**Practical No:1**

**Aim:** Installing and setting environment variables for Working with Apache Hadoop.

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Click on play button and u will get this screen.

Graphical user interface, text, application

Description automatically generated

**Practical No: 2**

**Aim:** Implementing Map-Reduce Program for Word Count problem.

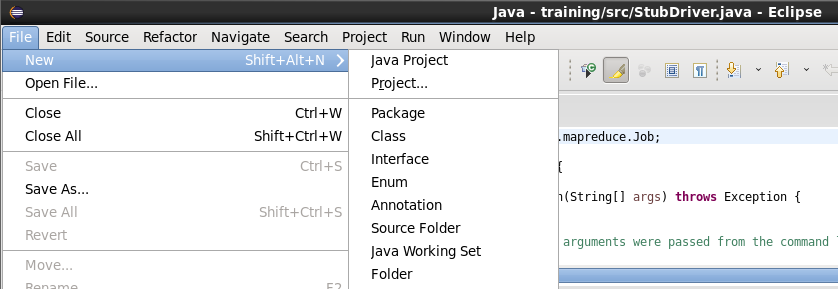
**Description:**

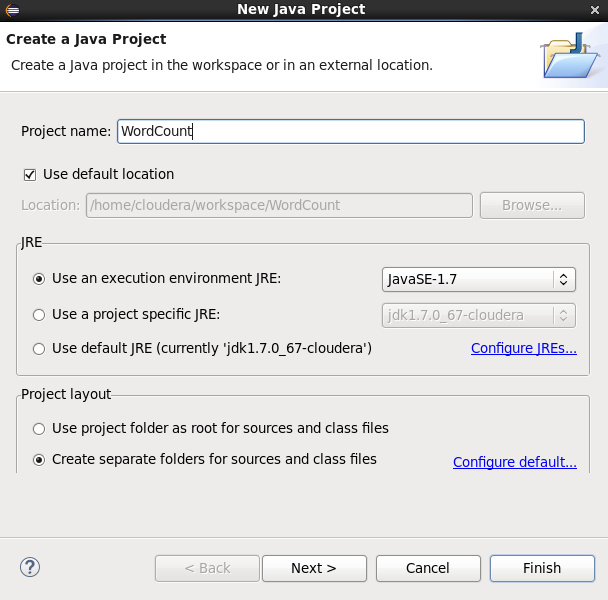
Map-Reduce is a software programming model designed in java programming language. It is combination of two individual task namely:

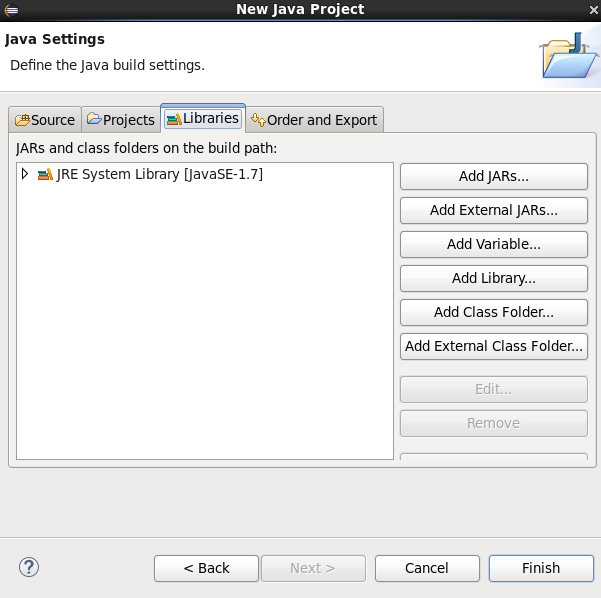
**1.Map:** Map takes the dataset and divides them into chunks such that they are converted into new format which should be in form of key-value pair.

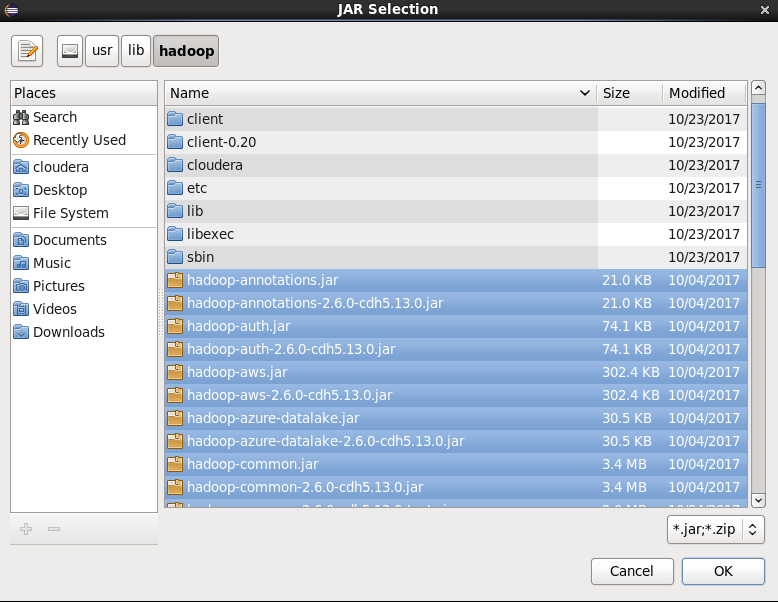
**2.Reduce:** Reduce is another part where key-value pair reduced to tuples.

