Problem Set 3

Jeffrey Kwarsick

October 2, 2017

1 Question 1

Question 1 was completed and uploaded as bestpractices.txt. It is located in the ps3 folder of this repository.

2 Question 2

2.1 Part (a)

To start this question, the file that contained all of the collected works of Shakespeare, pg100.txt, was downloaded. The plays were identified by the years of publication and the years were used to extract all of the text in between the lines containing the year of publication for each play. Starting this way, the sonnets and Lover's Complaint are included in the output list.

```
library('stringr')
fileString <- "/home/jkwarsick/Documents/STAT243_Fall2017/ps3-2017/pg100.txt"
filecon <- file(fileString)

play_texts <- list()
output_start <- FALSE
DEBUG <- TRUE
current_play_text_lines <- c()

for (line in readLines(filecon)) {
  new_play <- grepl("^[[:digit:]]{4}$", line)
  if (new_play && DEBUG) {
    print(line)
  }
  if (new_play) {
    output_start <- TRUE
    play_texts[[length(play_texts)+1]] <- current_play_text_lines
    current_play_text_lines <- c()
  }
  if (output_start) {
    current_play_text_lines <- c(current_play_text_lines, line)
  }
}

## [1] "1609"
## [1] "1603"
## [1] "1607"
## [1] "1601"
## [1] "1593"
## [1] "1608"
## [1] "1609"
## [1] "1604"
## [1] "1598"
## [1] "1598"
## [1] "1599"
## [1] "1592"
## [1] "1591"
## [1] "1591"
## [1] "1611"
## [1] "1597"
## [1] "1599"
## [1] "1606"
## [1] "1595"
## [1] "1606"
## [1] "1605"
## [1] "1597"
## [1] "1601"
## [1] "1596"
## [1] "1599"
## [1] "1605"
## [1] "1596"
## [1] "1593"
## [1] "1595"
## [1] "1594"
## [1] "1612"
## [1] "1608"
## [1] "1594"
## [1] "1602"
## [1] "1602"
## [1] "1595"
## [1] "1611"
## [1] "1609"
```

A list of length 37 is produced. This is all of Shakespeare's collected works, sonnets and all plays.

2.2 Part (b)

In this part, publication year, publication title, number of acts and the number of scenes are extracted from the play texts contained in the list play texts. In order to better determine the correct number of acts and scenes in each play, I conduct some preprocessing to clean up the occurences where the act and scene number are listed. This preprocessing made it easier to correctly determine the number of acts and the total number of scenes per play.

```
#This function changes the inconsistent format of occurrences of scene
scene_cleanup <- function(entry) {
  #entry <- entry[entry != ""]
  gsub("[[:punct:]]?([[:space:]]{1}SCENE[[:space:]]{1})|([[:space:]]{1}Scene[[:space:]]{1})|(SCENE[[:spa
}
#This function cleans up the format of the occurences of act
#Dont to determine number of acts with greater ease
act_cleanup <- function(entry) {
  gsub("(ACT )|(Act )", "ACT_", entry)
}
```

```
#Cleans the scene formatting in play texts
scene_clean_text <- lapply(play_texts[2:length(play_texts)], scene_cleanup)
#Cleans up ACT formatting in the play texts
act_clean_text <- lapply(scene_clean_text, act_cleanup)
```

After this point in the clean-up, I remove the sonnets from the processing list and proceed from this point, only looking at the plays by Shakespeare. The four pieces of meta-data required in this part are then extracted and contained in a list of lists, with each sub-list being length four, containing all the required meta-data.

```
#Extracts the play year, play title, number of acts, and number of scenes
extract_play_information <- function(entry) {
  acts_cnt <-length(grep("ACT_", entry))
  scenes <- length(grep("SC_", entry))
  entry <- entry[entry != ""]
  play_year  <- as.integer(entry[1])
  play_title <- entry[2]
  return(list(play_year, play_title, acts_cnt, scenes))
}
#Line to extract play objects of Part (b) of the problem
#Play year, Play Title, Number of Acts, Number of Scenes
play_objects <- lapply(act_clean_text[-4], extract_play_information)
```

