Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

Practical 4:- To determine maximum temperature using Hadoop MapReduce

Map Reduce : works by breaking the processing into two phases: the map phase and the reduce phase. Each phase has key-value pairs as input and output, the types of which may be chosen by the programmer. The programmer also specifies two functions: the map function and the reduce function.

Hadoop MaxTemprature operation occurs in 3 stages –

Mapper Phase

Reducer Phase

Driver code

Dataset – temperature.txt

0067011990999991950051507004+68750+023550FM-12+038299999V0203301N00671220001CN9999999N9+00001+99999999999

0043011990999991950051512004+68750+023550FM-12+038299999V0203201N00671220001CN9999999N9+00221+99999999999

0043011990999991950051518004+68750+023550FM-12+038299999V0203201N00261220001CN9999999N9-00111+99999999999

0043012650999991949032412004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+01111+99999999999

0043012650999991949032418004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+00781+99999999999

These lines are presented to the map function as the key-value pairs

(0, 0067011990999991950051507004...9999999N9+00001+99999999999...) (106, 0043011990999991950051512004...9999999N9+00221+99999999999...) (212, 0043011990999991950051518004...9999999N9-00111+99999999999...) (318, 0043012650999991949032412004...0500001N9+01111+99999999999...) (424, 0043012650999991949032418004...0500001N9+00781+99999999999...)

The keys are the line offsets within the file, which we ignore in our map function. The map function merely extracts the year and the air temperature (indicated in bold text), and emits them as its output (the temperature values have been interpreted as integers):

(1950, 0)

(1950, 22)

Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

The output from the map function is processed by the MapReduce framework before being sent to the reduce function. This processing sorts and groups the key-value pairs by key. So, continuing the example, our reduce function sees the following input:

(1949, [111, 78])

(1950, [0, 22, −11])

Each year appears with a list of all its air temperature readings. All the reduce function has to do now is iterate through the list and pick up the maximum reading:

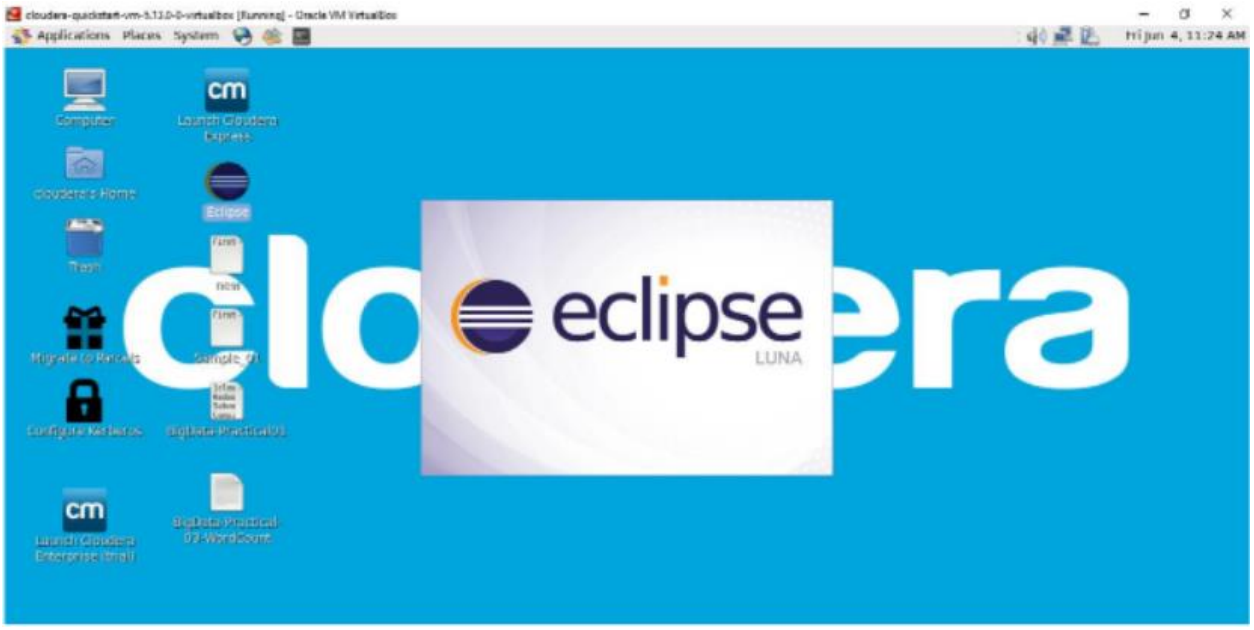
(1949, 111)

(1950, 22)

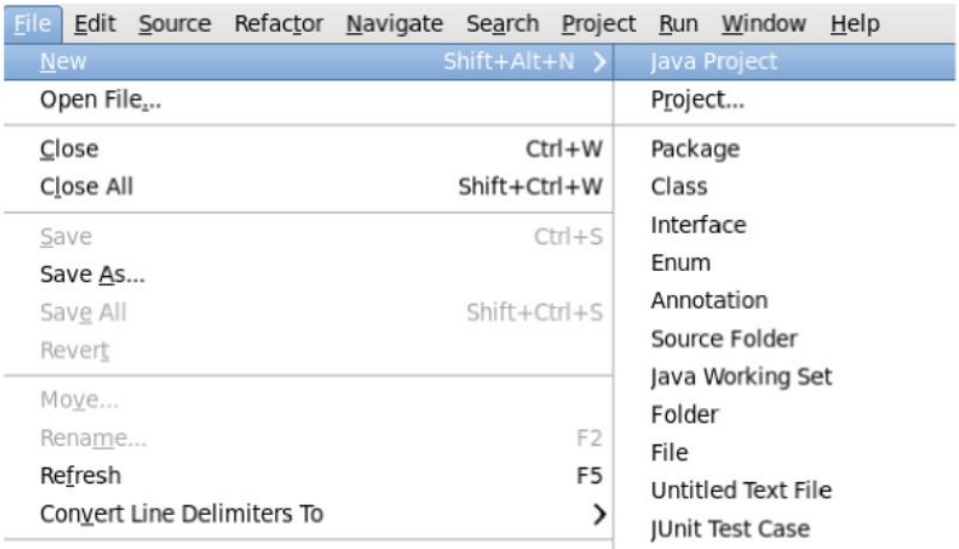
1) Open virtual box and then start cloudera quickstart

