VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

BIG DATA ANALYTICS

Submitted by

VAIBHAVI PATIL (1BM19CS217)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
May-2022 to July-2022

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "LAB COURSE **BIG DATA ANALYTICS**" carried out by **VAIBHAVI PATIL (1BM19CS217),** who is a bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Big Data Analytics - (20CS6PEBDA)** work prescribed for the said degree.

Dr Pallavi G B
Assistant professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

Index Sheet

10	**Using RDD and FlatMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark**	**55**

Course Outcome

CO1	**Apply the concept of NoSQL, Hadoop or Spark for a given task**
CO2	**Analyze the Big Data and obtain insight using data analytics mechanisms.**
CO3	**Design and implement Big data applications by applying NoSQL, Hadoop or Spark**

LAB 1

1. Perform the following DB operations using Cassandra.

1. Create a keyspace by name Employee

```
cqlsh> create keyspace Employee with replication = {
   ... 'class' : 'SimpleStrategy',
   ... 'replication_factor' : 1
   ... };
cqlsh> use Employee;
```

2. Create a column family by name

Employee-Info with attributes

Emp_Id Primary Key, Emp_Name,

Designation, Date_of_Joining, Salary, Dept_Name

```
cqlsh:employee> create table Employee_info(
          ... Emp_id int,
          ... Emp_name text,
          ... Designation text,
          ... DOJ timestamp,
          ... salary double,
          ... Dept_name text,
          ... primary key(Emp_id,salary)
          ... );
```

3. Insert the values into the table in batch

```
cqlsh:employee> begin batch
          ... insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(111,'John','Assistant professor','2022-05-11',75000,'CSE')
          ... insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(121,'Amber','Assistant professor','2022-05-11',85000,'CSE')
          ... insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(131,'Mary','Associate professor','2022-05-11',95000,'ECE')
          ... insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(141,'Jane','Associate professor','2022-05-11',105000,'ISE')
          ... insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(151,'Yelena','Associate professor','2022-05-11',95000,'ISE')
          ... apply batch;
cqlsh:employee> select * from Employee_info;


 emp_id | salary   | dept_name | designation        | doj                          |
emp_name
--------+----------+-----------+--------------------+--------------------------------+----------
   111 |    75000 |       CSE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     John
   151 |    95000 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |   Yelena
```

121 | 85000 | CSE | Assistant professor | 2022-05-10 18:30:00.000000+0000 | Amber

141 | 1.05e+05 | ISE | Associate professor | 2022-05-10 18:30:00.000000+0000 | Jane

131 | 95000 | ECE | Associate professor | 2022-05-10 18:30:00.000000+0000 | Mary

4. Update Employee name and Department of Emp-Id 121

cqlsh:employee> update Employee_info set Emp_name = 'Josh', Dept_name = 'ECE' where Emp_id = 121 and salary = 85000;

cqlsh:employee> select * from Employee_info;

```
 emp_id | salary   | dept_name | designation        | doj                         |
emp_name
--------+----------+-----------+--------------------+---------------------------------+----------
   111 |    75000 |       CSE | Assistant professor | 2022-05-10 18:30:00.000000+0000 |     John
   151 |    95000 |       ISE | Associate professor | 2022-05-10 18:30:00.000000+0000 |   Yelena
   121 |    85000 |       ECE | Assistant professor | 2022-05-10 18:30:00.000000+0000 |     Josh
   141 | 1.05e+05 |       ISE | Associate professor | 2022-05-10 18:30:00.000000+0000 |     Jane
   131 |    95000 |       ECE | Associate professor | 2022-05-10 18:30:00.000000+0000 |     Mary

(5 rows)
```

5. Sort the details of Employee records based on salary

```
cqlsh:employee> select * from Employee_info where Emp_id
in(111,121,131,141,151) order by salary desc;


 emp_id | salary   | dept_name | designation        | doj                         |
emp_name
--------+----------+-----------+--------------------+---------------------------------+----------
   141 | 1.05e+05 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Jane
   131 |    95000 |       ECE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Mary
   151 |    95000 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |   Yelena
   121 |    85000 |       ECE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     Josh
   111 |    75000 |       CSE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     John


(5 rows)
```

6. Alter the schema of the table Employee_Info to add a column Projects which stores a set of Projects done by the corresponding Employee.

```
cqlsh:employee> update Employee_info set project = project+{'AI','Data
warehouse'} where Emp_id = 111 and salary = 75000;

cqlsh:employee> update Employee_info set project = project+{'IOT','Data
warehouse'} where Emp_id = 121 and salary = 85000;
```

```
cqlsh:employee> update Employee_info set project = project+{'IOT','AI'} where
Emp_id = 131 and salary = 95000;

cqlsh:employee> update Employee_info set project = project+{'IOT','machine
learning'} where Emp_id = 141 and salary = 95000;

cqlsh:employee> update Employee_info set project = project+{'IOT','data science'}
where Emp_id = 141 and salary = 105000;

cqlsh:employee> select * from Employee_info;


 emp_id | salary   | dept_name | designation        | doj                          |
emp_name | project

--------+----------+-----------+--------------------+--------------------------------+----------+------
-----------------------

   111 |    75000 |       CSE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     John |    {'AI', 'Data warehouse'}

   151 |    95000 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |   Yelena |                      null

   121 |    85000 |       ECE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     Josh |   {'Data warehouse', 'IOT'}

   141 |    95000 |      null |               null |                          null |     null | {'IOT',
'machine learning'}

   141 | 1.05e+05 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Jane |     {'IOT', 'data science'}

   131 |    95000 |       ECE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Mary |              {'AI', 'IOT'}


(6 rows)
```