2.3 Part (c)

I start this part of the problem by removing the Dramatis Personae from the object containing all of the play text, play texts.

```
#Function looks for String = Dramatis Personae and <<
#Then removes the Dramatis Personae from the play text
rm_dramatis_personae <- function(entry) {
  dram_strt <- grep("(Dramatis Personae)|(DRAMATIS PERSONAE)", entry)
  dram_end <- grep("<<", entry)
  entry <- entry[-(dram_strt:dram_end[1]-1)]
}

#Remove Dramatis Personae from play text
dram_free_text <- lapply(act_clean_text, rm_dramatis_personae)
```

I then remove the stage directions from the object play texts.

```
#Function to remove the stage directions and descriptions
rm_stage_dir <- function(entry) {
  #removes whitespace, cases of enter, exeunt, and exit then other characters
  gsub("([[:space:]]+)((Enter)|(Exeunt)|(Exit))([[:print:]]+)?$", "", entry)
}
#Stage free text version 1.0, now to grab the characters and their word chunks
stagedir_free_txt <- lapply(dram_free_text, rm_stage_dir)
```

Next, I move on to extracting the lines of spoken chunks for each character in each play by Shakespeare. This is accomplished with the function, find characters lines. There is a considerable amount of processing that occurs in this function. Each spoken chunk is identified by the name of the character. In all of the plays, the character is identified at the start of their spoken chunk with their name followed by a period. I use grepl to identify the character name and mark the start of their spoken chunk. The indices coordinates for their full spoken chunk that occurs over multiple lines. Other functions, create charblock type get all spoken lines are used to assemble the list of all of the spoken chunks and each speaker associated with each chunk. In this function, I can determine the number of spoken chunks in each play and calculate the number of sentences in each of these chunks, as well as calculate the total number of word in each spoken chunk. The last will be important to determine the average number words per chunk in each play by using the total number of spoken chunks. The function also outputs a data frame containing the number of the unique speakers as well as the condensed spoken word chunks for each unique speaker. This will be useful for determining the unique number of words in each play after removal of the punctuation marks.

```r
#Sentence counter per chunk
count_sentences <- function(entry) {
  num_sentences <- str_count(entry, "(\\.)|(\\?)|(\\!)")
  return(num_sentences)
}
#counts the number of words per chunk
count_words <- function(entry) {
  num_words <- str_count(entry, "[[:alpha:]]+")
  return(num_words)
}
find_characters_lines <- function(entry) {
  charc_loc <- grepl("(^  )([[:alpha:]]+)([[:space:]]{1}[[:alpha:]]+){0,}(\\.)", entry)
  line_indices <- which(charc_loc)
  line_indices <- c(line_indices, length(entry))
  block_coordinates <- lapply(seq(1, length(line_indices)-1),
                              function(i) { seq(line_indices[i], line_indices[i+1]-1) } )

  str_blocks <- lapply(block_coordinates, function(i) { entry[i] })
  lapply(str_blocks, create_charblock_type)
  blocks <- as.data.frame(matrix(unlist(lapply(str_blocks, create_charblock_type)),
                                 ncol = 2, byrow = TRUE),
                          stringsAsFactors = FALSE)
  sentences_p_chunk <- lapply(blocks$V2, count_sentences)
  words_p_chunk <- lapply(blocks$V2, count_words)
  num_spoken_chunks <- length(blocks$V2)  #number of spoken chunks per play
  unique_character_names <- unique(blocks$V1)
  return_df <- data.frame(unique_character_names,
                          unlist(lapply(
                            unique_character_names, get_all_spoken_lines, my_blocks=blocks)))
  names(return_df) <- c("character", "all_text")
  returnables <- c(return_df, num_spoken_chunks, sentences_p_chunk, words_p_chunk)
  return(returnables)
}

create_charblock_type <- function(string_list) {
  # Get the character's name, this code can be cleaned up
  tmp.string_list <- string_list
  character_str <- trimws(strsplit(string_list[1], '\\.')[[1]][1])
  first_line_text <- strsplit(string_list[1], '\\.')[[1]][2]
  tmp.string_list[1] <- first_line_text
  return(c(character_str, paste(trimws(tmp.string_list), collapse=" ")))
}

get_all_spoken_lines <- function(char_name, my_blocks) {
  return(paste(my_blocks[my_blocks$V1 == char_name,]$V2, collapse=" "))
}

#Extracts all relevant character information.
#Excluding the play, Comedy of Errors
all_plays_spkr_processing <- lapply(stagedir_free_txt[-4], find_characters_lines)
```