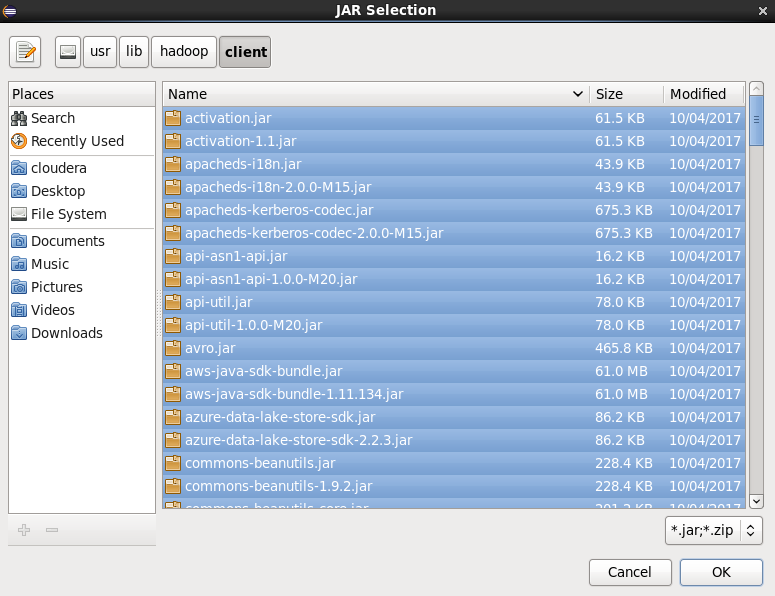
**Steps & Output:**

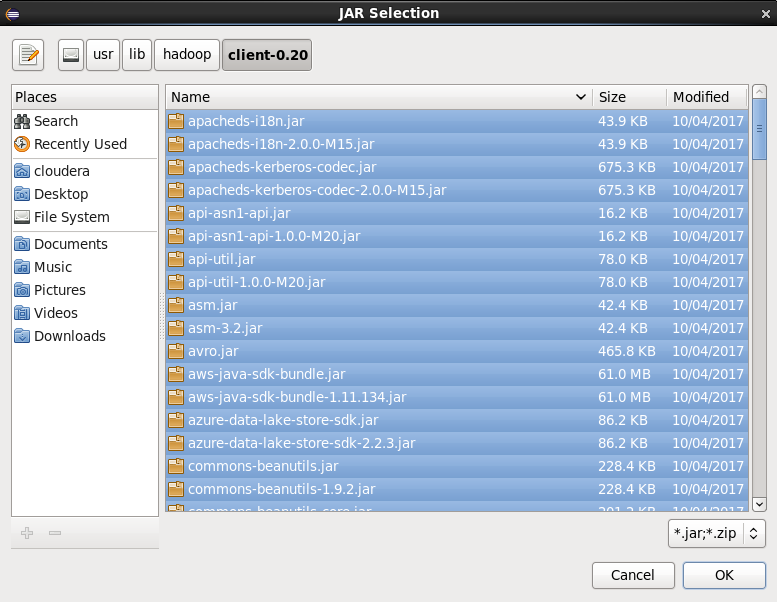


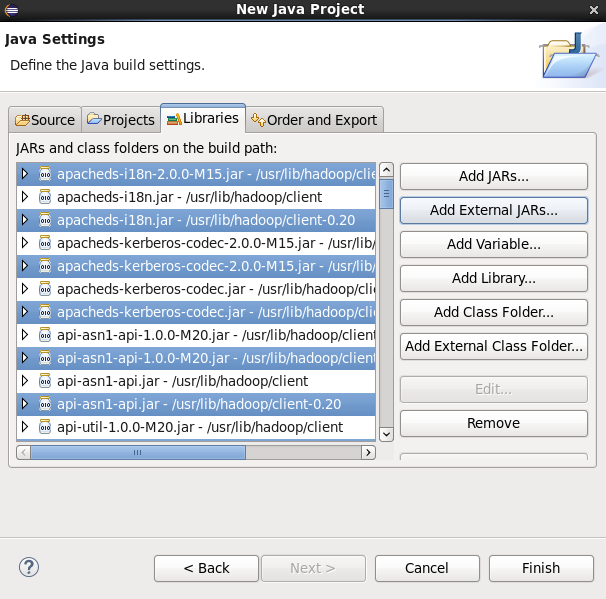


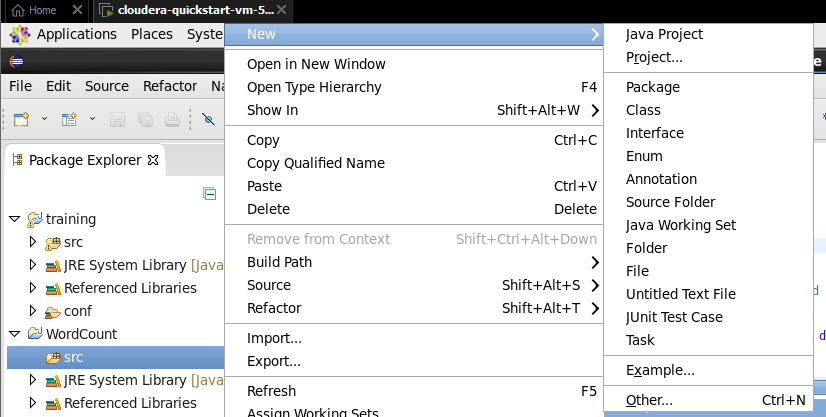


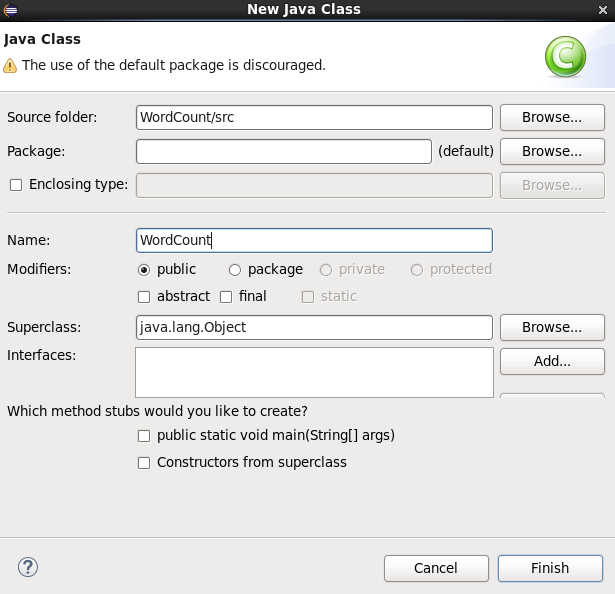


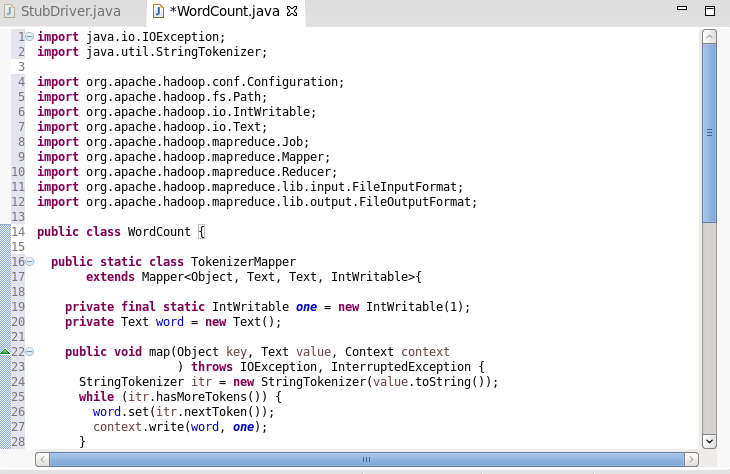


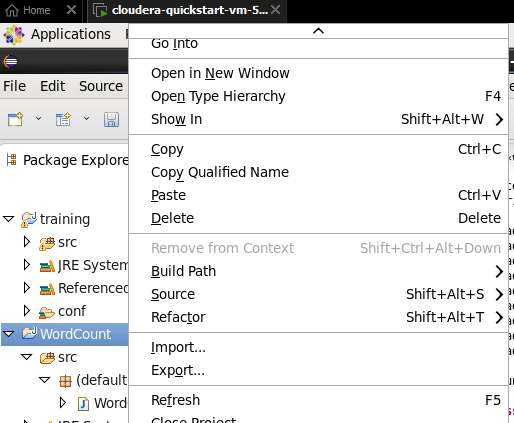


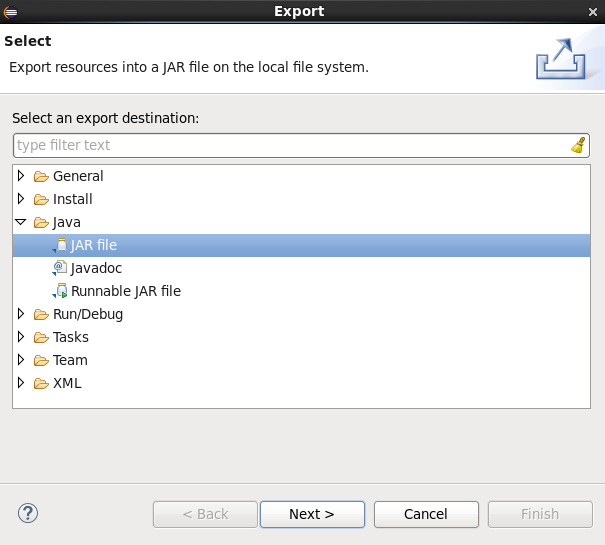


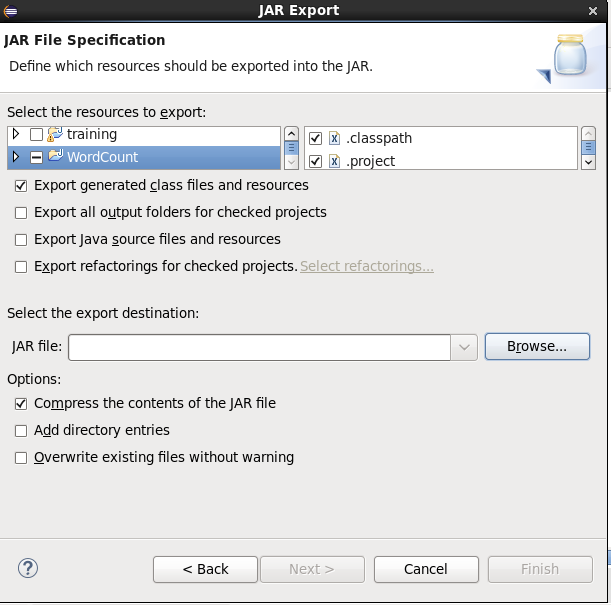


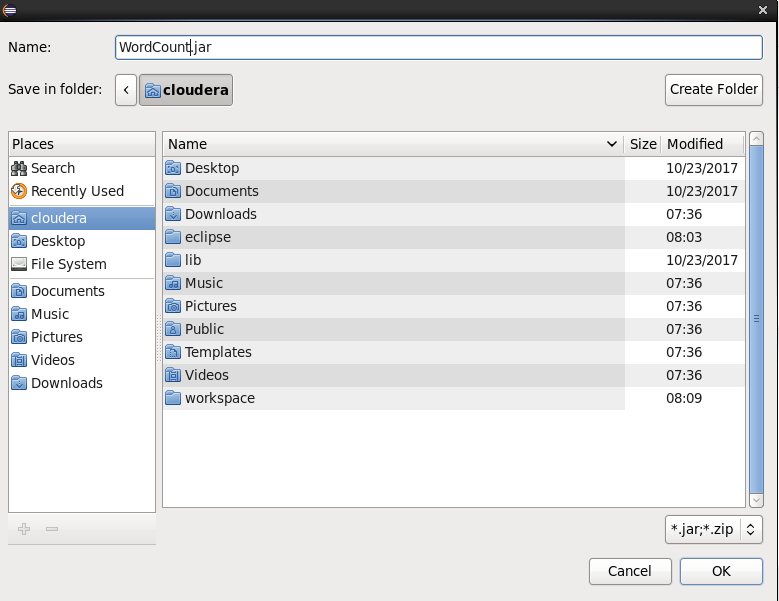


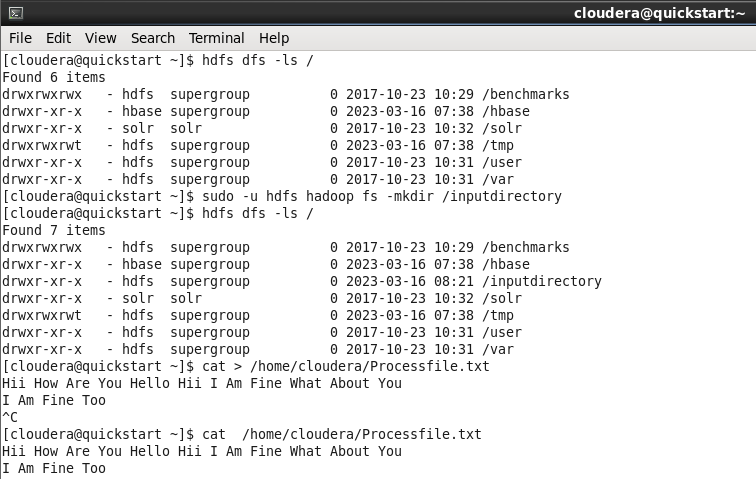