7. Update the altered table to add project names.

```
cqlsh:employee> update Employee_info set project = project+{'IOT','AI'} where
Emp_id = 151 and salary = 95000;

cqlsh:employee> select * from Employee_info;


 emp_id | salary   | dept_name | designation        | doj                          |
emp_name | project

--------+----------+-----------+--------------------+--------------------------------+----------+------
-----------------------

   111 |    75000 |       CSE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     John |    {'AI', 'Data warehouse'}

   151 |    95000 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |   Yelena |              {'AI', 'IOT'}

   121 |    85000 |       ECE | Assistant professor | 2022-05-10
18:30:00.000000+0000 |     Josh |   {'Data warehouse', 'IOT'}

   141 |    95000 |      null |               null |                         null |     null | {'IOT',
'machine learning'}

   141 | 1.05e+05 |       ISE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Jane |     {'IOT', 'data science'}

   131 |    95000 |       ECE | Associate professor | 2022-05-10
18:30:00.000000+0000 |     Mary |              {'AI', 'IOT'}


(6 rows)
```

8.Create a TTL of 15 seconds to display the values of Employees.

```
cqlsh:employee> insert into
Employee_info(Emp_id,Emp_name,Designation,DOJ,salary,Dept_name) values
(161,'Ryan','Associate professor','2022-05-11',95000,'ISE') using ttl 60;

cqlsh:employee> select ttl(Emp_name) from Employee_info where Emp_id = 161
and salary = 95000;


 ttl(emp_name)

---------------

          53


(1 rows)
```

LAB 2

2. Perform the following DB operations using Cassandra.

1.Create a keyspace by name Library

```
cqlsh> create keyspace library with replication = {
   ... 'class' : 'SimpleStrategy',
   ... 'replication_factor' : 1
   ... };
cqlsh> use library
   ... ;
```

2. Create a column family by name Library-Info with attributes

Stud_Id Primary Key, Counter_value of type Counter,

Stud_Name, Book-Name, Book-Id, Date_of_issue

```
cqlsh:library> create table library_info (
          ... stud_id int,
          ... stud_name text,
          ... book_id int,
          ... book_name text,
          ... date_of_issue timestamp,
          ... counter_value counter,
          ... primary key ((stud_id,book_id),stud_name,book_name,date_of_issue)
          ... );
```

3. Insert the values into the table in batch

```
cqlsh:library> update library_info
          ... set counter_value = counter_value+1
          ... where stud_id = 111 and stud_name = 'Raj' and book_id = 100 and
book_name = 'ADA' and date_of_issue = '2022-04-05';
cqlsh:library> update library_info
          ... set counter_value = counter_value+1
          ... where stud_id = 112 and stud_name = 'Ram' and book_id = 200 and
book_name = 'DSA' and date_of_issue = '2022-04-06';
cqlsh:library> update library_info
          ... set counter_value = counter_value+1
          ... where stud_id = 113 and stud_name = 'sohan' and book_id = 300 and
book_name = 'JAVA' and date_of_issue = '2022-04-07';
cqlsh:library> update library_info
          ... set counter_value = counter_value+1
          ... where stud_id = 114 and stud_name = 'rohan' and book_id = 400 and
book_name = 'UNIX' and date_of_issue = '2022-04-07';
```

4. Display the details of the table created and increase the value of the counter

```
cqlsh:library> select * from library_info;

 stud_id | book_id | stud_name | book_name | date_of_issue                  |
counter_value
---------+---------+-----------+-----------+--------------------------------+---------------
    114 |     400 |     rohan |      UNIX | 2022-04-06 18:30:00.000000+0000 |
1
    111 |     100 |       Raj |       ADA | 2022-04-04 18:30:00.000000+0000 |           1
```

```
    112 |     200 |      Ram |      DSA | 2022-04-05 18:30:00.000000+0000 |           1
    113 |     300 |    sohan |     JAVA | 2022-04-06 18:30:00.000000+0000 |
1
cqlsh:library> update library_info
          ... set counter_value = counter_value+1
          ... where stud_id = 114 and stud_name = 'rohan' and book_id = 400 and
book_name = 'UNIX' and date_of_issue = '2022-04-07';
```

5. Write a query to show that a student with id 114 has taken a book "UNIX" 2 times.

```
cqlsh:library> select stud_id from library_info where book_name = 'UNIX' and
counter_value = 2 allow filtering;

 stud_id
---------
    114

(1 rows)
```

6. Export the created column to a csv file

```
cqlsh:library> copy
library_info(stud_id,stud_name,book_id,book_name,date_of_issue,counter_value
) to 'd:\library_info.csv';
Using 15 child processes

Starting copy of library.library_info with columns [stud_id, stud_name, book_id,
book_name, date_of_issue, counter_value].
Processed: 4 rows; Rate:      1 rows/s; Avg. rate:      1 rows/s
```

4 rows exported to 1 files in 5.025 seconds.