In order extract the total number of words in each of the plays, I create 35 separate variables and step through the lists where the number of words per spoken chunk are stored. These numbers are summed to these 35 variables. They are then divided by the number of spoken chunks in each play to give the average number of words per spoken chunk for each play. I then determine the total number of words per play.

```
#Sample of code used to create variable for the number of words per play.
word_play_tot <- c()
for (i in 1:length(all_plays_spkr_processing)) {
  tmp <- 0
  low_bound <- length(all_plays_spkr_processing[[i]])-all_plays_spkr_processing[[i]][[3]]
  for (j in low_bound:length(all_plays_spkr_processing[[i]])) {
    tmp <- tmp + all_plays_spkr_processing[[i]][[j]]
  }
  word_play_tot <- c(word_play_tot, tmp)
}

#Calulates the number of words per chunk for each play
#then saves it to a list
avg_wdp_chunk_per_play <- c()
for (i in 1:length(word_play_tot)) {
  tmp <- 0
  tmp <- word_play_tot[i]/all_plays_spkr_processing[[i]][[3]]
  avg_wdp_chunk_per_play <- c(avg_wdp_chunk_per_play, tmp)
}
```

Next, the chunks output from the find character lines functions are operated on to strip away the punctuation marks from the text. After completing this, the text can then be operated on to determine the number of unique words per play.

