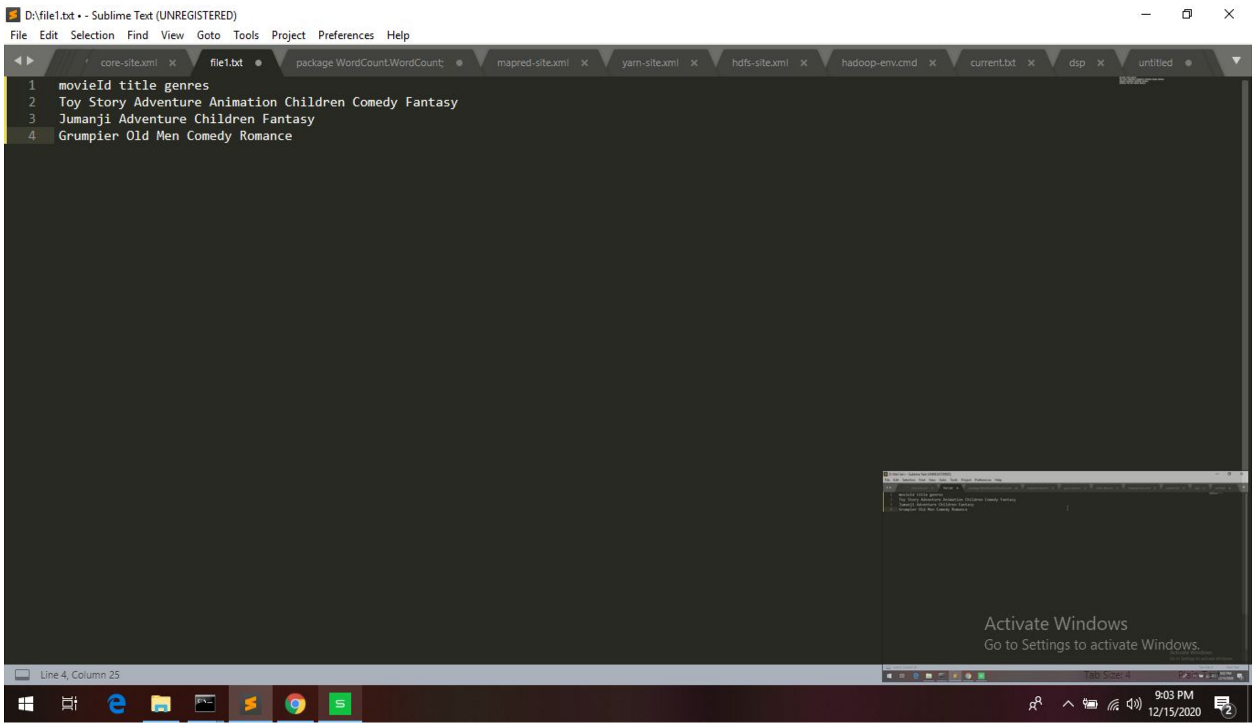
STEP1:CREATE DATA FILE

FIRST create the data file named as file1.txt



D:\file1.txt • - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
core-site.xml
file1.txt
package WordCount.WordCount;
mapred-site.xml
yarn-site.xml
hdfs-site.xml
hadoop-env.cmd
current.txt
dsp
untitled
Activate Windows
Go to Settings to activate Windows.
Line 4, Column 25
9:03 PM
12/15/2020

```
movieId title genres
Toy Story Adventure Animation Children Comedy Fantasy
Jumanji Adventure Children Fantasy
Grumpier Old Men Comedy Romance
```

STEP2: Start HDFS (Namenode and Datanode) and YARN (Resource Manager and Node Manager)

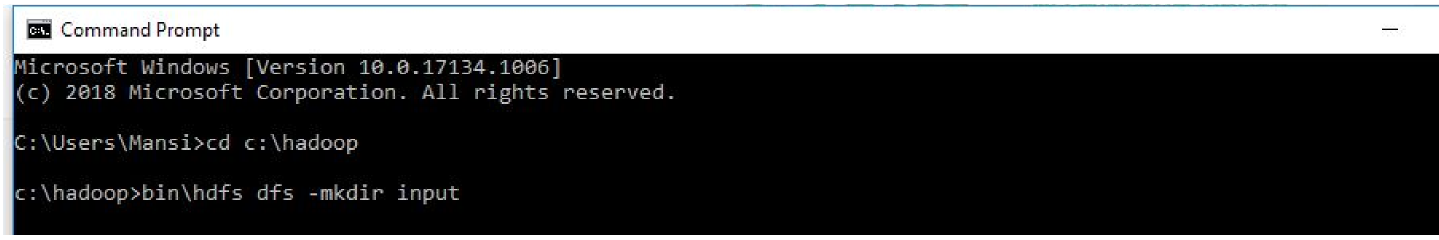
Run following commands

```
C:\Users\Mansi>cd c:\hadoop
c:\hadoop>sbin\start-dfs
c:\hadoop>sbin\start-yarn
starting yarn daemons..
```

Namenode, **Datanode**, **Resource Manager** and **Node Manager** will be started in few minutes and

3) STEPS 3

- Create a directory (say 'input') in HDFS to keep all the text files (say 'file1.txt') to be used for counting words.