7. Import a given csv dataset from local file system into Cassandra column family

```
cqlsh:library> truncate library_info;

cqlsh:library> select * from library_info;


 stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
---------+---------+-----------+-----------+---------------+---------------


(0 rows)


cqlsh:library> truncate library_info;

cqlsh:library> select * from library_info;


 stud_id | book_id | stud_name | book_name | date_of_issue | counter_value
---------+---------+-----------+-----------+---------------+---------------


(0 rows)

cqlsh:library> copy
library_info(stud_id,book_id,stud_name,book_name,date_of_issue,counter_value
) from 'd:\library_info.csv' with header = true;

Using 15 child processes


Starting copy of library.library_info with columns [stud_id, book_id, stud_name,
book_name, date_of_issue, counter_value].

Process ImportProcess-256:     1 rows/s; Avg. rate:       1 rows/s
```

```
cqlsh:library> select * from library_info;

 stud_id | book_id | stud_name | book_name | date_of_issue                    |
counter_value
---------+---------+-----------+-----------+---------------------------------+---------------
     111 |     100 |       ram |       ada | 2022-05-04 18:30:00.000000+0000 |             1
     112 |     200 |       raj |       dsa | 2022-05-05 18:30:00.000000+0000 |             2
     113 |     300 |     shyam |       ada | 2022-05-06 18:30:00.000000+0000 |
1

(3 rows)
```

LAB 3

3. MongoDB- CRUD Demonstration

```
use my_db
switched to db my_db

db.Student.insert({_id:1,name:"Michael",grade:"VII",hobbies:"reading"})
WriteResult({ "nInserted" : 1 })

db.Student.update({_id:1},{$set:{hobbies:"cricket"}},{upsert:true})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.Student.find()
{ "_id" : 1, "name" : "Michael", "grade" : "VII", "hobbies" : "cricket" }

db.Student.insert({id:1,name:"Latha",grade:"VIII",hobbies:"Singing"})
WriteResult({ "nInserted" : 1 })

db.Student.find({name:"Latha"}).pretty()
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
```

```
}

db.Student.find({},{name:1,grade:1,_id:0})
{ "name" : "Michael", "grade" : "VII" }
{ "name" : "Latha", "grade" : "VIII" }

db.Student.find({grade:{$eq:"VII"}}).pretty()
{ "_id" : 1, "name" : "Michael", "grade" : "VII", "hobbies" : "cricket" }

db.Student.find({name:/^L/}).pretty()
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
}

db.Student.find({name:/a/}).pretty()
{ "_id" : 1, "name" : "Michael", "grade" : "VII", "hobbies" : "cricket" }
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
```

```
        "grade" : "VIII",
        "hobbies" : "Singing"
}

db.Student.count()
2

db.Student.find().sort({name:1}).pretty()
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
}
{ "_id" : 1, "name" : "Michael", "grade" : "VII", "hobbies" : "cricket" }

db.Student.save({name:"Ratan",grade:"VII",_id:1})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.Student.find()
{ "_id" : 1, "name" : "Ratan", "grade" : "VII" }
{ "_id" : ObjectId("6253f120f7936958d67f3c07"), "id" : 1, "name" : "Latha",
"grade" : "VIII", "hobbies" : "Singing" }
```

```
db.Student.update({_id:1},{$set:{location:"network"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.Student.update({_id:1},{$unset:{location:"network"}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.Student.find({name:/n$/}).pretty()
{ "_id" : 1, "name" : "Ratan", "grade" : "VII" }

db.Student.find({grade:"VII"}).limit(3).pretty()
{ "_id" : 1, "name" : "Ratan", "grade" : "VII" }

db.Student.count({grade:"VIII"})
1

db.Student.find().sort({name:1}).pretty()
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
}
{ "_id" : 1, "name" : "Ratan", "grade" : "VII" }
```

```
db.Student.find().sort({name:-1}).pretty()
{ "_id" : 1, "name" : "Ratan", "grade" : "VII" }
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
}

db.Student.find().skip(1).pretty()
{
        "_id" : ObjectId("6253f120f7936958d67f3c07"),
        "id" : 1,
        "name" : "Latha",
        "grade" : "VIII",
        "hobbies" : "Singing"
}

db.createCollection("food")
{ "ok" : 1 }

db.food.insert({_id:1,fruits:['grapes','mango']})
```

```
WriteResult({ "nInserted" : 1 })

db.food.insert({_id:2,fruits:['grapes','mango','cherry']})
WriteResult({ "nInserted" : 1 })

db.food.insert({_id:3,fruits:['banana','cherry']})
WriteResult({ "nInserted" : 1 })

db.food.find({fruits:['grapes','mango']})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }

db.food.find({'fruits':{$size:2}})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
{ "_id" : 3, "fruits" : [ "banana", "cherry" ] }

db.food.find({_id:2},{'fruits':{$slice:2}})
{ "_id" : 2, "fruits" : [ "grapes", "mango" ] }

db.food.find({fruits:{$all:['grapes','mango']}})
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }

db.food.update({_id:3},{$set:{'fruits.1':'apple'}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