2) Open Eclipse present on the cloudera desktop

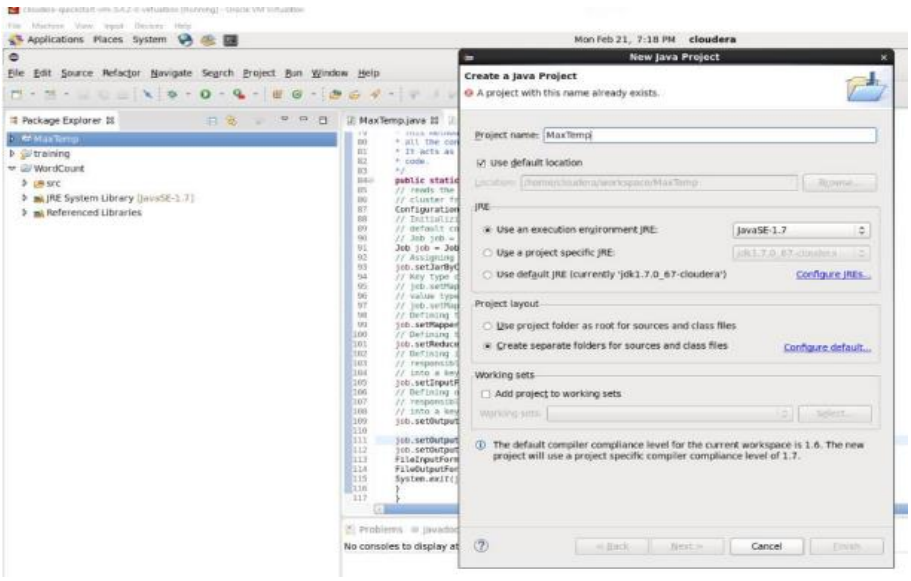
Name: sahil shaikh Mushtaq Ahmed
Roll No: 47



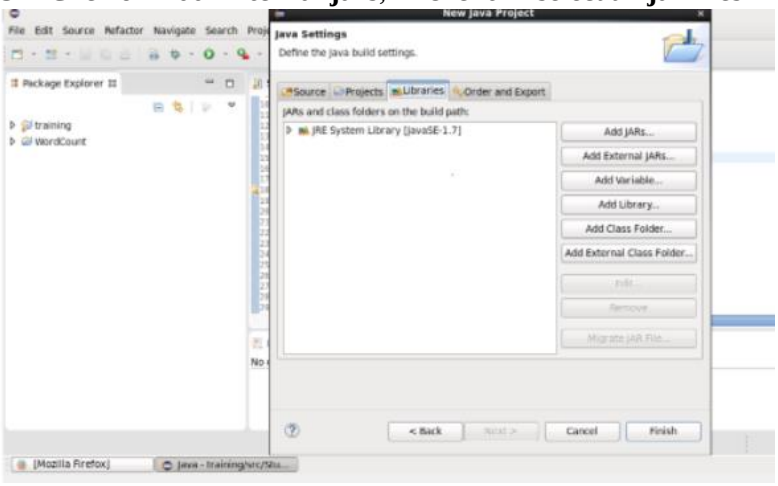
3) Create a new Java project clicking: **File** -> **New** -> **Project** -> **Java Project** -> Next ("**MaxTemp**" is
the project name).



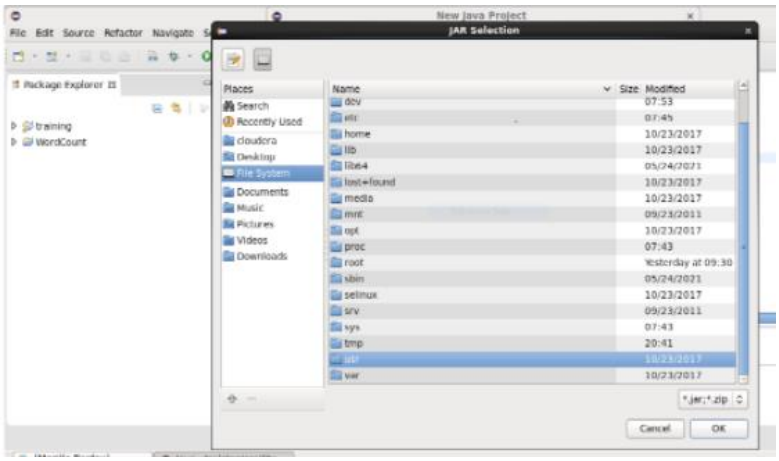
Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