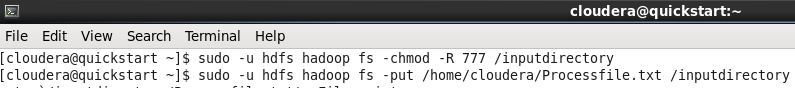


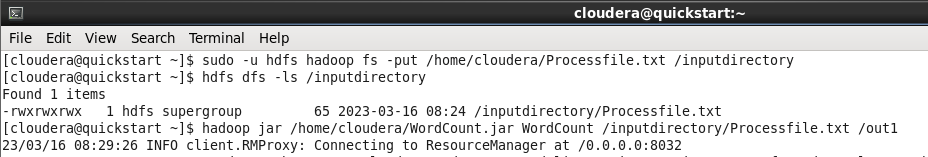


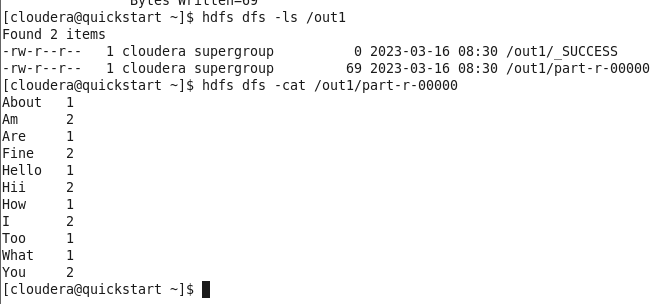












**Practical No: 3**

**Aim:** Write a Pig Script for solving counting problems.

**Description:**

Apache Pig is an abstraction over MapReduce. It is a tool/platform which is used to analyze larger sets of data representing them as data flows. Pig is generally used with **Hadoop**; we can perform all the data manipulation operations in Hadoop using Apache Pig.