```
db.food.find()
{ "_id" : 1, "fruits" : [ "grapes", "mango" ] }
{ "_id" : 2, "fruits" : [ "grapes", "mango", "cherry" ] }
{ "_id" : 3, "fruits" : [ "banana", "apple" ] }

db.food.update({_id:2},{$push:{price:{grapes:80,mango:200,cherry:100}}})
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })

db.createCollection("Customers")
{ "ok" : 1 }

db.Customers.insert({custId:1,acctBal:1000,acctType:"current"})
WriteResult({ "nInserted" : 1 })

db.Customers.insert({custId:2,acctBal:2000,acctType:"current"})
WriteResult({ "nInserted" : 1 })

db.Customers.insert({custId:3,acctBal:3000,acctType:"savings"})
WriteResult({ "nInserted" : 1 })

db.Customers.aggregate({$group:{_id:"$custId",toAcctBal:{$sum:"$acctBal"}}})
{ "_id" : 3, "toAcctBal" : 3000 }
{ "_id" : 1, "toAcctBal" : 1000 }
```

```
{ "_id" : 2, "toAcctBal" : 2000 }
```

```
db.Customers.aggregate({$match:{acctType:"current"}},{$group:{_id:"$custId",toAcctBal:{$sum:"$acctBal"}}})
{ "_id" : 2, "toAcctBal" : 2000 }
{ "_id" : 1, "toAcctBal" : 1000 }
```

```
db.Customers.aggregate({$match:{acctType:"current"}},{$group:{_id:"$custId",toAcctBal:{$sum:"$acctBal"}}},{$match:{toAcctBal:{$gt:500}}})
{ "_id" : 2, "toAcctBal" : 2000 }
{ "_id" : 1, "toAcctBal" : 1000 }
```

```
db.Student.drop()
false
```

LAB 4

4.Screenshot of Hadoop installed

```
Command Prompt
Microsoft Windows [Version 10.0.22000.739]
(c) Microsoft Corporation. All rights reserved.

C:\Users\VAIBHAVI PATIL>cd ..

C:\Users>cd ..

C:\>cd hadoop-3.3.3\sbin

C:\hadoop-3.3.3\sbin>start-all.cmd
This script is Deprecated. Instead use start-dfs.cmd and start-yarn.cmd
starting yarn daemons

C:\hadoop-3.3.3\sbin>jps
13472 NameNode
21024 DataNode
22736 Jps
15972 NameNode
29364 ResourceManager
38956 NodeManager

C:\hadoop-3.3.3\sbin>
```

LAB 5

5. Execution of HDFS Commands for interaction with Hadoop Environment. (Minimum 10 commands to be executed)

```
Activities   Terminal ▾                    Jun 3 15:56
hduser@bmsce-OptiPlex-3060: ~
drwxr-xr-x   - hduser supergroup          0 2022-06-03 12:06 /demo
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:47 /one
drwxr-xr-x   - hduser supergroup          0 2022-06-01 15:12 /rahil
drwxr-xr-x   - hduser supergroup          0 2022-05-31 10:34 /rits
drwxr-xr-x   - hduser supergroup          0 2022-06-03 12:31 /rohit
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:42 /test1
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:29 /test2
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:47 /two
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:27 /vaib1
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:27 /vaibhavi
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -put /home/bmsce/file.txt /one/copied1.txt
22/06/03 15:48:23 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:48:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:48 /one/copied1.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyFromLocal /home/bmsce/file.txt /one/copied2.txt
22/06/03 15:49:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:49:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:48 /one/copied1.txt
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:49 /one/copied2.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -get /one/copied1.txt Desktop/copy1
22/06/03 15:50:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ ls Desktop
ask.txt  copy1  file1_copy  file.c  file_copy  output.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /one/copied.txt Desktop/copy2
22/06/03 15:51:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
copyToLocal: `/one/copied.txt': No such file or directory
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /one/copied1.txt Desktop/copy2
22/06/03 15:51:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ ls Desktop
ask.txt  copy1  copy2  file1_copy  file.c  file_copy  output.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -cp /one/copied1.txt /two/copied2.txt
22/06/03 15:52:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /two
22/06/03 15:52:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:52 /two/copied2.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -mv /one/copied1.txt /two/copiedusingmv.txt
22/06/03 15:53:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:53:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:49 /one/copied2.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /two
22/06/03 15:53:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:52 /two/copied2.txt
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:48 /two/copiedusingmv.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -cat /two/copied2.txt
22/06/03 15:54:04 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
bda lab
hduser@bmsce-OptiPlex-3060:~$
```