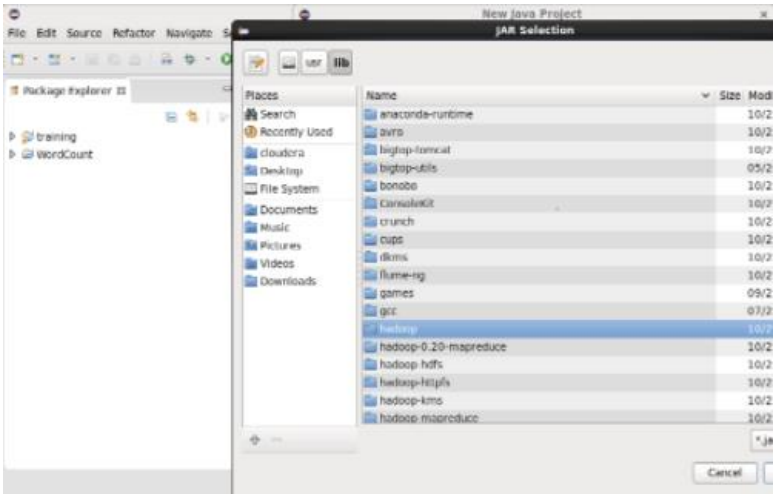


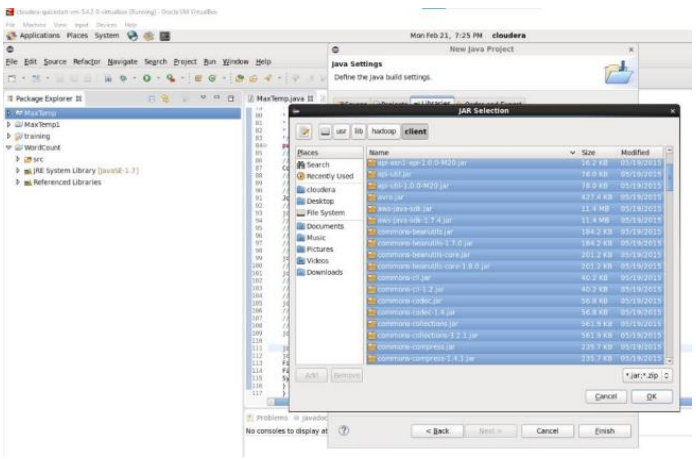
4) Adding the Hadoop libraries to the project Click on **Libraries** -> Add External JARs Click on **File System** -> **usr** -> **lib** -> Hadoop Select all the **libraries (JAR Files)** -> click **OK** Click on Add **External jars**, -> **client** -> select all **jar files** -> **ok** -> **Finish.**



Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

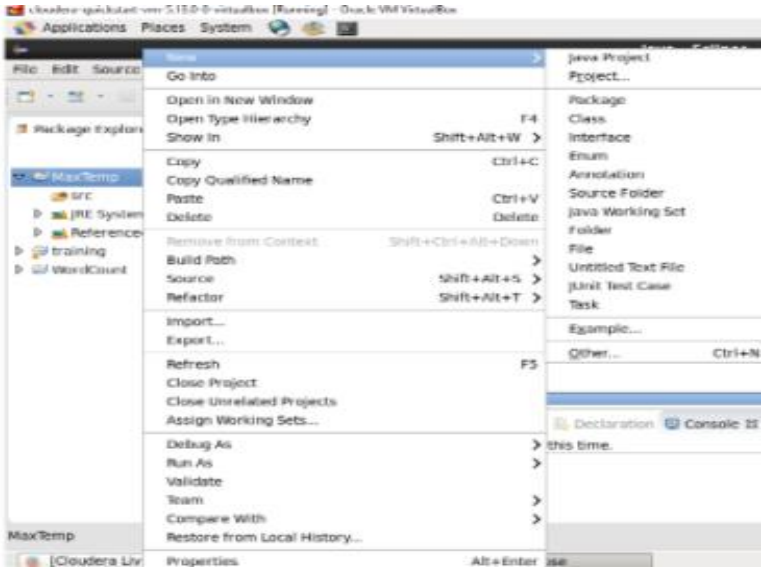


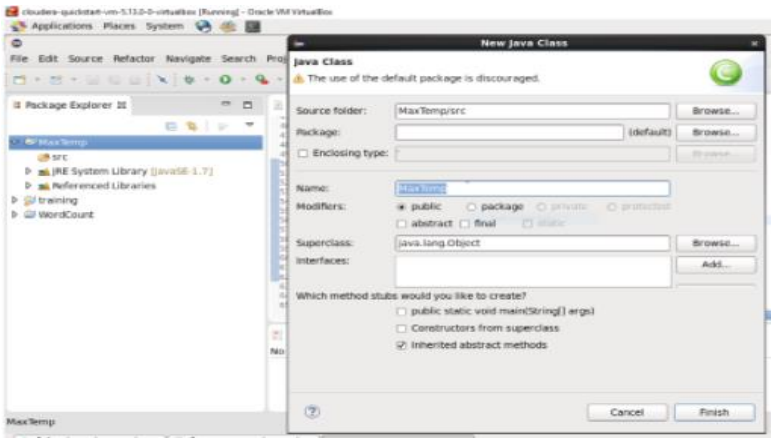


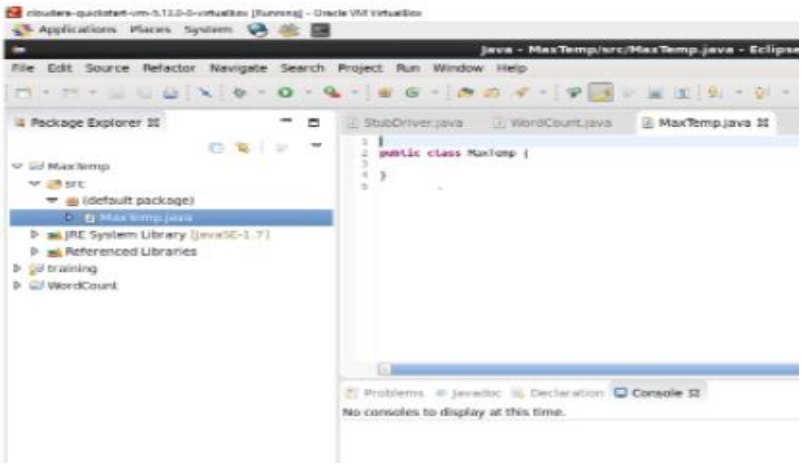


5) Right Click on the name of Project “MaxTemp” -> New -> class don’t write anything for package Write Name Textbox write “MaxTemp” -> Finish Then MaxTemp.java window will pop up.

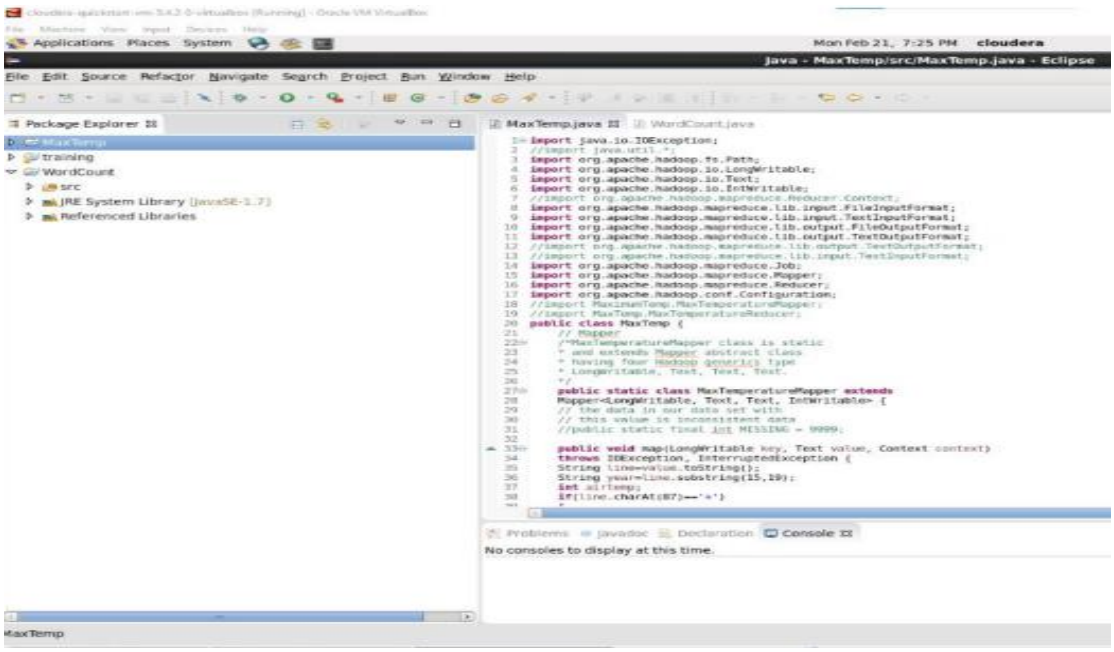
Name: sahil shaikh Mushtaq Ahmed
Roll No: 47







Name: sahil shaikh Mushtaq Ahmed
Roll No: 47



Source code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.IntWritable;
//import org.apache.hadoop.mapreduce.Reducer.Context;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
//import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
//import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;
```

Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

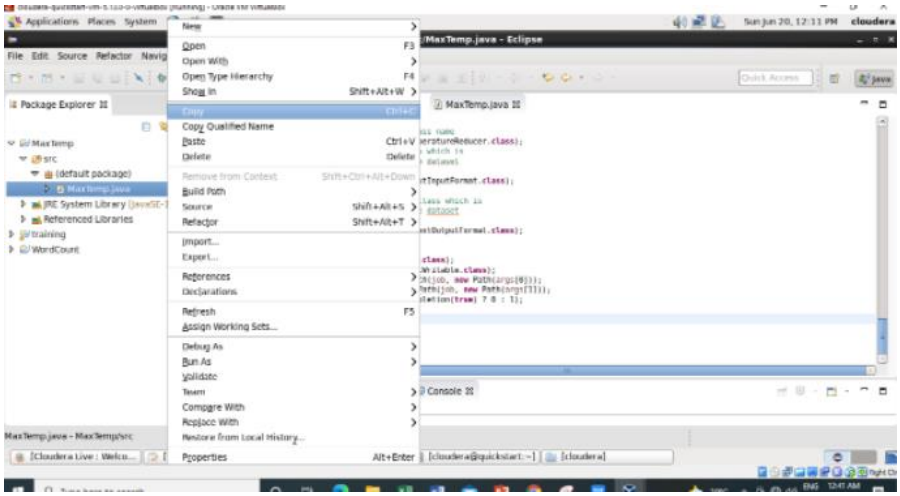
```
//import MaximumTemp.MaxTemperatureMapper;
//import MaxTemp.MaxTemperatureReducer;
public class MaxTemp {
// Mapper
/*MaxTemperatureMapper class is static
* And extends Mapper abstract class
* having four Hadoop generics type
* Long Writable, Text, Text, Text.
*/
public static class MaxTemperatureMapper extends
Mapper<LongWritable, Text, Text, IntWritable> {
// the data in our data set with
// this value is inconsistent data
//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 airtemp;
if(line.charAt(87)=='+')
{
airtemp=Integer.parseInt(line.substring(88,92));
}
else
airtemp=Integer.parseInt(line.substring(87,92));
String q=line.substring(92,93);
if(airtemp!=9999 && q.matches("[01459]"))
{
```