To write data analysis programs, Pig provides a high-level language known as **Pig Latin**. This language provides various operators using which programmers can develop their own functions for reading, writing, and processing data.

To analyze data using **Apache Pig**, programmers need to write scripts using Pig Latin language. All these scripts are internally converted to Map and Reduce tasks. Apache Pig has a component known as **Pig Engine** that accepts the Pig Latin scripts as input and converts those scripts into MapReduce jobs.

**Steps & Output:**

cat> /home/cloudera/input.csv

cat /home/cloudera/input.csv

pig -x local

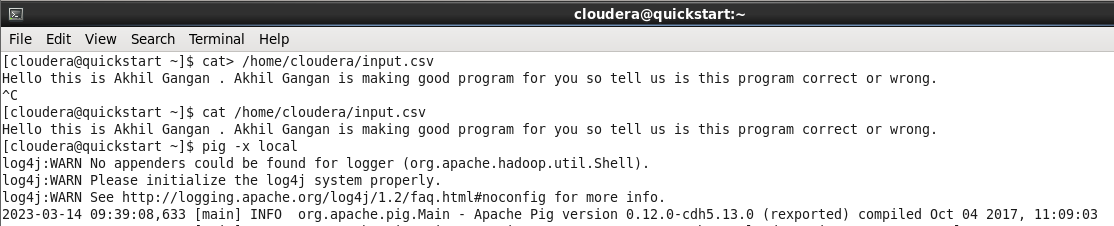
lines = load '/home/cloudera/input.csv' as (line:chararray);

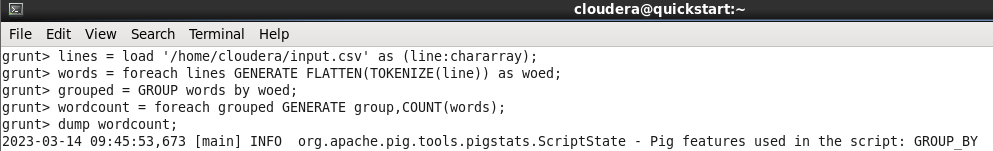
words = foreach lines GENERATE FLATTEN(TOKENIZE(line)) as woed;

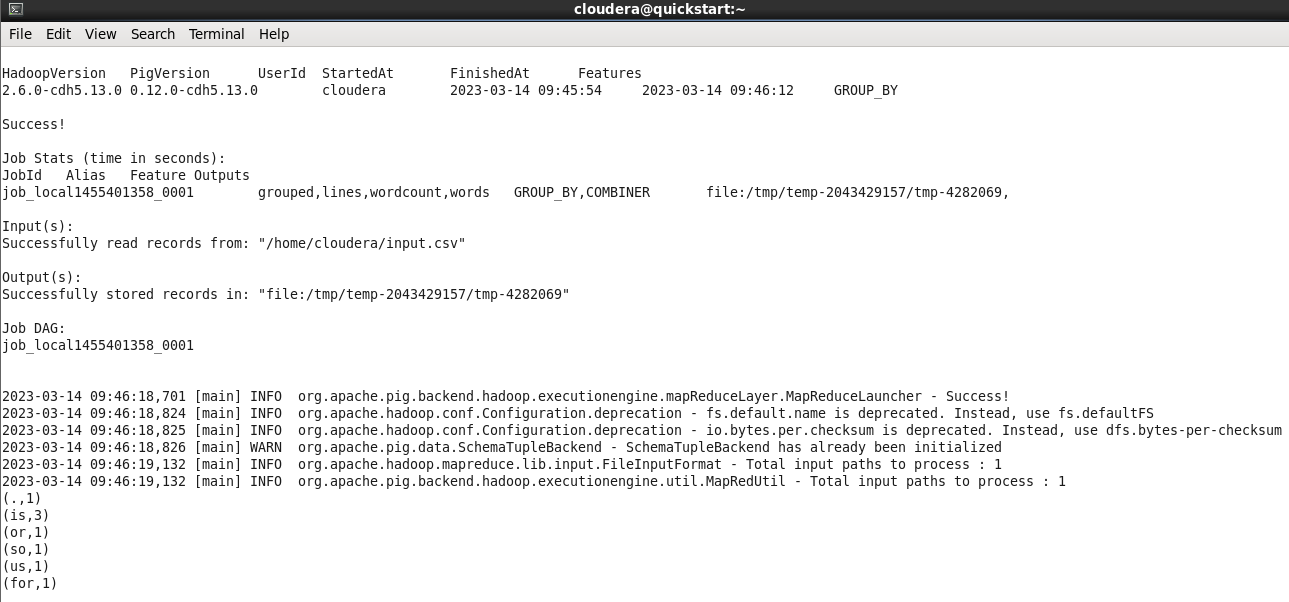
grouped = GROUP words by woed;

wordcount = foreach grouped GENERATE group, COUNT(words);

dump wordcount;

****

****

****

****

**Practical No: 4**

**Aim:** Install HBase and use the HBase Data model Store and retrieve data.

**Description:**

HBase is a distributed column-oriented database built on top of the Hadoop file system. It is an open-source project and is horizontally scalable.

HBase is a data model that is similar to Google’s big table designed to provide quick random access to huge amounts of structured data.

It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System.