```
Activities   Terminal ▾                    Jun 3 15:55
hduser@bmsce-OptiPlex-3060: ~
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /test1/copied1.txt /Desktop/file1_copy
22/06/03 15:44:31 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
copyToLocal: `/Desktop/file1_copy': No such file or directory
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /test1/copied1.txt Desktop/file1_copy
22/06/03 15:45:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -mkdir /one /two
22/06/03 15:47:06 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /
22/06/03 15:47:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 12 items
drwxr-xr-x   - hduser supergroup          0 2022-06-01 14:44 /abc
-rw-r--r--   1 hduser supergroup         18 2022-06-01 14:54 /abc.txt
drwxr-xr-x   - hduser supergroup          0 2022-06-03 12:06 /demo
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:47 /one
drwxr-xr-x   - hduser supergroup          0 2022-06-01 15:12 /rahil
drwxr-xr-x   - hduser supergroup          0 2022-05-31 10:34 /rits
drwxr-xr-x   - hduser supergroup          0 2022-06-03 12:31 /rohit
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:42 /test1
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:29 /test2
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:47 /two
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:27 /vaib1
drwxr-xr-x   - hduser supergroup          0 2022-06-03 15:27 /vaibhavi
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -put /home/bmsce/file.txt /one/copied1.txt
22/06/03 15:48:23 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:48:50 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:48 /one/copied1.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyFromLocal /home/bmsce/file.txt /one/copied2.txt
22/06/03 15:49:34 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:49:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:48 /one/copied1.txt
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:49 /one/copied2.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -get /one/copied1.txt Desktop/copy1
22/06/03 15:50:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ ls Desktop
ask.txt  copy1  file1_copy  file.c  file_copy  output.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /one/copied.txt Desktop/copy2
22/06/03 15:51:30 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
copyToLocal: `/one/copied.txt': No such file or directory
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -copyToLocal /one/copied1.txt Desktop/copy2
22/06/03 15:51:44 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ ls Desktop
ask.txt  copy1  copy2  file1_copy  file.c  file_copy  output.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -cp /one/copied1.txt /two/copied2.txt
22/06/03 15:52:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /two
22/06/03 15:52:43 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 1 items
-rw-r--r--   1 hduser supergroup          8 2022-06-03 15:52 /two/copied2.txt
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -mv /one/copied1.txt /two/copiedusingmv.txt
22/06/03 15:53:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
hduser@bmsce-OptiPlex-3060:~$ hdfs dfs -ls /one
22/06/03 15:53:24 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

LAB 6

6. From the following link extract the weather data

https://github.com/tomwhite/hadoop-book/tree/master/input/ncdc/all

Create a Map Reduce program to

a) find average temperature for each year from the NCDC data set.

```
// AverageDriver.java package temperature;


import org.apache.hadoop.io.*; import org.apache.hadoop.fs.*; import
org.apache.hadoop.mapreduce.*; import
org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;


public class AverageDriver

{		public static void main (String[] args) throws Exception

	{

		if (args.length != 2)

		{

			System.err.println("Please Enter the input and output
parameters");

			System.exit(-1);

		}
```

```
                Job job = new Job();
job.setJarByClass(AverageDriver.class);                job.setJobName("Max
temperature");

                FileInputFormat.addInputPath(job,new Path(args[0]));

                FileOutputFormat.setOutputPath(job,new Path (args[1]));



                job.setMapperClass(AverageMapper.class);
job.setReducerClass(AverageReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true)?0:1);

        }

}



//AverageMapper.java package temperature;



import org.apache.hadoop.io.*; import org.apache.hadoop.mapreduce.*; import
java.io.IOException;



public class AverageMapper extends Mapper <LongWritable, Text, Text,
IntWritable>

{ public static final int MISSING = 9999;



public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException
```

```
{

        String line = value.toString();      String year = line.substring(15,19);
int temperature;     if (line.charAt(87)=='+')                              temperature =
Integer.parseInt(line.substring(88, 92));

        else

                temperature = Integer.parseInt(line.substring(87, 92));  String quality
= line.substring(92, 93);    if(temperature != MISSING &&
quality.matches("[01459]"))                        context.write(new Text(year),new
IntWritable(temperature)); }

}


//AverageReducer.java package temperature;


import org.apache.hadoop.io.IntWritable; import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.*; import java.io.IOException;


public class AverageReducer extends Reducer <Text, IntWritable,Text, IntWritable>

{

        public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException,InterruptedException

        {

                int max_temp = 0;                    int count = 0;

                for (IntWritable value : values)

                {
```

```
                max_temp += value.get();
                count+=1;
            }
            context.write(key, new IntWritable(max_temp/count));
        }
}
```

```
c:\hadoop_new\sbin>hdfs dfs -cat /tempAverageOutput/part-r-00000
1901    46
1949    94
1950    3
```

b) find the mean max temperature for every month

```
//TempDriver.java package temperatureMax;

import org.apache.hadoop.io.*; import
org.apache.hadoop.fs.*; import
org.apache.hadoop.mapreduce.*; import
org.apache.hadoop.mapreduce.lib.input.FileInputForma
t; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFo
rmat;

public class TempDriver
```

```
{                   public static void main (String[] args)
throws Exception

                    {
                              if (args.length != 2)
                              {
                                        System.err.println("Please Enter the input and output
parameters");
                                        System.exit(-1);
                              }
                    Job job = new Job();
job.setJarByClass(TempDriver.class);     job.setJobName("Max
temperature");

                              FileInputFormat.addInputPath(job,new Path(args[0]));
                              FileOutputFormat.setOutputPath(job,new Path (args[1]));


                    job.setMapperClass(TempMapper.class);
job.setReducerClass(TempReducer.class);
                    job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
System.exit(job.waitForCompletion(true)?0:1);

                    }
}
```

```
//TempMapper.java
package
temperatureMax;


import org.apache.hadoop.io.*;
import
org.apache.hadoop.mapreduce.*
; import java.io.IOException;


public class TempMapper extends Mapper <LongWritable, Text, Text, IntWritable>
{ public static final int MISSING =
9999;


public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException
{
        String line = value.toString();     String month =
line.substring(19,21);    int temperature;                if
(line.charAt(87)=='+')                          temperature =
Integer.parseInt(line.substring(88, 92));

                    else
```

```
                    temperature = Integer.parseInt(line.substring(87,
92));  String quality = line.substring(92, 93);     if(temperature !=
MISSING && quality.matches("[01459]"))
context.write(new Text(month),new IntWritable(temperature)); }

}


//TempReducer.java
package
temperatureMax;


import org.apache.hadoop.io.*;
import
org.apache.hadoop.mapreduce.*
; import java.io.IOException;


public class TempMapper extends Mapper <LongWritable, Text, Text, IntWritable>
{ public static final int MISSING =
9999;


public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException
{
```

```
String line = value.toString();      String month =
line.substring(19,21);     int temperature;                if
(line.charAt(87)=='+')                          temperature =
Integer.parseInt(line.substring(88, 92));

                else

                temperature = Integer.parseInt(line.substring(87,
92));  String quality = line.substring(92, 93);     if(temperature !=
MISSING && quality.matches("[01459]"))
context.write(new Text(month),new IntWritable(temperature));

                }

}
```

```
c:\hadoop_new\sbin>hdfs dfs -cat /tempMaxOutput/part-r-00000
01      44
02      17
03      111
04      194
05      256
06      278
07      317
08      283
09      211
10      156
11      89
12      117
```