```
context.write(new Text(year), new IntWritable(airtemp));
}
}
}
// Reducer
/*MaxTemperatureReducer class is static
and extends Reducer abstract class
having four Hadoop generics type
Text, Text, Text, Text.
*/
public static class MaxTemperatureReducer extends
Reducer<Text, IntWritable, Text, IntWritable> {
/**
* @method reduce
* This method takes the input as key and
* list of values pair from the mapper,
* it does aggregation based on keys and
* produces the final context.
*/
public void reduce(Text key,Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
int maxvalue= Integer.MIN_VALUE;
for (IntWritable value : values) {
maxvalue=Math.max(maxvalue, value.get());
}
context.write(key, new IntWritable(maxvalue));
}
}

/**
```

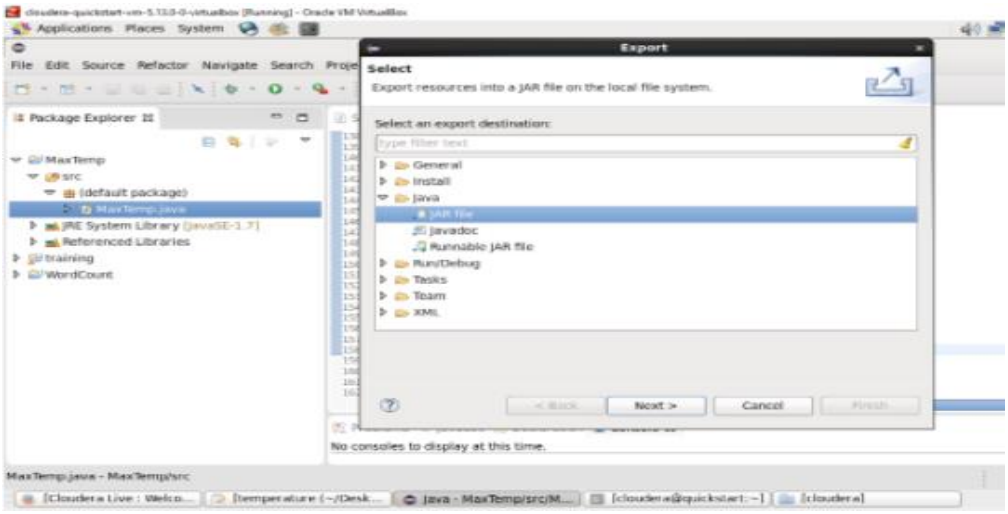
```
* @method main
* This method is used for setting
* all the configuration properties.
* It acts as a driver for map-reduce
* code.
*/
public static void main(String[] args) throws Exception {
// reads the default configuration of the
// cluster from the configuration XML files
Configuration conf = new Configuration();
// initializing the job with the
// default configuration of the cluster
// Job job = new Job(conf, "weather example");
Job job = Job.getInstance(conf, "weather example");
// Assigning the driver class name
job.setJarByClass(MaxTemp.class);
// Key type coming out of mapper
// job.setMapOutputKeyClass(Text.class);
// value type coming out of mapper
// job.setMapOutputValueClass(Text.class);
// Defining the mapper class name
job.setMapperClass(MaxTemperatureMapper.class);
// Defining the reducer class name
job.setReducerClass(MaxTemperatureReducer.class);
// defining input Format class which is
// responsible to parse the dataset
// into a key value pair
job.setInputFormatClass(TextInputFormat.class);
// Defining output Format class which is
// responsible to parse the dataset
```

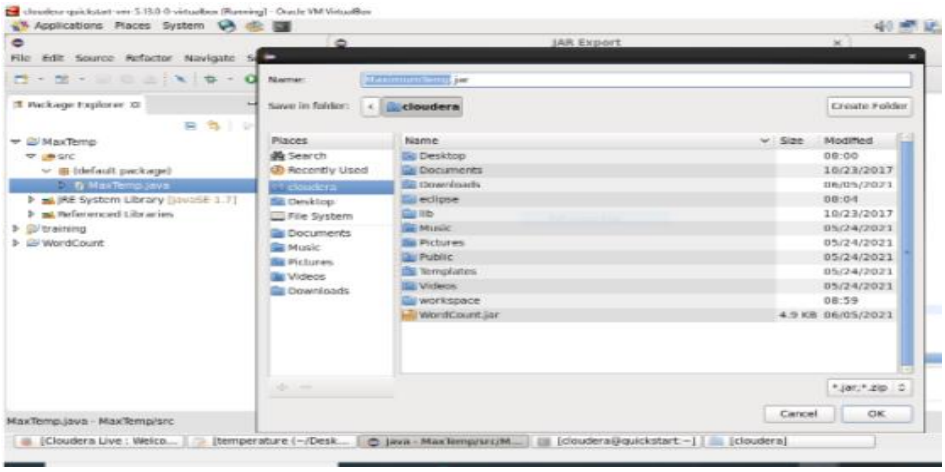
```
// into a key value pair
job.setOutputFormatClass(TextOutputFormat.class);