**Steps & Output:**

//Start HBase

hbase shell

//HBase Commands

status

version,

table\_help

whoami

//Data Definition Language

create ‘employee’, ‘Name’, ‘ID’, ‘Designation’, ‘Salary’, ‘Department’

//Verify created table

list

//Disable single table

disable ‘employee’

scan ‘employee’

//or

is\_disable ‘employee’

//Disable multiple tables

disable\_all ‘e.\*’

// Enabling table

enable‘employee’

//Or

is\_enabled ‘employee’

//create new table

create‘student’, ‘name’, ‘age’, ‘course’

put ‘student’, ‘sharath’, ‘name:fullname’, ‘sharathkumar’

put ‘student’, ‘sharath’, ‘age:presentage’, ‘24’

put ‘student’, ‘sharath’, ‘course:pursuing’, ‘Hadoop’

put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank R

put ‘student’, ‘shashank’, ‘age:presentage’, ‘23’

put ‘student’, ‘shashank’, ‘course:pursuing’, ‘Java’

//Get Information

get ‘student’, ‘shashank’

get ‘student’, ‘sharath’

get ‘student’, ‘sharath’, ‘course’

get ‘student’, ‘shashank’, ‘course’

get ‘student’, ‘sharath’, ‘name

//Scan

scan ‘student’

//Count

Count ‘student’

//Alter

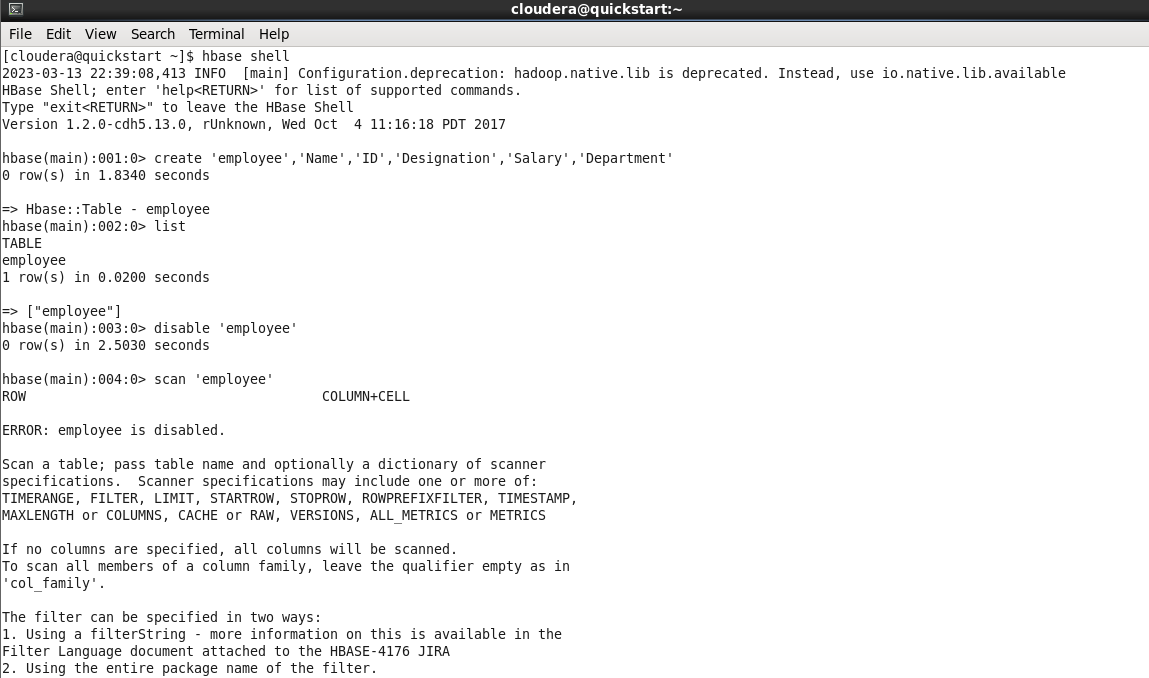
alter ‘student’, NAME=>’name’, VERSIONS=>5

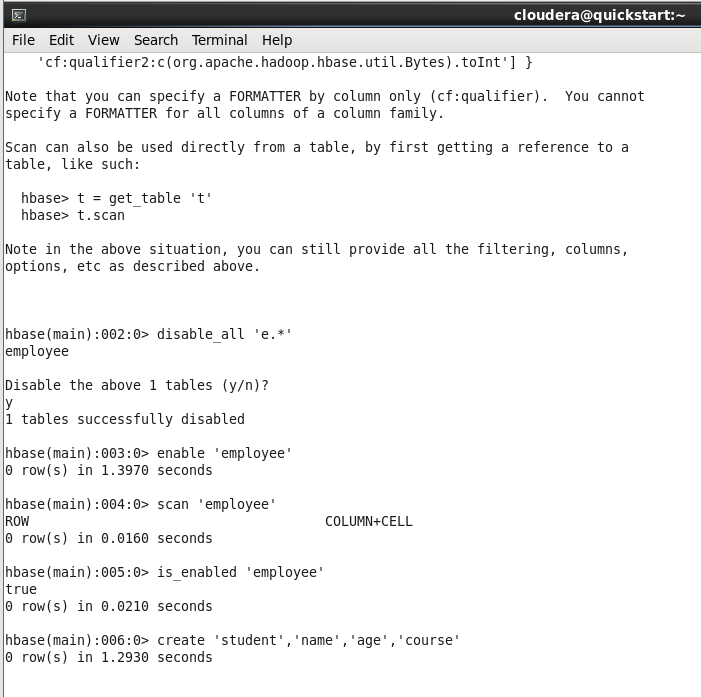
put ‘student’, ‘shashank’, ‘name:fullname’, ‘shashank Rao’

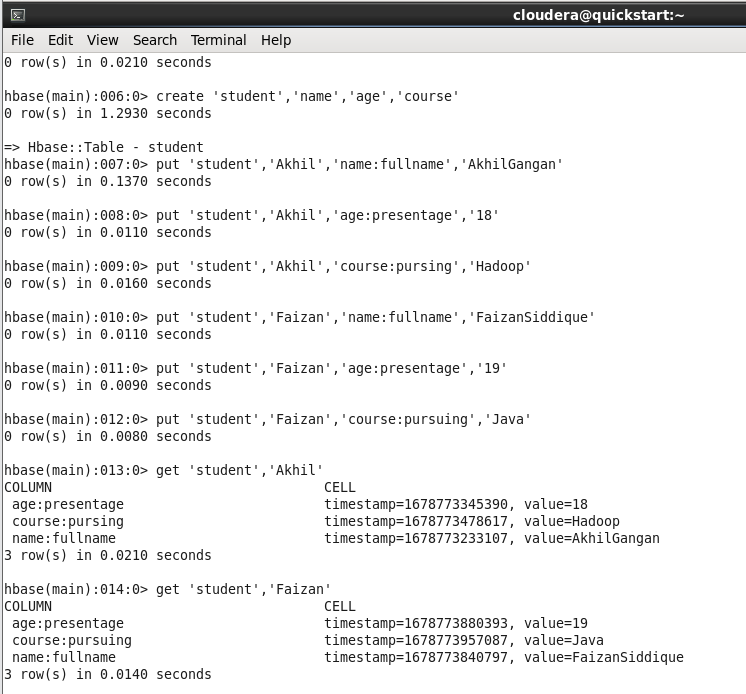
scan ‘student’