2.4 Part (d)

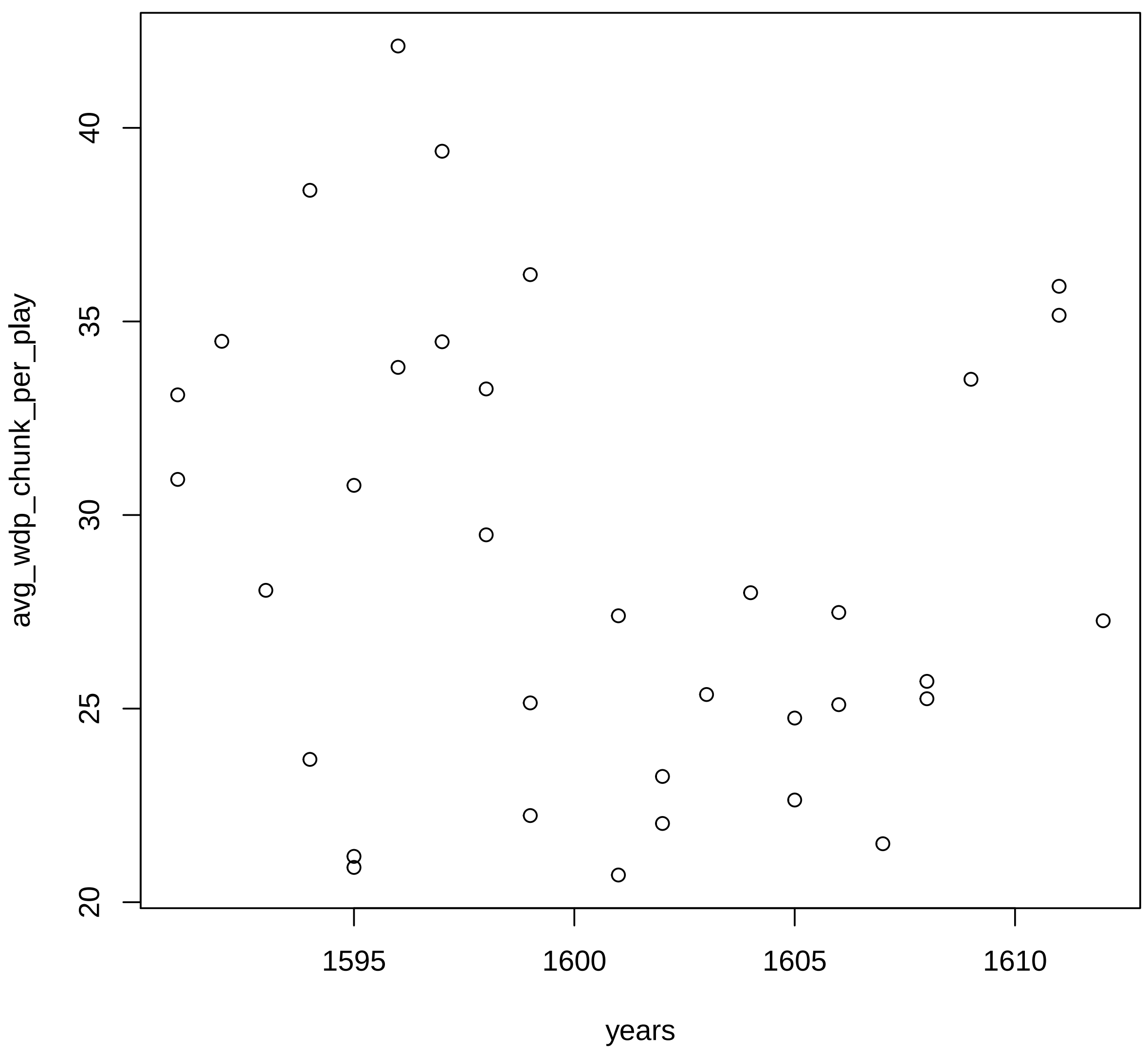
I then extract various information from my R objects in order to use them for plotting. This includes the publication year from my list of lists containing the meta-data, play objects. The number of acts and scenes are also extracted. From my other objects created in the course of this assignment, I extracted number of unique speakers, total number of words per play, average number of words per chunk, and number of chunks per play. All of these are plotted against publication year.

```
#Extract play years into their own list to eas plot making
years <- c()
for (i in 1:length(play_objects)) {
  year <- play_objects[[i]][[1]]
  years <- c(years, year)
}
#extracting the number of chunks per play for plotting
chunks_p_play <- c()
for (i in 1:length(all_plays_spkr_processing)) {
  num_chunks <- all_plays_spkr_processing[[i]][[3]]
  chunks_p_play <- c(chunks_p_play, num_chunks)
}
#Extract Number of Acts per play
acts <- c()
for (i in 1:length(play_objects)) {
  num_act <- play_objects[[i]][[3]]
  acts <- c(acts, num_act)
}
#Extract the scenes per play
scenes <- c()
for (i in 1:length(play_objects)) {
  num_scene <- play_objects[[i]][[4]]
  scenes <- c(scenes, num_scene)
}
#extracting the number of unique speakers per play
uniq_spkrs_p_play <- c()
for (i in 1:length(all_plays_spkr_processing)) {
  num_spkrs <- length(all_plays_spkr_processing[[i]][[1]])
  uniq_spkrs_p_play <- c(uniq_spkrs_p_play, num_spkrs)
}
#Extract the titles of the plays
titles <- c()
for (i in 1:length(play_objects)) {
  title1 <- play_objects[[i]][[2]]
  titles <- c(titles, title1)
}
#Assemble data frame of title, number of acts and number of scenes
meta_data_df <- data.frame(titles, acts, scenes)
meta_data_df
```

