O behavior for getting the largest anagram groups is N^2. This is due to the fact that it relies on the
 # insertion method which is N^2.
 # -Does the growth rate of the plotted running times match the Big-O behavior you predicted?
 # Yes, there is an obvious quadratic curve.

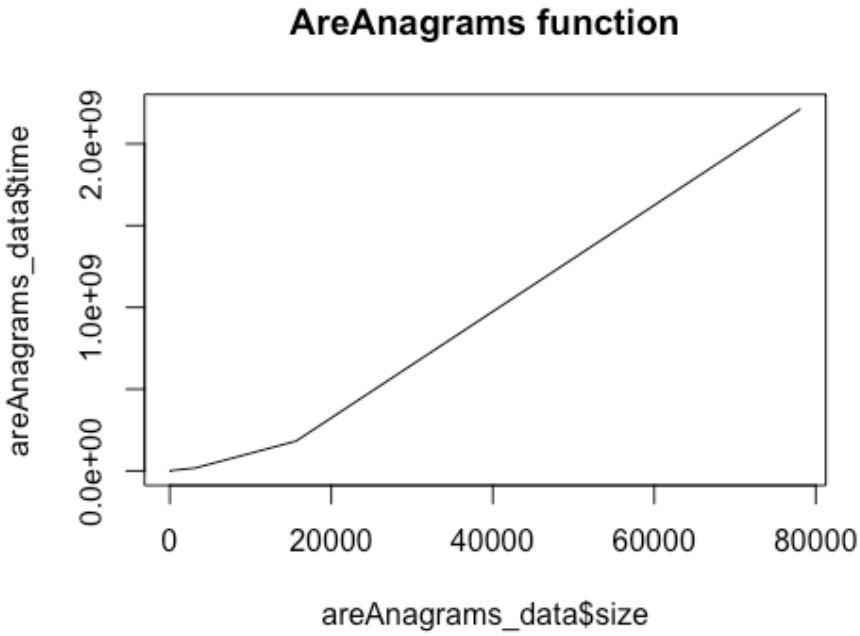
##This section measures the time it takes for our areAnagram method to compare strings of increasingly larger sizes
##

#Read in the data

```
areAnagrams_data <- read_csv("DerekOlson/CS6012/assignment03/areAnagrams.csv", col_types = cols(X3 = col_skip(), X4 = col_skip(), X5 = col_skip(), X6 = col_skip(), X7 = col_skip()))
colnames(areAnagrams_data)<-c("size", "time")
```

#Plot areAnagrams times

```
plot(areAnagrams_data$size, areAnagrams_data$time, type="l")
title("AreAnagrams function")
```

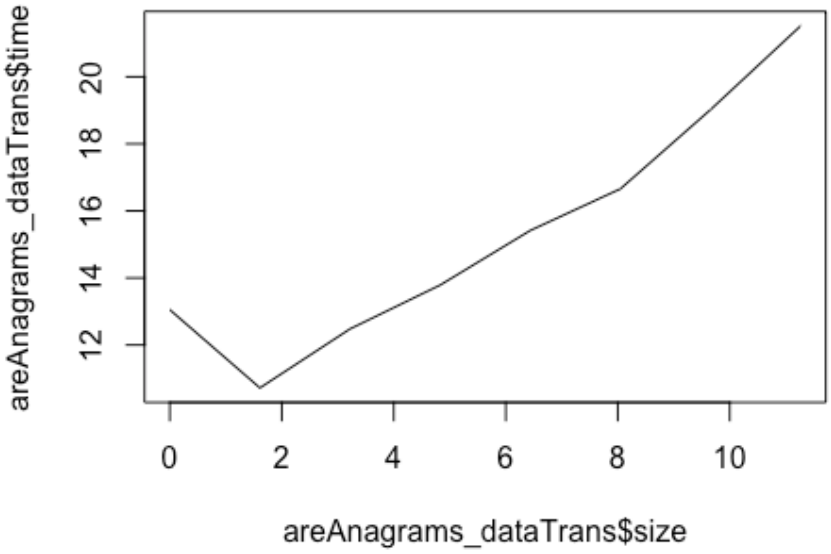


#Log transform the data

```
areAnagrams_dataTrans = log(areAnagrams_data)
```

#Plot the trnasformed data

```
plot(areAnagrams_dataTrans$size, areAnagrams_dataTrans$time, type = "l")
title("AreAnagrams function: log transformed")
```




```
#   -What is the Big-O behavior and why? Be sure to define N.
# The Big-O behavior for the areAnagams function is N^2. This is due to the fact that it relies on
the
# insertion method which is N^2.
#   -Does the growth rate of the plotted running times match the Big-O behavior you predicted?
# Yes, there is an obvious quadratic curve.
```