LAB 7

For a given Text file, create a Map Reduce program to sort the content in an alphabetic order listing only top 'n' maximum occurrence of words.

```
// TopN.java package sortWords;


import org.apache.hadoop.conf.Configuration; import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable; import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job; import
org.apache.hadoop.mapreduce.Mapper; import
org.apache.hadoop.mapreduce.Reducer; import
org.apache.hadoop.mapreduce.lib.input.FileInputFormat; import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat; import
org.apache.hadoop.util.GenericOptionsParser; import utils.MiscUtils;


import java.io.IOException; import java.util.*;


public class TopN {


   public static void main(String[] args) throws Exception {

      Configuration conf = new Configuration();

      String[] otherArgs = new GenericOptionsParser(conf,
args).getRemainingArgs();         if (otherArgs.length != 2) {

         System.err.println("Usage: TopN <in> <out>");

         System.exit(2);
```

```
        }

        Job job = Job.getInstance(conf);        job.setJobName("Top N");
job.setJarByClass(TopN.class);              job.setMapperClass(TopNMapper.class);
//job.setCombinerClass(TopNReducer.class);
job.setReducerClass(TopNReducer.class);      job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);

        FileInputFormat.addInputPath(job, new Path(otherArgs[0]));

        FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);

    }



    /**

     * The mapper reads one line at the time, splits it into an array of single words
and emits every      * word to the reducers with the value of 1.

     */

    public static class TopNMapper extends Mapper<Object, Text, Text, IntWritable>
{



        private final static IntWritable one = new IntWritable(1);     private Text word
= new Text();

        private String tokens = "[_|$#<>\\^=\\[\\]\\*/\\\\,;,.\\-:()?!\"']";



        @Override

        public void map(Object key, Text value, Context context) throws IOException,
```

```
InterruptedException {

        String cleanLine = value.toString().toLowerCase().replaceAll(tokens, " ");
        StringTokenizer itr = new StringTokenizer(cleanLine);           while
(itr.hasMoreTokens()) {

            word.set(itr.nextToken().trim());           context.write(word, one);

        }

      }

   }



   /**

    * The reducer retrieves every word and puts it into a Map: if the word already
exists in the            * map, increments its value, otherwise sets it to 1.

    */

   public static class TopNReducer extends Reducer<Text, IntWritable, Text,
IntWritable> {



      private Map<Text, IntWritable> countMap = new HashMap<>();



      @Override

      public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {



        // computes the number of occurrences of a single word           int sum = 0;
        for (IntWritable val : values) {               sum += val.get();
```

```
        }

        // puts the number of occurrences of this word into the map.

        // We need to create another Text object because the Text instance

        // we receive is the same for all the words            countMap.put(new
Text(key), new IntWritable(sum));

      }

@Override

      protected void cleanup(Context context) throws IOException,
InterruptedException {



        Map<Text, IntWritable> sortedMap = MiscUtils.sortByValues(countMap);



        int counter = 0;              for (Text key : sortedMap.keySet()) {            if
(counter++ == 3) {                    break;

                }

                context.write(key, sortedMap.get(key));

        }

      }

   }



   /**

    * The combiner retrieves every word and puts it into a Map: if the word already
exists in the          * map, increments its value, otherwise sets it to 1.
```

```
    */

   public static class TopNCombiner extends Reducer<Text, IntWritable, Text,
IntWritable> {



       @Override

       public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {



          // computes the number of occurrences of a single word            int sum = 0;
             for (IntWritable val : values) {                    sum += val.get();

          }

          context.write(key, new IntWritable(sum));

}

   }

}



// MiscUtils.java package utils;



import java.util.*;



public class MiscUtils {
```

```
   /**
sorts the map by values. Taken from:
http://javarevisited.blogspot.it/2012/12/how-to-sort-hashmap-java-by-key-and-value.html
    */
   public static <K extends Comparable, V extends Comparable> Map<K, V> sortByValues(Map<K, V> map) {
       List<Map.Entry<K, V>> entries = new LinkedList<Map.Entry<K, V>>(map.entrySet());


       Collections.sort(entries, new Comparator<Map.Entry<K, V>>() {


           @Override            public int compare(Map.Entry<K, V> o1, Map.Entry<K, V> o2) {                return o2.getValue().compareTo(o1.getValue());
           }
       });


       //LinkedHashMap will keep the keys in the order they are inserted
       //which is currently sorted on natural ordering
       Map<K, V> sortedMap = new LinkedHashMap<K, V>();
for (Map.Entry<K, V> entry : entries) {
           sortedMap.put(entry.getKey(), entry.getValue());
       }
```

```
        return sortedMap;

    }

}
```