job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

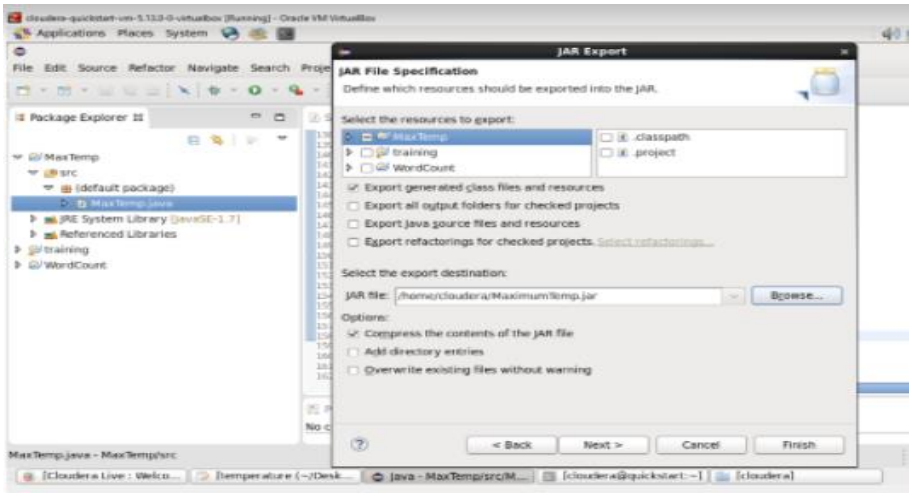
6) Right Click on the project **name MaxTemp** -> **Export** -> **Java** -> **JAR File** -> **Next** -> for select the **export destination** for **JAR file: browse** -> Name : **MaxTemp.jar** -> save in folder - > **cloudera** -> **Finish** -> **OK**



Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

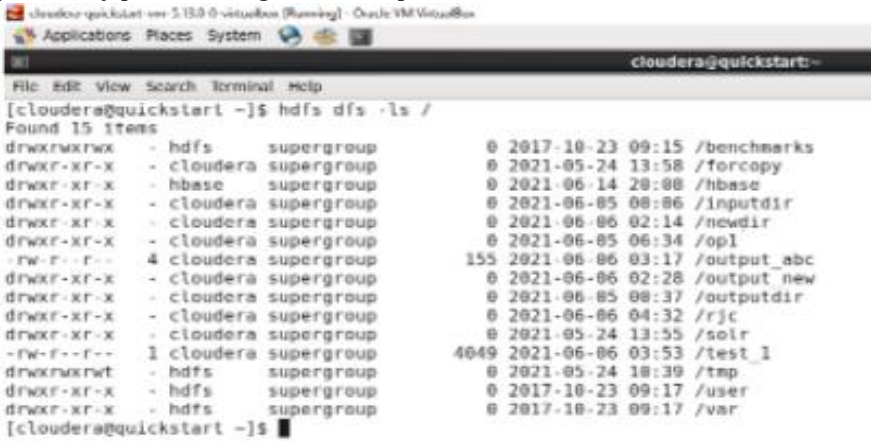




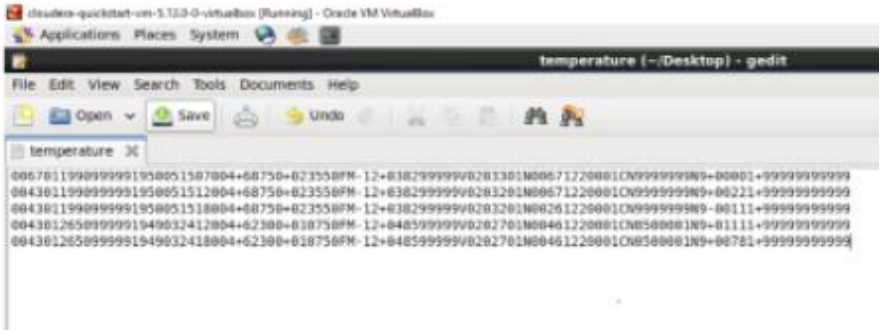


Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

7) Open terminal & type **->hdfs dfs -ls /**
HDFS Command to display the list of Files and Directories in HDFS. It lists the contents of the directory specified by path, showing the names, permissions, owner, size and modification date for each entry**.**



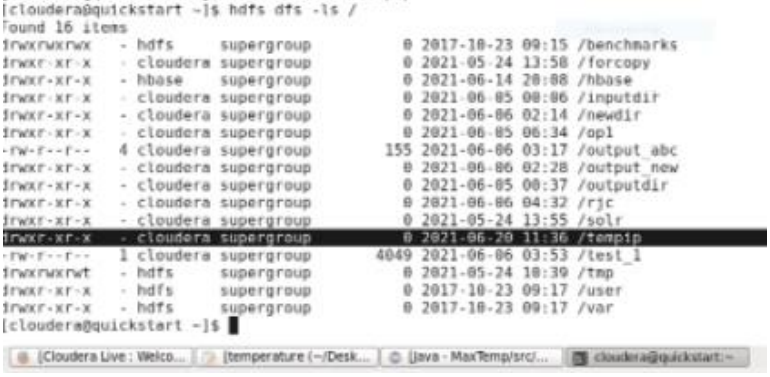
8) Input file named as temperature which is present on desktop i.e. in local file system.