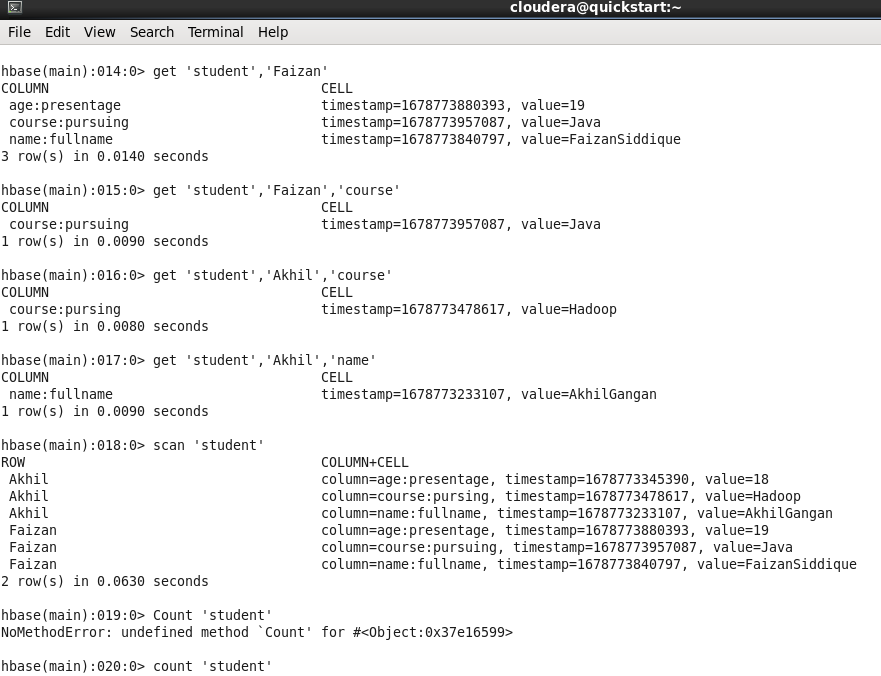
//Delete

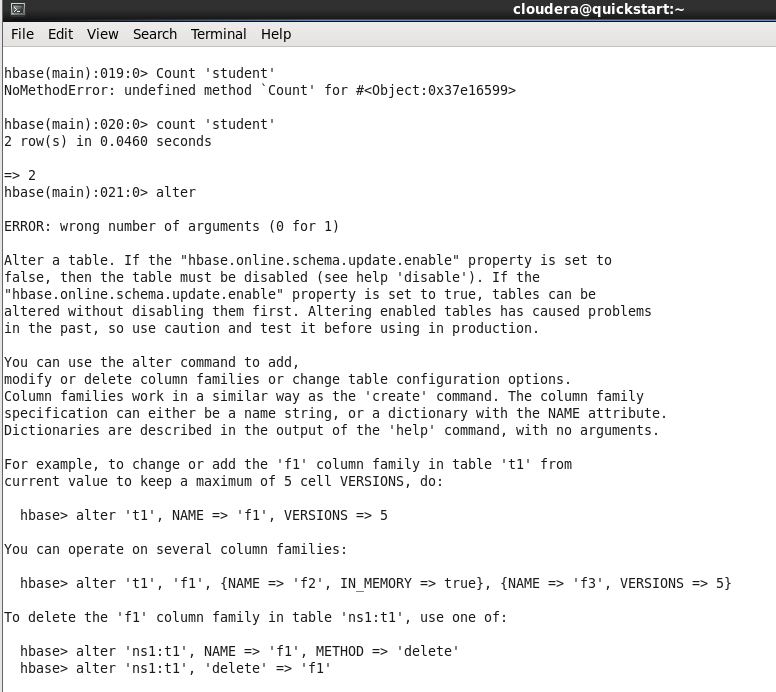
delete ‘student’, ‘shashank’, ‘name:fullname’

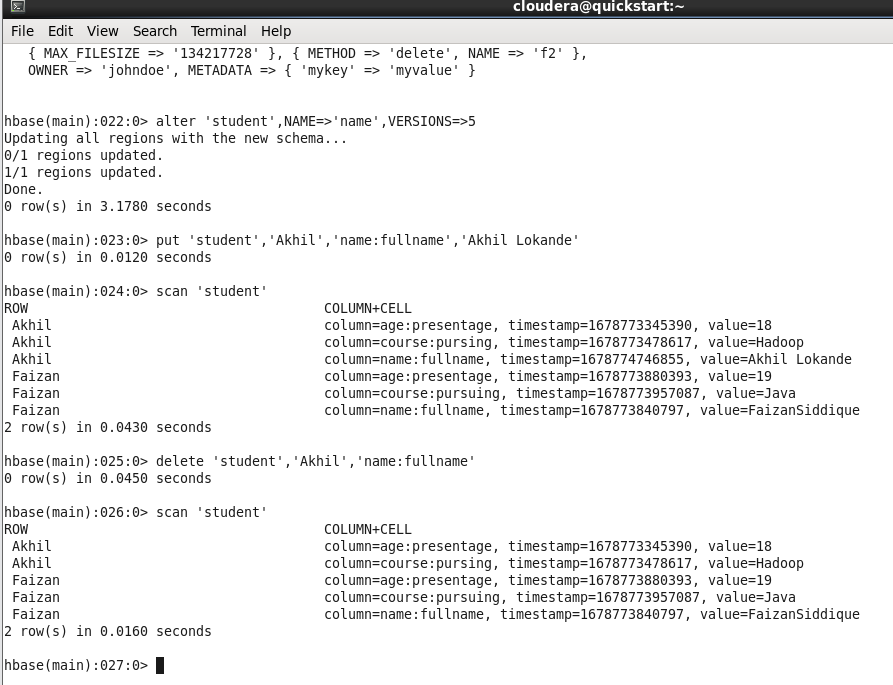












**Practical No: 5**

**Aim:** Install Hive and use Hive Create and store structured databases.

**Description:**

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy.

Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive. It is used by different companies. For example, Amazon uses it in Amazon Elastic MapReduce.

It is a platform used to develop SQL type scripts to do MapReduce operations.