OUTPUT

```
C:\hadoop_new\share\hadoop\mapreduce>hdfs dfs -cat \sortwordsOutput\part-r-00000
car     7
deer    6
bear    3
```

LAB 8

Create a Hadoop Map Reduce program to combine information from the users file along with Information from the posts file by using the concept of join and display user_id, Reputation and Score.

```
// JoinDriver.java import org.apache.hadoop.conf.Configured; import
org.apache.hadoop.fs.Path; import org.apache.hadoop.io.Text; import
org.apache.hadoop.mapred.*; import
org.apache.hadoop.mapred.lib.MultipleInputs; import org.apache.hadoop.util.*;



public class JoinDriver extends Configured implements Tool {


	public static class KeyPartitioner implements Partitioner<TextPair, Text> {

			@Override

			public void configure(JobConf job) {}


			@Override

	public int getPartition(TextPair key, Text value, int numPartitions) {
return (key.getFirst().hashCode() & Integer.MAX_VALUE) % numPartitions;

			}

	}
```

```
@Override public int run(String[] args) throws Exception {                    if
(args.length != 3) {

                        System.out.println("Usage: <Department Emp Strength input>

<Department Name input> <output>");

                        return -1;

                }


                JobConf conf = new JobConf(getConf(), getClass());
conf.setJobName("Join 'Department Emp Strength input' with 'Department Name
input'");


                Path AInputPath = new Path(args[0]);

                Path BInputPath = new Path(args[1]);

                Path outputPath = new Path(args[2]);


                MultipleInputs.addInputPath(conf, AInputPath,
TextInputFormat.class,

Posts.class);

                MultipleInputs.addInputPath(conf, BInputPath,
TextInputFormat.class,

User.class);


                FileOutputFormat.setOutputPath(conf, outputPath);
```

```
        conf.setPartitionerClass(KeyPartitioner.class);
conf.setOutputValueGroupingComparator(TextPair.FirstComparator.class);
            conf.setMapOutputKeyClass(TextPair.class);
            conf.setReducerClass(JoinReducer.class);
            conf.setOutputKeyClass(Text.class);
        JobClient.runJob(conf);
        return 0;
    }
    public static void main(String[] args) throws Exception {
        int exitCode = ToolRunner.run(new JoinDriver(), args);
        System.exit(exitCode);
    }
}
```

```
// JoinReducer.java import java.io.IOException; import java.util.Iterator;


import org.apache.hadoop.io.Text; import org.apache.hadoop.mapred.*;


public class JoinReducer extends MapReduceBase implements Reducer<TextPair,
Text, Text, Text> {


        @Override

        public void reduce (TextPair key, Iterator<Text> values,
OutputCollector<Text, Text> output, Reporter reporter)

                        throws IOException

        {


                Text nodeId = new Text(values.next());  while (values.hasNext()) {

                        Text node = values.next();

                Text outValue = new Text(nodeId.toString() + "\t\t" + node.toString());
                output.collect(key.getFirst(), outValue);

                }

        }

}
```

```
// User.java import java.io.IOException; import java.util.Iterator; import
org.apache.hadoop.conf.Configuration; import
org.apache.hadoop.fs.FSDataInputStream; import
org.apache.hadoop.fs.FSDataOutputStream; import
org.apache.hadoop.fs.FileSystem; import org.apache.hadoop.fs.Path; import
org.apache.hadoop.io.LongWritable; import org.apache.hadoop.io.Text; import
org.apache.hadoop.mapred.*;


import org.apache.hadoop.io.IntWritable;


public class User extends MapReduceBase implements Mapper<LongWritable,
Text, TextPair, Text> {


        @Override

 public void map(LongWritable key, Text value, OutputCollector<TextPair, Text>
output, Reporter reporter)

                        throws IOException

        {


                String valueString = value.toString();

                String[] SingleNodeData = valueString.split("\t");

        output.collect(new TextPair(SingleNodeData[0], "1"), new

Text(SingleNodeData[1]));

        }
```

```
}

//Posts.java import java.io.IOException;

import org.apache.hadoop.io.*; import org.apache.hadoop.mapred.*;

public class Posts extends MapReduceBase implements Mapper<LongWritable,
Text, TextPair, Text>  {

        @Override

 public void map(LongWritable key, Text value, OutputCollector<TextPair, Text>
output, Reporter reporter)

                                throws IOException

        {

                String valueString = value.toString();

                String[] SingleNodeData = valueString.split("\t");
output.collect(new TextPair(SingleNodeData[3], "0"), new

Text(SingleNodeData[9]));

        }

}

// TextPair.java import java.io.*;
```

```
import org.apache.hadoop.io.*;

public class TextPair implements WritableComparable<TextPair> {


  private Text first;   private Text second;


  public TextPair() { set(new Text(), new Text());

  }


  public TextPair(String first, String second) {  set(new Text(first), new
Text(second));

  }


  public TextPair(Text first, Text second) {       set(first, second);

  }


  public void set(Text first, Text second) {       this.first = first;       this.second =
second;

  }


  public Text getFirst() {     return first;

  }
```

```
  public Text getSecond() {          return second;

  }


  @Override

  public void write(DataOutput out) throws IOException { first.write(out);
second.write(out);

  }


  @Override   public void readFields(DataInput in) throws IOException {
first.readFields(in);  second.readFields(in);

  }


  @Override   public int hashCode() {   return first.hashCode() * 163 +
second.hashCode();

  }


  @Override   public boolean equals(Object o) {     if (o instanceof TextPair) {
TextPair tp = (TextPair) o;              return first.equals(tp.first) &&
second.equals(tp.second);

          }     return false;

  }
```

```
  @Override   public String toString() {     return first + "\t" + second;

  }


  @Override

  public int compareTo(TextPair tp) {     int cmp = first.compareTo(tp.first);       if
(cmp != 0) {             return cmp;

          }

          return second.compareTo(tp.second);

  }

  // ^^ TextPair


  // vv TextPairComparator   public static class Comparator extends
WritableComparator {


          private static final Text.Comparator TEXT_COMPARATOR = new
Text.Comparator();


          public Comparator() {       super(TextPair.class);

          }


          @Override   public int compare(byte[] b1, int s1, int l1,                    byte[]
b2, int s2, int l2) {

                  try {
```

```
        int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);            int
cmp = TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);        if (cmp !=
0) {            return cmp;

        }

        return TEXT_COMPARATOR.compare(b1, s1 + firstL1, l1 - firstL1,

                                b2, s2 + firstL2, l2 - firstL2);

        } catch (IOException e) {            throw new IllegalArgumentException(e);

        }

        }

  }


  static {

        WritableComparator.define(TextPair.class, new Comparator());

  }

  public static class FirstComparator extends WritableComparator {


        private static final Text.Comparator TEXT_COMPARATOR = new
Text.Comparator();


        public FirstComparator() {         super(TextPair.class);

        }
```

```
        @Override   public int compare(byte[] b1, int s1, int l1,                    byte[]
b2, int s2, int l2) {

        try {

        int firstL1 = WritableUtils.decodeVIntSize(b1[s1]) + readVInt(b1, s1);
int firstL2 = WritableUtils.decodeVIntSize(b2[s2]) + readVInt(b2, s2);            return
TEXT_COMPARATOR.compare(b1, s1, firstL1, b2, s2, firstL2);

        } catch (IOException e) {            throw new IllegalArgumentException(e);

        }

        }



        @Override

        public int compare(WritableComparable a, WritableComparable b) {   if (a
instanceof TextPair && b instanceof TextPair) {       return ((TextPair)
a).first.compareTo(((TextPair) b).first);

        }

        return super.compare(a, b);

        }

  }

}
```