9) Now we have to move this input file to hdfs. For this we create a direcory on hdfs using command ->**hdfs dfs -mkdir /tempip.**
So here I create tempip directory.

```
[cloudera@quickstart ~]$ hdfs dfs -mkdir /tempip
[cloudera@quickstart ~]$ hdfs dfs -ls /tempip
```

Then we can verify whether this directory is created or not using ls command **hdfs dfs -ls /**



Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

10) Move the input file i.e. temperature to this directory created in hdfs by using either put command or copyFromLocal command.
hdfs dfs –put /home/cloudera**/Desktop/temperature /tempip/**

```
[cloudera@quickstart ~]$ hdfs dfs -put /home/cloudera/Desktop/temperature /tempip/
[cloudera@quickstart ~]$
```

11) Now checking whether the "temperature" present in /tempip directory of hdfs or not using **hdfs dfs –ls/tempip** command

```
[cloudera@quickstart ~]$ hdfs dfs -ls /tempip
Found 1 items
-rw-r--r--   1 cloudera supergroup        530 2021-06-20 11:50 /tempip/temperature
[cloudera@quickstart ~]$
```

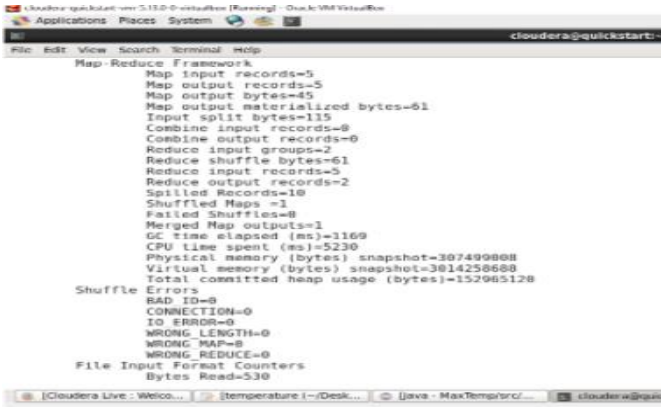
12) As we can see "temperature" file is present in /tempip directory of hdfs. Now we will see the content of this file using hdfs **dfs –cat /tempip/temperature** command.

```
[cloudera@quickstart ~]$ hdfs dfs -cat /tempip/temperature
0067011990999991950051507004+68750+023550FM-12+038299999V0203301N00671220001CN9999999N9+00001+99999999999
0043011990999991950051512004+68750+023550FM-12+038299999V0203201N00671220001CN9999999N9+00221+99999999999
0043011990999991950051518004+68750+023550FM-12+038299999V0203201N00261220001CN9999999N9-00111+99999999999
0043012650999991949032412004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+01111+99999999999
0043012650999991949032418004+62300+010750FM-12+048599999V0202701N00461220001CN0500001N9+00781+99999999999
[cloudera@quickstart ~]$
```

13) Running Mapreduce Program on Hadoop, syntax is hadoop jar jarFileName.jar ClassName /InputFileAddress /outputdir
i.e. **hadoop jar /home/cloudera/MaximumTemp.jar MaxTemp /tempip/temperature /tempop1**

Map-Reduce Framework

Name: sahil shaikh Mushtaq Ahmed
Roll No: 47



14) Verify the content of tempop1 directory and in that part-r file has the actual output by using the command Hdfs dfs -cat /tempop1/part-r-00000. This will give us final output. The same file can also be accessed using a browser. For every execution of this program we need to delete the output directory or give a new name to the output directory every time**.**

- checking whether the tempop1 directory is created in hdfs or not using command

-> hdfs dfs -ls /

```
[cloudera@quickstart ~]$ hdfs dfs -ls /
Found 17 items
drwxrwxrwx   - hdfs     supergroup          0 2017-10-23 09:15 /benchmarks
drwxr-xr-x   - cloudera supergroup          0 2021-05-24 13:58 /forcopy
drwxr-xr-x   - hbase    supergroup          0 2021-06-14 20:08 /hbase
drwxr-xr-x   - cloudera supergroup          0 2021-06-05 00:06 /inputdir
drwxr-xr-x   - cloudera supergroup          0 2021-06-06 02:14 /newdir
drwxr-xr-x   - cloudera supergroup          0 2021-06-05 06:34 /op1
-rw-r--r--   4 cloudera supergroup        155 2021-06-06 03:17 /output_abc
drwxr-xr-x   - cloudera supergroup          0 2021-06-06 02:28 /output_new
drwxr-xr-x   - cloudera supergroup          0 2021-06-05 00:37 /outputdir
drwxr-xr-x   - cloudera supergroup          0 2021-06-06 04:32 /rjc
drwxr-xr-x   - cloudera supergroup          0 2021-05-24 13:55 /solr
drwxr-xr-x   - cloudera supergroup          0 2021-06-20 11:50 /tempip
drwxr-xr-x   - cloudera supergroup          0 2021-06-20 12:37 /tempop1
-rw-r--r--   1 cloudera supergroup       4049 2021-06-06 03:53 /test_1
drwxrwxrwt   - hdfs     supergroup          0 2021-05-24 10:39 /tmp
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /user
drwxr-xr-x   - hdfs     supergroup          0 2017-10-23 09:17 /var
[cloudera@quickstart ~]$
```

Check what we have inside this **tempop1** directory using command as

->**hdfs dfs –ls/tempop1**

```
[cloudera@quickstart ~]$ hdfs dfs -ls /tempop1
Found 2 items
-rw-r--r--   1 cloudera supergroup          0 2021-06-20 12:37 /tempop1/_SUCCESS
-rw-r--r--   1 cloudera supergroup         17 2021-06-20 12:37 /tempop1/part-r-00000
[cloudera@quickstart ~]$
```

Now we want to read the content of the **part-r-00000** file which present inside the **tempop1** using command