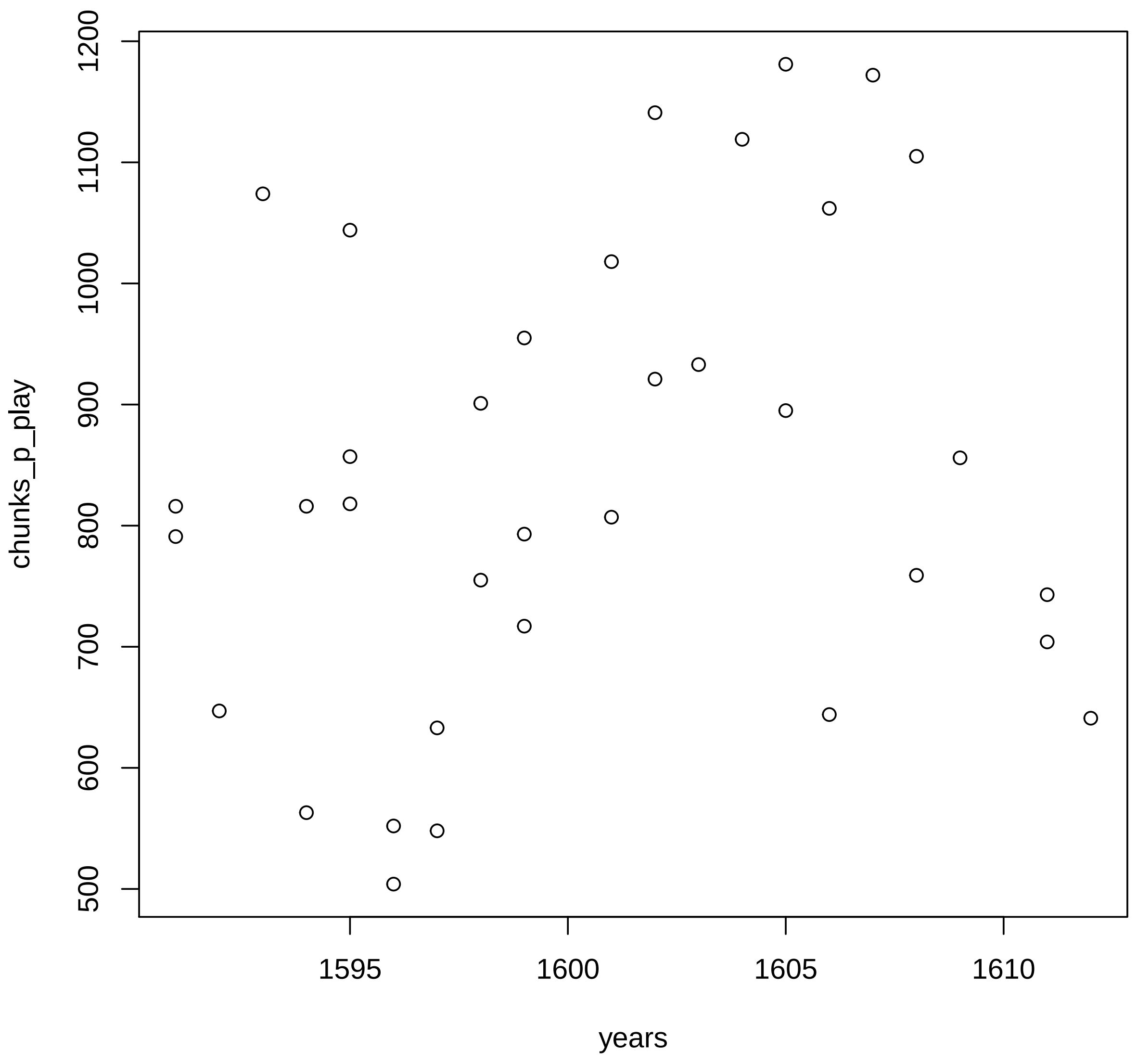
```
##                                        titles acts scenes
## 1                   ALLS WELL THAT ENDS WELL   23     23
## 2        THE TRAGEDY OF ANTONY AND CLEOPATRA   36     73
## 3                              AS YOU LIKE IT    5     22
## 4                   THE TRAGEDY OF CORIOLANUS    5     29
## 5                                   CYMBELINE    5     27
## 6  THE TRAGEDY OF HAMLET, PRINCE OF DENMARK    5     20
## 7   THE FIRST PART OF KING HENRY THE FOURTH    5     19
## 8                SECOND PART OF KING HENRY IV    5     19
## 9           THE LIFE OF KING HENRY THE FIFTH    5     23
## 10        THE FIRST PART OF HENRY THE SIXTH    5     27
## 11  THE SECOND PART OF KING HENRY THE SIXTH    5     24
## 12   THE THIRD PART OF KING HENRY THE SIXTH    5     28
## 13                    KING HENRY THE EIGHTH   17     17
## 14                                KING JOHN    5     16
## 15             THE TRAGEDY OF JULIUS CAESAR    5     18
## 16                THE TRAGEDY OF KING LEAR    5     26
## 17                     LOVE’S LABOUR’S LOST    5      9
## 18                  THE TRAGEDY OF MACBETH    5     29
## 19                     MEASURE FOR MEASURE    5     17
## 20                   THE MERCHANT OF VENICE    5     20
## 21                THE MERRY WIVES OF WINDSOR    5     23
## 22                 A MIDSUMMER NIGHT’S DREAM    5      9
## 23                    MUCH ADO ABOUT NOTHING    5     17
## 24   THE TRAGEDY OF OTHELLO, MOOR OF VENICE    5     15
## 25                  KING RICHARD THE SECOND    5     19
## 26                         KING RICHARD III    5     25
## 27          THE TRAGEDY OF ROMEO AND JULIET    5     24
## 28                  THE TAMING OF THE SHREW    5     16
## 29                              THE TEMPEST    5      9
## 30              THE LIFE OF TIMON OF ATHENS    5     17
## 31          THE TRAGEDY OF TITUS ANDRONICUS    5     14
## 32      THE HISTORY OF TROILUS AND CRESSIDA   24     24
## 33         TWELFTH NIGHT; OR, WHAT YOU WILL    5     18
## 34              THE TWO GENTLEMEN OF VERONA    6     21
## 35                        THE WINTER'S TALE    5     15
```