LAB 9

Program to print word count on scala shell and print “Hello world” on scala IDE

```
val data=sc.textFile("sparkdata.txt")

data.collect;

val splitdata = data.flatMap(line => line.split(" "));

splitdata.collect;

val mapdata = splitdata.map(word => (word,1));

mapdata.collect;

val reducedata = mapdata.reduceByKey(_+_);

reducedata.collect;
```

```
scala> reducedata.collect;
res8: Array[(String, Int)] = Array(("",1), (hello,5), (lab,3), (begin,3), (spark
,5), (9,1))
```

LAB 10

10. Using RDD and FlaMap count how many times each word appears in a file and write out a list of words whose count is strictly greater than 4 using Spark.

```
Activities   Terminal ▾                    Jul 9 12:51
bmsce@bmsce-OptiPlex-3060: ~
bmsce@bmsce-OptiPlex-3060:~$ spark-shell
22/07/09 12:48:38 WARN Utils: Your hostname, bmsce-OptiPlex-3060 resolves to a loopback address: 127.0.1.1; using 10.124.7.73 instead (on interface enp1s0)
22/07/09 12:48:38 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
22/07/09 12:48:38 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
22/07/09 12:48:41 WARN Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
22/07/09 12:48:41 WARN Utils: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.
Spark context Web UI available at http://10.124.7.73:4042
Spark context available as 'sc' (master = local[*], app id = local-1657351121489).
Spark session available as 'spark'.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 2.4.8
      /_/

Using Scala version 2.11.12 (OpenJDK 64-Bit Server VM, Java 1.8.0_312)
Type in expressions to have them evaluated.
Type :help for more information.

scala> val textFile = sc.textFile("sparkdata2.txt")
textFile: org.apache.spark.rdd.RDD[String] = sparkdata2.txt MapPartitionsRDD[1] at textFile at <console>:24

scala> val counts = textFile.flatMap(line => line.split(" ")).map(word => (word, 1)).reduceByKey(_ + _)
counts: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[4] at reduceByKey at <console>:25

scala> import scala.collection.immutable.ListMap
import scala.collection.immutable.ListMap

scala> val sorted=ListMap(counts.collect.sortWith(_._2 > _._2):_*)
sorted: scala.collection.immutable.ListMap[String,Int] = Map(BMS -> 5, COLLEGE -> 3, LAW -> 1, ARCHITECTURE -> 1, ENGINEERING -> 1)

scala> println(sorted)
Map(BMS -> 5, COLLEGE -> 3, LAW -> 1, ARCHITECTURE -> 1, ENGINEERING -> 1)

scala> for((k,v)<-sorted)
     | {
     | if(v>4)
     | {
     | print(k+",")
     | print(v)
     | println()
     | }
     | }
BMS,5

scala>
```