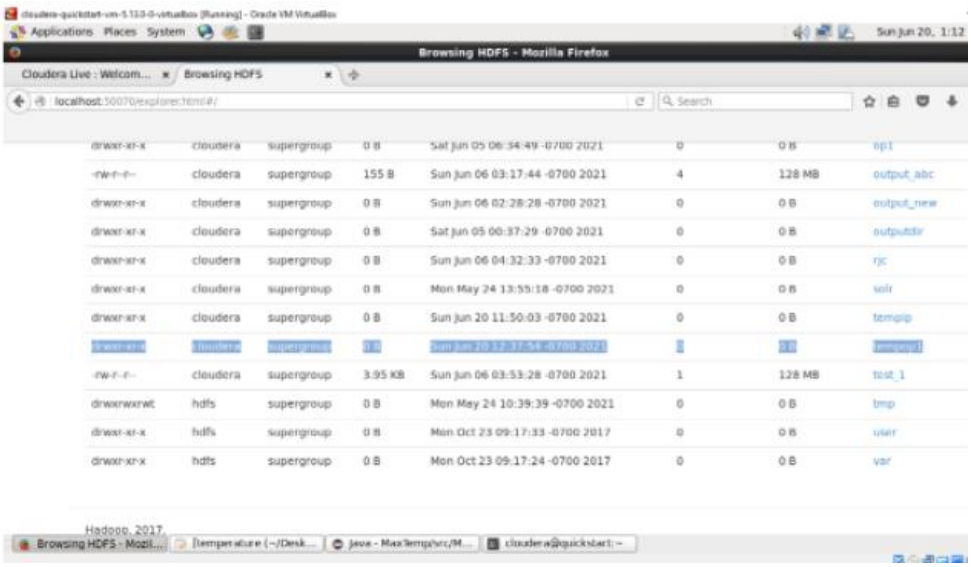
- **hdfs dfs -cat /tempop1/part-r-00000**

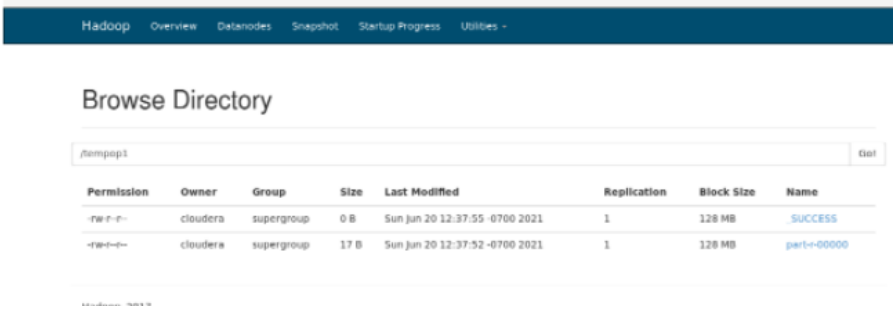
```
[cloudera@quickstart ~]$ hdfs dfs -cat /tempop1/part-r-00000
1949    111
1950    22
[cloudera@quickstart ~]$
```

So the maximum temperature for the year 1949 is 111 and for the year 1950 is 22.

Name: sahil shaikh Mushtaq Ahmed
Roll No: 47

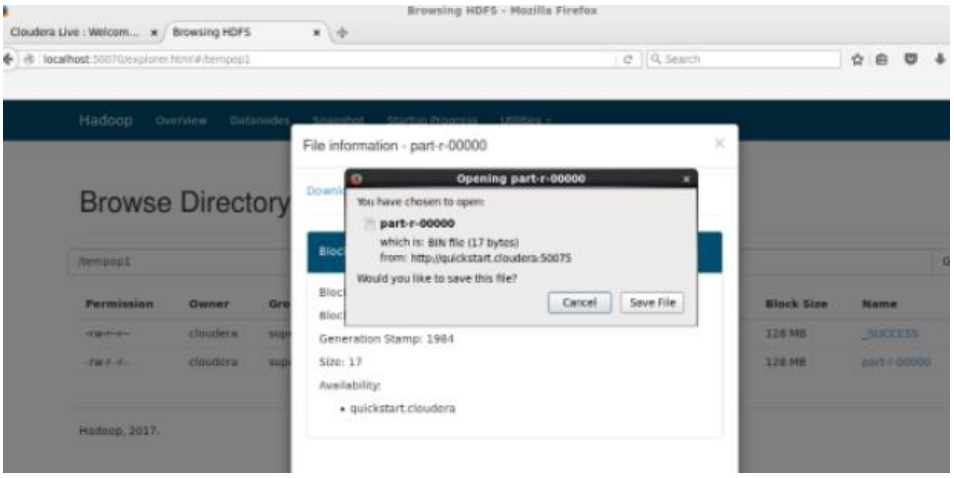
15) The same file can also be accessed using a browser.
Browse the Directory by
Hadoop->**HDFS Namenode->Ultilities ->Browse the file system**





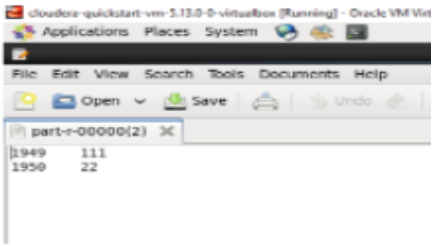
Now downloading the **part-r-00000** file.

Name: sahil shaikh Mushtaq Ahmed
Roll No: 47



Inside the **part-r-00000** file it will have the same output as we are getting after executing using command.

-hadoop jar /home/cloudera/MaximumTemp.jar MaxTemp /tempip/temperature/tempop1



NOTE: - For every execution of this program we need to delete the output directory or give a new name to the output directory every time.