I did have some issues with a few of the plays when I went to extract the acts and scenes. This was due to repetitions of scenes and act labels within the plays that had issue. Now, I move on to printing the out some plots.

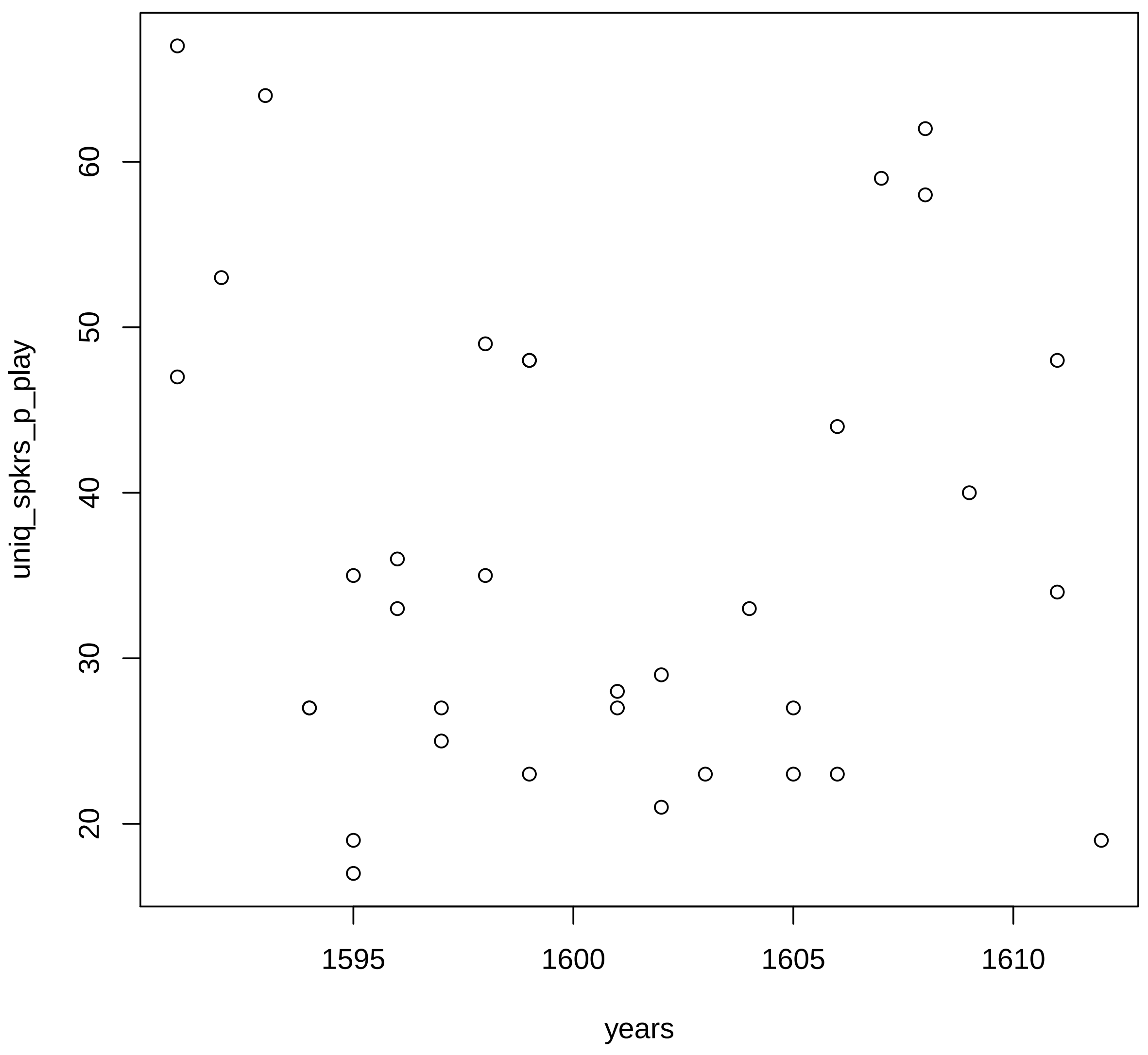
```
#plot of years vs words per spoken chunk per play
plot(years, avg_wdp_chunk_per_play)
```