**Steps & Output:**

Cat > /home/cloudera/employee.txt

1~Sachine~Pune~Product Engineering~100000~Big Data

2~Gaurav~Banglore~Sales~90000~CRM

3~Manish~Chennai~Recruiter~125000~HR

4~Bhushan~Hyderabad~Developer~50000~BFSI

cat /home/cloudera/employee.txt

sudo -u hdfs hadoop fs -put /home/cloudera/employee.txt /inputdirectoryy

hdfs dfs -ls /

hdfs dfs -ls /inputdirectory

hadoop fs -cat /inputdirectory/employee.txt

hive

show databases;

create database organization;

show databases;

use organization;

show tables;

hive> create table employee(

> id int,

> name string,

> city string,

> department string,

> salary int,

> domain string)

> row format delimited

> fields terminated by '~';

show tables;

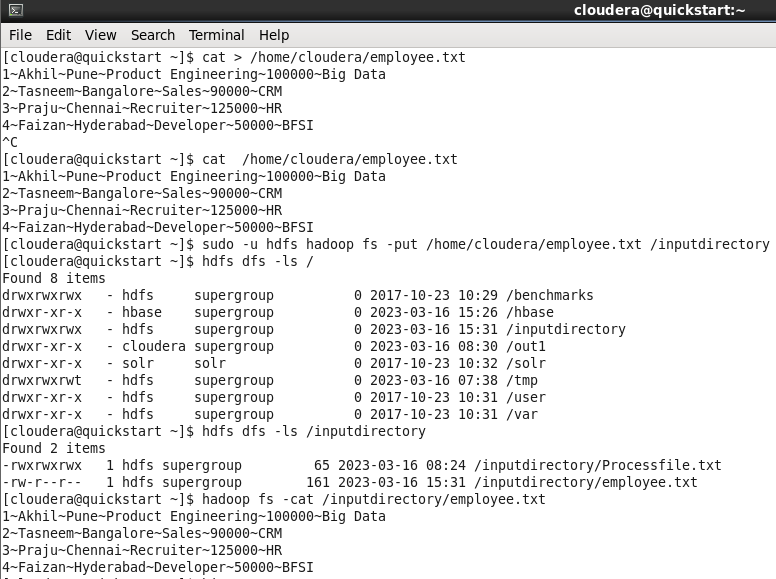
select \* from employee;

show tables;

load data inpath '/inputdirectory/employee.txt' overwrite into table employee;

show tables;

select \* from employee;



Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

**Practical No: 6**

**Aim:** Write a program to construct different types of k-shingles for a given document.

**Description:**

A k -shingle (or k -gram) for a document is a sequence of k characters that appears in the document.

Example: k=2; doc = abcab. Set of 2- shingles = {ab, bc, ca, ab}.

**Steps & Output:**

install.packages("tm")

require("tm")

install.packages("devtools")

readinteger<-function()

{

n<-readline(prompt="enter value of k-1:")

k<-as.integer(n)

u1<-readLines("E:/MSC Notes/file.txt")

shingle<-0

i<-0

while(i<nchar(u1)-k+1){

shingle[i]<-substr(u1,start=i,stop=i+k)

print(shingle[i])

i=i+1

}

}

if(interactive())readinteger()

**OUTPUT:**

Graphical user interface, text, application

Description automatically generated

**Practical No: 7**

**Aim:** Write a program for measuring similarity among documents and detecting passages which have been reused.

**Description:**

Document similarity, as the name suggests determines how similar are the two given documents. By “documents”, we mean a collection of strings. For example, an essay or a .txt file. Many organizations use this principle of document similarity to check plagiarism. It is also used by many exams conducting institutions to check if a student cheated from the other.

**Steps & Output:**

install.packages("tm")

require("tm")

install.packages("ggplot2")

install.packages("textreuse")

install.packages("devtools")

my.corpus<-Corpus(DirSource("E:/MSC Notes"))

my.corpus<-tm\_map(my.corpus,removeWords,stopwords("english"))

my.tdm<-TermDocumentMatrix(my.corpus)

my.dtm<-DocumentTermMatrix(my.courpus,contral=list(weighting=weightTfldf,stopwords=TRUE))

my.df<-as.data.frame(inspect(my.tdm))

my.df.scale<-scale(my.df)

d<-dist(my.df.scale,method="euclidean")

fit<-hclust(d,method="ward")

plot(fit)

OUTPUT:

Graphical user interface, application

Description automatically generated

**Practical No: 8**

**Aim:** Write a program to compute the n-moment for a given stream where n is given.

**Description:**

For a random variable x, its Nth moment is the expected value of the Nth power of x, where N is a positive integer. The Nth moment of the deviation of x from its mean is called "the Nth central moment".

The 1st moment is the mean, the 2nd central moment is the variance.

**Steps & Output:**

import java.io.\*;

import java.util.\*;

public class n\_moment

{

public static void main(String args[]){

int n = 15;

String stream[]={"a","b","c","b","d","a","c","d","a","b","d","c","a","a","b"};

int zero\_moment=0,first\_moment=0,second\_moment=0,count=1,flag=0;

ArrayList<Integer>arrlist=new ArrayList();

System.out.println("Arraylist elements are::");

for(int i=0;i<15;i++)

{

System.out.println(stream[i]+"");

}

Arrays.sort(stream);

for(int i =1;i<n;i++)

{

if(stream[i]==stream[i-1])

{

count++;

}

else

{

arrlist.add(count);

count=1;

}

}

arrlist.add(count);

zero\_moment=arrlist.size();

System.out.println("\n\n\nValue of Zeroth moment for given stream::" +zero\_moment);

for(int i=0;i<arrlist.size();i++)

{

first\_moment+=arrlist.get(i);

}

System.out.println("\n\nValue of first moment for given stream::"+first\_moment);

for (int i=0;i<arrlist.size();i++)

{

int j=arrlist.get(i);