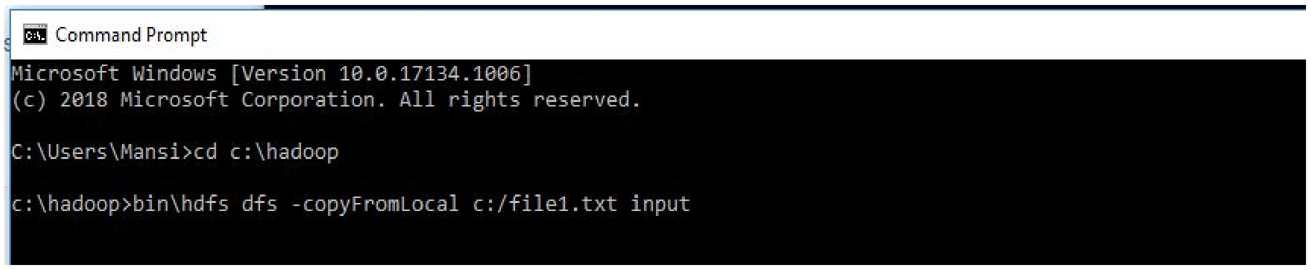


```
Microsoft Windows [Version 10.0.17134.1006]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Mansi>cd c:\hadoop

c:\hadoop>bin\hdfs dfs -mkdir input
```

- Copy the text file(say 'file1.txt') from local disk to the newly created 'input' directory in HDFS.



```
Microsoft Windows [Version 10.0.17134.1006]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Mansi>cd c:\hadoop

c:\hadoop>bin\hdfs dfs -copyFromLocal c:/file1.txt input
```

- Run the wordcount job provided in **%HADOOP_HOME%\share\hadoop\mapreduce\hadoop-mapreduce-examples-2.2.0.jar**
  Command :
  C:\hadoop>bin\yarn jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.2.0.jar wordcount input output

Output of command line after executing above command:

```
13:22:02 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
13:22:03 INFO input.FileInputFormat: Total input paths to process : 1
13:22:03 INFO mapreduce.JobSubmitter: number of splits:1
:
:
13:22:04 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1391412385921_0002
13:22:04 INFO impl.YarnClientImpl: Submitted application application_1391412385921_0002 to
ResourceManager at /0.0.0.0:8032
13:22:04 INFO mapreduce.Job: Running job: job_1391412385921_0002
13:22:14 INFO mapreduce.Job: Job job_1391412385921_0002 running in uber mode : false
```

```
13:22:14 INFO mapreduce.Job:  map 0% reduce 0%
13:22:22 INFO mapreduce.Job:  map 100% reduce 0%
13:22:30 INFO mapreduce.Job:  map 100% reduce 100%
13:22:30 INFO mapreduce.Job: Job job_1391412385921_0002 completed successfully
13:22:31 INFO mapreduce.Job: Counters: 43
        File System Counters
                FILE: Number of bytes read=89
                FILE: Number of bytes written=160142
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=171
                HDFS: Number of bytes written=59
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=5657
                Total time spent by all reduces in occupied slots (ms)=6128
        Map-Reduce Framework
                Map input records=2
                Map output records=7
                Map output bytes=82
                Map output materialized bytes=89
                Input split bytes=116
                Combine input records=7
                Combine output records=6
                Reduce input groups=6
                Reduce shuffle bytes=89
                Reduce input records=6
                Reduce output records=6
                Spilled Records=12
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=145
                CPU time spent (ms)=1418
                Physical memory (bytes) snapshot=368246784
                Virtual memory (bytes) snapshot=513716224
                Total committed heap usage (bytes)=307757056
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=55
        File Output Format Counters
                Bytes Written=59
```

MAPREDUCE CODES:

MAPPER:

```
package WordCount.WordCount;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class WC_Mapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>{
  private final static IntWritable one = new IntWritable(1);
  private Text word = new Text();
  public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
       Reporter reporter) throws IOException{
    String line = value.toString();
    StringTokenizer  tokenizer = new StringTokenizer(line);
    while (tokenizer.hasMoreTokens()){
      word.set(tokenizer.nextToken());
      output.collect(word, one);
    }
    }

}
```

REDUCER:

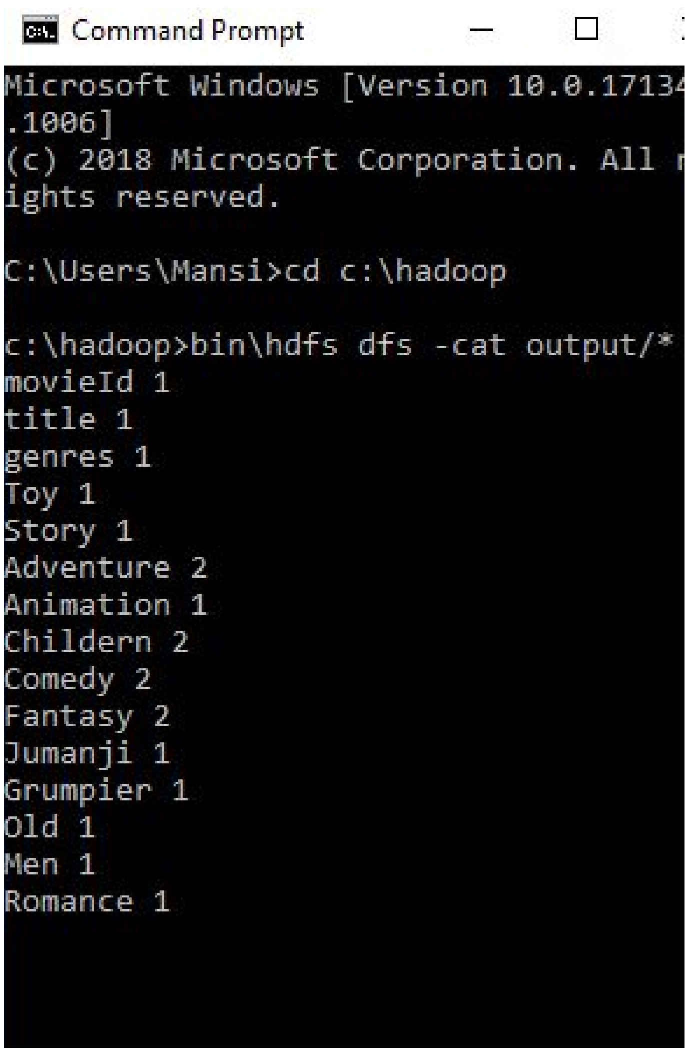
```
    import java.io.IOException;
    import java.util.Iterator;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.MapReduceBase;
    import org.apache.hadoop.mapred.OutputCollector;
    import org.apache.hadoop.mapred.Reducer;
    import org.apache.hadoop.mapred.Reporter;

    public class WC_Reducer  extends MapReduceBase implements
Reducer<Text,IntWritable,Text,IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable>
output,
     Reporter reporter) throws IOException {
    int sum=0;
    while (values.hasNext()) {
    sum+=values.next().get();
    }
    output.collect(key,new IntWritable(sum));
    }
    }
```

Wordcount runner program

```
   import java.io.IOException;
    import org.apache.hadoop.fs.Path;
    import org.apache.hadoop.io.IntWritable;
    import org.apache.hadoop.io.Text;
    import org.apache.hadoop.mapred.FileInputFormat;
    import org.apache.hadoop.mapred.FileOutputFormat;
    import org.apache.hadoop.mapred.JobClient;
    import org.apache.hadoop.mapred.JobConf;
    import org.apache.hadoop.mapred.TextInputFormat;
    import org.apache.hadoop.mapred.TextOutputFormat;
    public class WC_Runner {
        public static void main(String[] args) throws IOException{
            JobConf conf = new JobConf(WC_Runner.class);
        conf.setJobName("WordCount");
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(IntWritable.class);
        conf.setMapperClass(WC_Mapper.class);
        conf.setCombinerClass(WC_Reducer.class);
        conf.setReducerClass(WC_Reducer.class);
        conf.setInputFormat(TextInputFormat.class);
        conf.setOutputFormat(TextOutputFormat.class);
        FileInputFormat.setInputPaths(conf,new Path(args[0]));
        FileOutputFormat.setOutputPath(conf,new Path(args[1]));
        JobClient.runJob(conf);
    }
}
```

CHECK OUTPUT :



```
Microsoft Windows [Version 10.0.17134
.1006]
(c) 2018 Microsoft Corporation. All r
ights reserved.

C:\Users\Mansi>cd c:\hadoop

c:\hadoop>bin\hdfs dfs -cat output/*
movieId 1
title 1
genres 1
Toy 1
Story 1
Adventure 2
Animation 1
Childern 2
Comedy 2
Fantasy 2
Jumanji 1
Grumpier 1
Old 1
Men 1
Romance 1
```

Conclusion: I have extracted the Count of each word in the file given named file1.txt