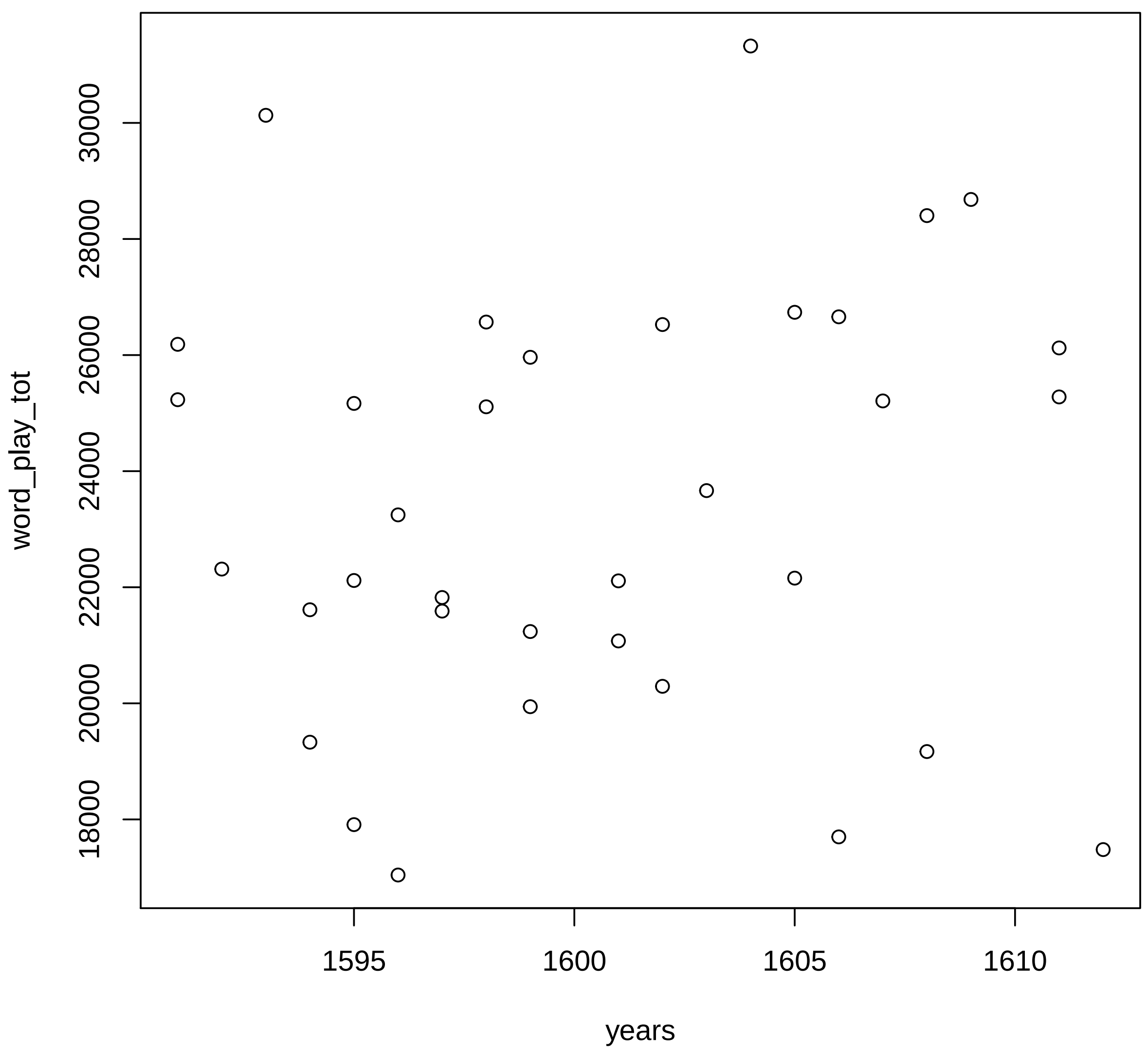
```
#Number of Chunks per play.
plot(years, chunks_p_play)
```



```
#plot of years vs number of unique speakers per play
plot(years, uniq_spkrs_p_play)
```



```
#Plot of years vs words per play
plot(years, word_play_tot)
```



