NAME:- DUBEY KARAN SANJEEV
CLASS:- B.E - 4
ROLL NO:- 04
BATCH:- A

Experiment No-9

AIM : To build WordCloud, a text mining method using R for easier to understand and visualization than table data..

THEORY:

Text mining, also referred to as text data mining, roughly equivalent to text analytics, is the process of deriving high-quality information from text. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning. Text mining usually involves the process of structuring the input text (usually parsing, along with the addition of some derived linguistic features and the removal of others, and subsequent insertion into a database), deriving patterns within the structured data, and finally evaluation and interpretation of the output. 'High quality' in text mining usually refers to some combination of relevance, novelty, and interest. Typical text mining tasks include text categorization, text clustering, concept/entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modeling (i.e., learning relations between named entities).

The five fundamental steps involved in text mining are:

- Gathering unstructured data from multiple data sources like plain text, web pages, pdf files, emails, and blogs, to name a few.
- Detect and remove anomalies from data by conducting pre-processing and cleansing operations. Data cleansing allows you to extract and retain the valuable information hidden within the data and to help identify the roots of specific words.
- For this, you get a number of text mining tools and text mining applications.
- Convert all the relevant information extracted from unstructured data into structured formats.
- Analyze the patterns within the data via the Management Information System (MIS).
- Store all the valuable information into a secure database to drive trend analysis and enhance the decision-making process of the organization.

Text analysis involves information retrieval, lexical analysis to study word frequency distributions, pattern recognition, tagging/annotation, information extraction, data mining techniques including link and association analysis, visualization, and predictive analytics. The overarching goal is, essentially, to turn text into data for analysis, via application of natural language processing (NLP) and analytical methods.
A word cloud is a text mining method that allows us to highlight the most frequently used keywords in a paragraph of texts. It is also referred to as a text cloud or tag cloud. The procedure of creating word cloud is very simple in R software if you know the different steps to execute. A text mining package (tm) and word cloud generator package (wordcloud) are available in R for helping us to analyze texts and to quickly visualize the keywords words as a word cloud.

3 reasons you should use word clouds to present your text data

- Tag cloud is a powerful method for text mining and, it add simplicity and clarity. The most used keywords stand out better in a word cloud
- Word clouds are a potent communication tool. They are easy to understand, to be shared and are impactful
- Word clouds are visually engaging than a table data

CONCLUSION:

In conclusion, A word cloud is a simple yet informative way to understand textual data and to do text analysis..

Program formation/ Execution/ ethical practices (06)	**Timely Submission and Documentation (02)**	**Viva Answer (02)**	**Experimen t Marks (10)**	**Teacher Signature with date**

Program:

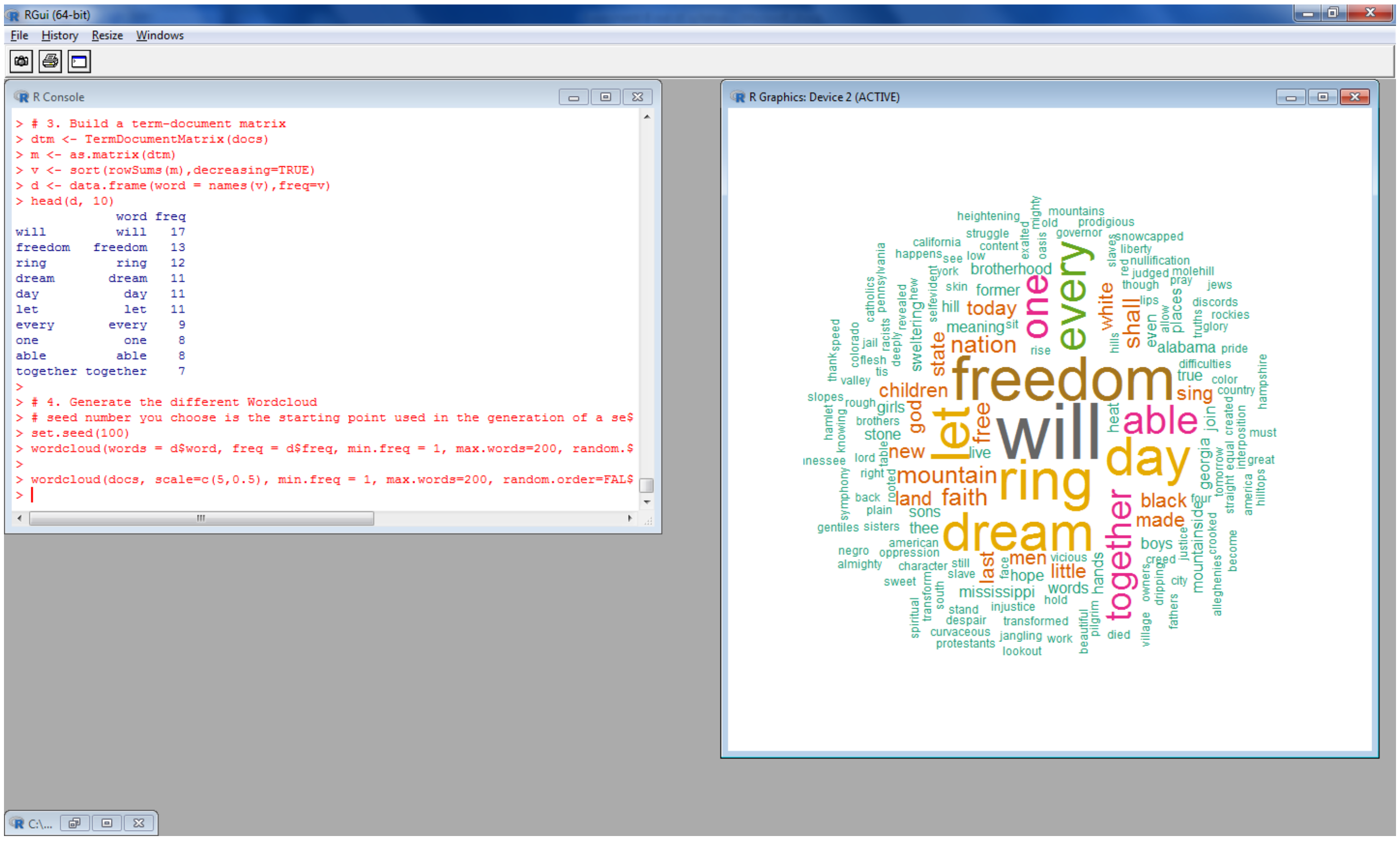
```
# 1a. Install the required packages
install.packages("tm")  # for text mining
install.packages("SnowballC") # for text stemming
install.packages("wordcloud") # word-cloud generator
install.packages("RColorBrewer") # color palettes

# 1b. Load the required packages
library("tm")
library("SnowballC")
library("wordcloud")
library("RColorBrewer")

# 2. Text mining
# load the text
# getpath and setpath
> getwd()
> list.files()
> filePath = paste(getwd(), "/dream-speech.txt", sep="")
> help.search("concatenate")
> str(filePath)
> text <- readLines(filePath)
> docs <- Corpus(VectorSource(text))
> ls()
> list.files()
> names(docs)
> str(docs)
> inspect(docs)

# Text transformation
# tm_map() is used to remove unnecessary white space, to convert the text to lower case, to remove
common stopwords like ‘the’, “we”.
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "/")
docs <- tm_map(docs, toSpace, "@")
docs <- tm_map(docs, toSpace, "\\|")


# Cleaning the text
# Slso remove numbers and punctuation with removeNumbers and removePunctuation arguments using
tm_map()


# Convert the text to lower case
docs <- tm_map(docs, content_transformer(tolower))
# Remove numbers
docs <- tm_map(docs, removeNumbers)
# Remove english common stopwords
docs <- tm_map(docs, removeWords, stopwords("english"))
# Remove your own stop word
# specify your stopwords as a character vector
docs <- tm_map(docs, removeWords, c("blabla1", "blabla2"))
# Remove punctuations
```

```
docs <- tm_map(docs, removePunctuation)
# Eliminate extra white spaces
docs <- tm_map(docs, stripWhitespace)
# Text stemming
# docs <- tm_map(docs, stemDocument)


# 3. Build a term-document matrix
dtm <- TermDocumentMatrix(docs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 10)

# 4. Generate the different Wordcloud
# seed number you choose is the starting point used in the generation of a sequence of random numbers
set.seed(100)
wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.words=200, random.order=FALSE,
rot.per=0.35, colors=brewer.pal(8, 'Dark2'))

wordcloud(docs, scale=c(5,0.5), min.freq = 1, max.words=200, random.order=FALSE, rot.per=0.35,
colors=brewer.pal(8, 'Dark2'))

wordcloud(docs, scale=c(5,0.5), max.words=100, random.order=FALSE, rot.per=0.35, use.r.layout=FALSE,
colors=brewer.pal(8, 'Dark2'))

# Arguments of the word cloud generator function :


  # words : the words to be plotted
  # freq : their frequencies
  # min.freq : words with frequency below min.freq will not be plotted
  # max.words : maximum number of words to be plotted
  # random.order : plot words in random order. If false, they will be plotted in decreasing frequency
  # rot.per : proportion words with 90 degree rotation (vertical text)
  # colors : color words from least to most frequent. Use, for example, colors ="black" for single color.
```

OUTPUT:

```
> head(d, 10)
             word freq
will         will   17
freedom   freedom   13
ring         ring   12
dream       dream   11
day           day   11
let           let   11
every       every    9
one           one    8
able         able    8
together together    7
```

OUTPUT:



```
> # 3. Build a term-document matrix
> dtm <- TermDocumentMatrix(docs)
> m <- as.matrix(dtm)
> v <- sort(rowSums(m),decreasing=TRUE)
> d <- data.frame(word = names(v),freq=v)
> head(d, 10)
             word freq
will         will   17
freedom   freedom   13
ring         ring   12
dream       dream   11
day           day   11
let           let   11
every       every    9
one           one    8
able         able    8
together together    7
>
> # 4. Generate the different Wordcloud
> # seed number you choose is the starting point used in the generation of a se$
> set.seed(100)
> wordcloud(words = d$word, freq = d$freq, min.freq = 1, max.words=200, random.$
>
> wordcloud(docs, scale=c(5,0.5), min.freq = 1, max.words=200, random.order=FAL$
> 
```