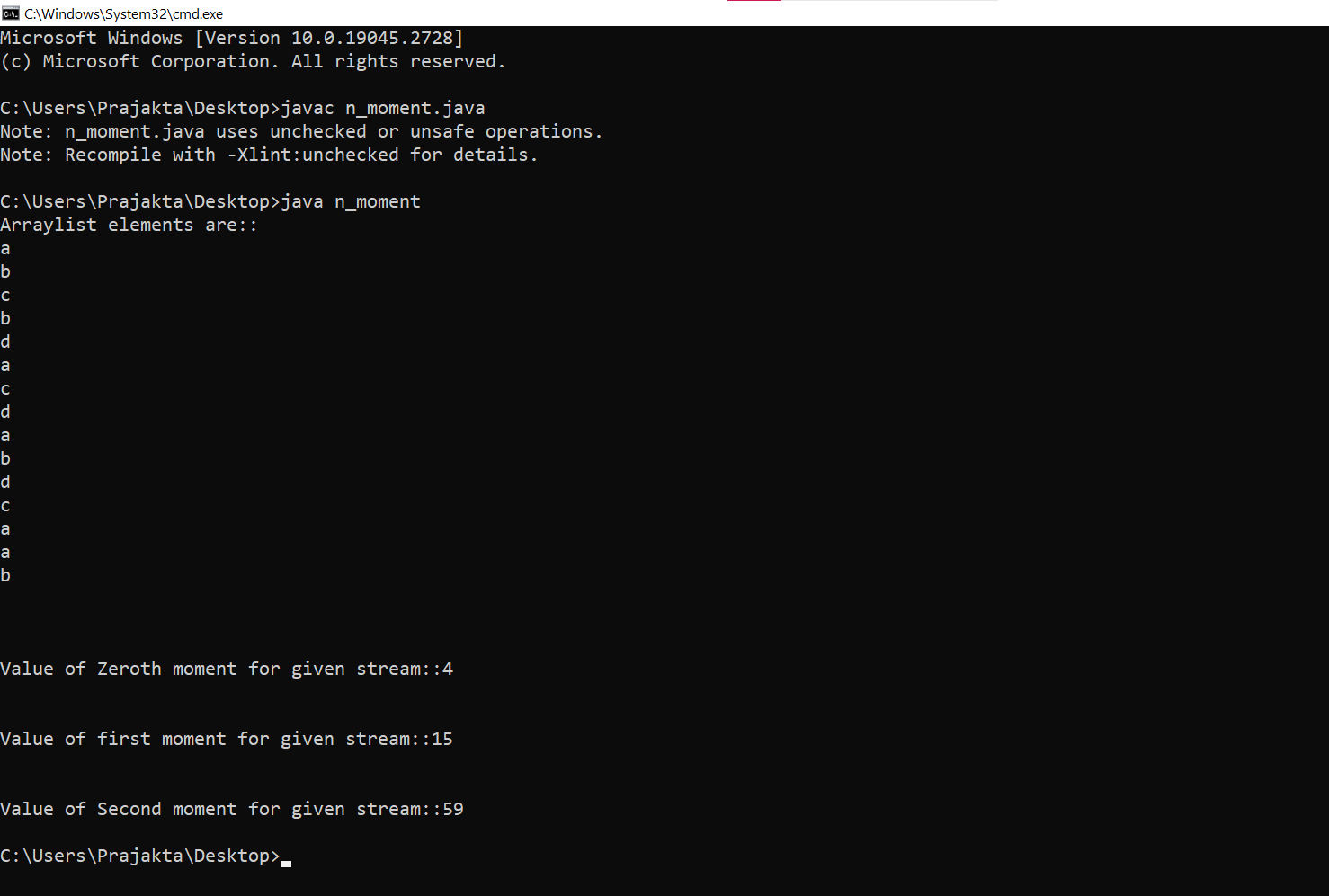
second\_moment+=(j\*j);

}

System.out.println("\n\nValue of Second moment for given stream::"+second\_moment);

}

}

OUTPUT: 

**Practical No: 9**

**Aim:** Write a program to demonstrate the Alon-Matias-Szegedy Algorithm for second moments.

**Description:**

The Alon-Matias-Szegedy Algorithm (AMS algorithm), that estimate the second moment using this formula:

E= (n \*(2 \* X.value − 1))

In which X is an univocal element of the stream, randomically selected, and X.value is a counter, that, as we read the stream, add to 1 each time we encounter another occurrence of the x element from the time we selected it.

n represents the length of the data stream

**Steps & Output:**

import java.io.\*;

import java.util.\*;

class AMSA

{

public static int findCharCount(String stream,char XE,int random,int n)

{

int countoccurance=0;

for(int i=random;i<n;i++)

{

if(stream.charAt(i)==XE)

{

countoccurance++;

}

}

return countoccurance;

}

public static int estimateValue(int XV1,int n)

{

int ExpValue;

ExpValue=n\*(2\*XV1-1);

return ExpValue;

}

public static void main(String args[])

{

int n=15;

String stream="abcbdacdabdcaab";

int random1=3,random2=8,random3=13;

char XE1,XE2,XE3;

int XV1,XV2,XV3;

int ExpValuXE1,ExpValuXE2,ExpValuXE3;

int apprSecondMomentValue;

XE1=stream.charAt(random1-1);

XE2=stream.charAt(random2-1);

XE3=stream.charAt(random3-1);

XV1=findCharCount(stream,XE1,random1-1,n);

XV2=findCharCount(stream,XE2,random2-1,n);

XV3=findCharCount(stream,XE3,random3-1,n);

System.out.println(XE1+"="+XV1+" "+XE2+"="+XV2+" "+XE3+"="+XV3);

ExpValuXE1=estimateValue(XV1,n);

ExpValuXE2=estimateValue(XV2,n);

ExpValuXE3=estimateValue(XV3,n);

System.out.println("Expected value for"+XE1+" is::"+ExpValuXE1);

System.out.println("Expected value for"+XE2+" is::"+ExpValuXE2);

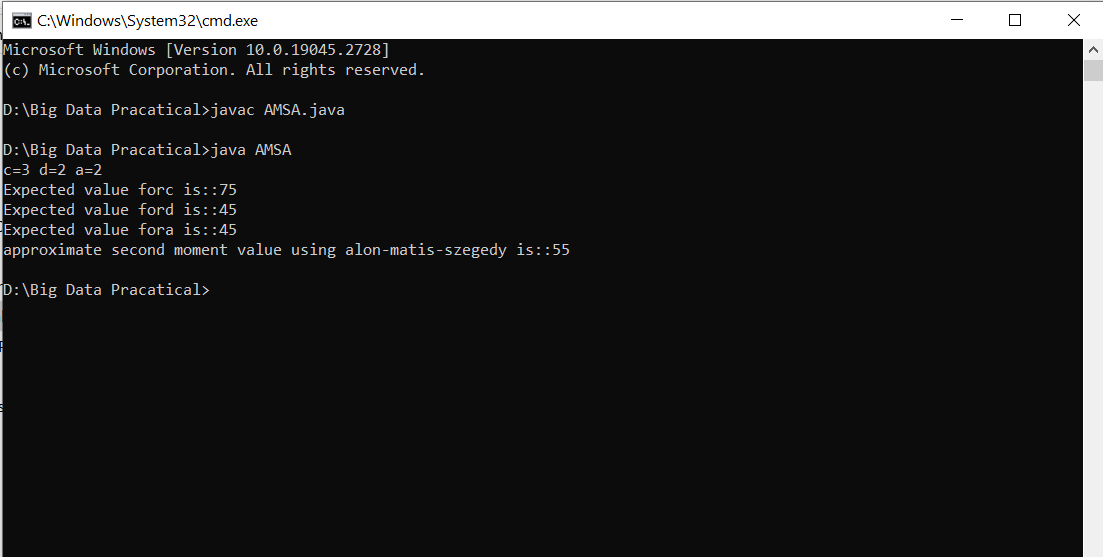
System.out.println("Expected value for"+XE3+" is::"+ExpValuXE3);

apprSecondMomentValue=(ExpValuXE1+ExpValuXE2+ExpValuXE3)/3;

System.out.println("approximate second moment value using alon-matis-szegedy is::"+apprSecondMomentValue);

}

}

**OUTPUT:**