Starting with total words per play vs. years, I observed that the Shakespeare tended to write plays that varied in length from roughly 17000 words and as high as 31000 words. Overall, he tended to average about 24000 words per play over his writing career. He also showed no trend for how words per play.

For unique speakers per play versus years, I observed that Shakespeare peaked early on in his career for number of speakers. He then hit a minimum for the bulk of his years active (1595 - 1605), before rising and falling again in speaker count per play. In chunks per play vs. years, he started his career off on the lower side before peaking in number of word chunks per play around 1605. Finally, the average words per chunk per play reaches a maximum early on in Shakespeare's career, around 1595. From there, it reaches a minimum around 1605 before beginning to rise again.

3 Question 3

3.1 Part (a)

In order to build a S4 class object to contain all the relevant play information, I would have the following fields to complete the problem.

- Title of play = "character"
- Year of Publication = "numeric"
- Play text = "list of character strings"

All of these items in the system would yield an object that contained all of the information that we needed for problem 2 of this problem set. Applying different methods would be able to extract out the information requested in question 2 of this problem. These include: the number of acts per play, the total number of scenes per play, number of words per play, the number of unique words per play, the number of spoken chunks along with their speakers, the number of unique speakers, and the average number of words per chunk per play.

3.2 Part (b)

In order to get from gain relevant information from the items in our S4 class, different methods would need to be applied.

- play.num.acts would extract the number of acts in each play. This would require the the play text information with the reference class and would look and count for instances of "act" and would return a numeric answer to the query.
- play.num.scenes would extract the total number of scenes per play. This would act similarly to the previous function to find the number of acts. It would look for instances of "scene" in each play text chunk, sum the number of occurences and return a numeric value as an output.
- num.spoken.chunks would be a method to extract all occurences of speakers and their lines within a particular play. This method would need to search for the occurence of each speaker at the start of their line and extract it in the first element of a list of length two. The second element would be the speaker's spoken lines. The output would be a list of sub-lists of length two and would be the length equal to the number of spoken chunks per play. This output could then be acted on further to determine the number of unique speakers in each play by invoking the unique() function in R. This would have to ensure that play directions and other descriptions not relevant to the play are excluded.
- total.words would be a method to count all of the words that occur in the play. This would have to ensure that play directions and other descriptions not relevant to the play are excluded. With this output would be as a numeric and as a list of all the words that occur within the play text. The unique function could then be invoked on the list of words per play to determine the number of unique words per play.

All of these methods would act in a similar manner on the objects in my reference class as the functions used in problem 2 to extract all the necessary information requested by the problem. Unlike what I did for problem two, all of the base information would be contained in the object and invoking the methods would produce the information that I would seek. There would likely not be as much step-wise processing of the play text and other information in order to get it into a form that could be used to generate statistics for each play, it would all be contained within the method.