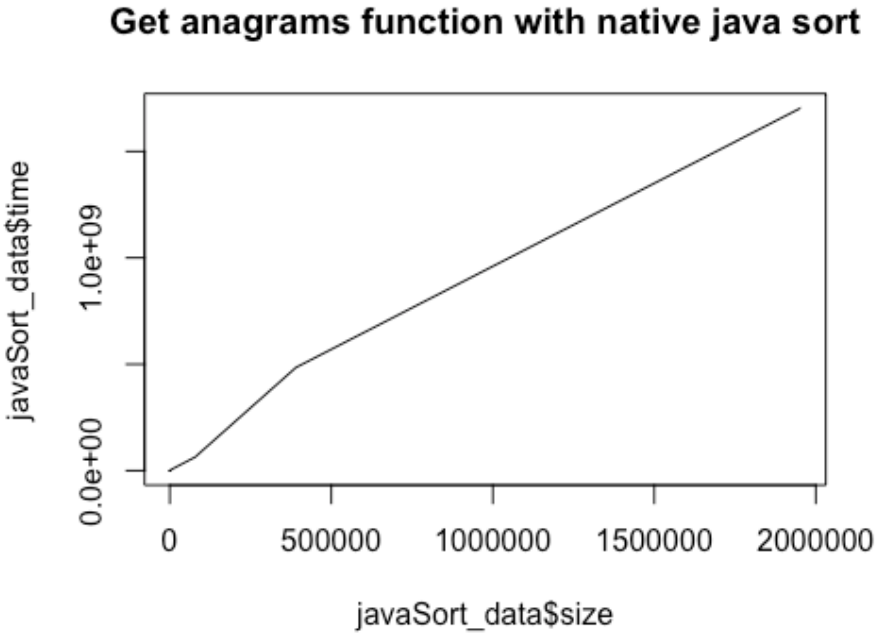
##This section measures the time that it takes to get groups of anagrams with Java's native sort
##

#Read in the data

```
javaSort_data <- read_csv("DerekOlson/CS6012/assignment03/javaSort.csv", col_types = cols(X3 = col_skip(), X4 = col_skip(), X5 = col_skip(), X6 = col_skip(), X7 = col_skip()))
colnames(javaSort_data)<-c("size", "time")
```

#Plot the data

```
plot(javaSort_data$size, javaSort_data$time, type="l")
title("Get anagrams function with native java sort")
```



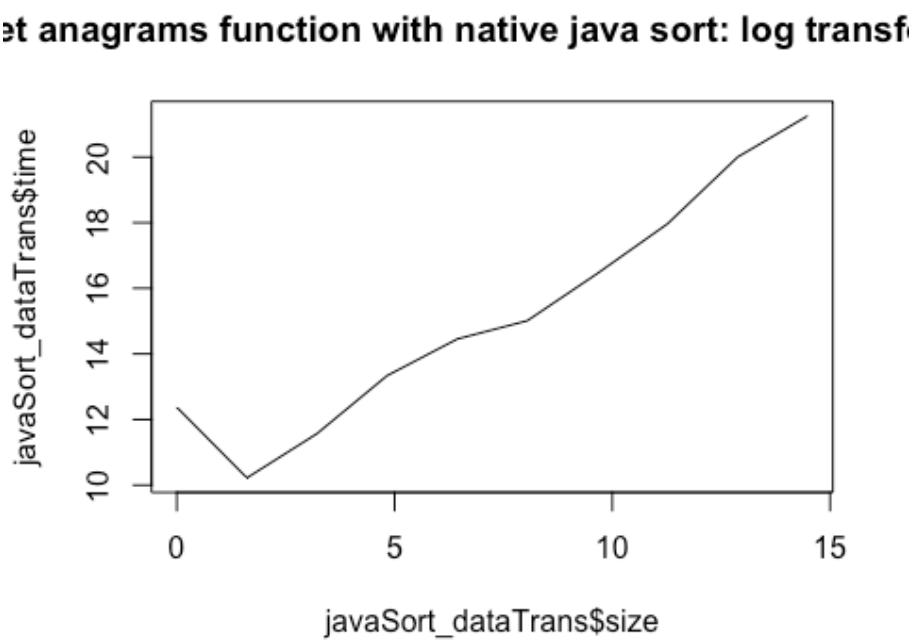
#Log transform the data

```
javaSort_dataTrans = log(javaSort_data)
```

#Plot the transformed data

#Plot the transformed data

```
plot(javaSort_dataTrans$size, javaSort_dataTrans$time, type="l")
title("Get anagrams function with native java sort: log transformed")
```




```
#   -What is the Big-O behavior and why? Be sure to define N.
# The Big-O behavior for the getting the largest group of anagrams using the java sort is N logN.
#   -Does the growth rate of the plotted running times match the Big-O behavior you predicted?
# Yes, the slope of the line is quite linear.
```