Date __/__/__
Page

Lab 6.

Word count :-

WCDriver.

```
package hadoopwordcount;

public class WCDriver.
    public int run (String[] args ) throws
      IOException{
      if (arg.length < 2){
          System.out.println("Please provide
          valid input and output paths");
          return -1;
      }
      JobConf conf = new JobConf(WCDriver.class);
      conf.setJobName ("Word count");

      FileInputFormat.setInputPaths(conf, new Path(args[0]));
      FileOutputFormat.setOutputPath(conf, new
        Path(args[1]));

      conf.setMapperClass(WCMapper.class);
      conf.setReducerClass(WCReducer.class);

      conf.setMapOutputKeyClass(WCMapper.Text.class);

      return 0;
  }
  public static void main (String[] args)
  throws Exception{
      int exitCode = ToolRunner.run(new
      WCDriver(), args);
      System.out.println("Exit code: "+exitCode);
  }
}
```

Date __/__/__
Page _____

WCMapper.

```
public class WCMapper.
    public void map(LongWritable key, Text
    value, OutputCollector<Text, IntWritable>
    output, Reporter reporter) throws.
    IOException{.

    String line = value.toString();

    for(String word: line.split(" "))}
        if(word.length()>0){
            output.collect(new Text(word),
            new IntWritable(1));
    }
}}.
```

WCReducer.java

```
public class WCReducer.{
    public void reduce{
        int count=0;
        while(values.hasNext()){
            count+=values.next().get();
    }
        output.collect(key, new IntWritable(count));
}}.
```

Output

are 1
Brother 1
family 1
hi 1

Date
Page

there 5
is 4
solo 1
sister 1
you 1
your 4.

Average temperature:

Average Mapper

```
public class AverageMapper.

    public static final int MISSING=9999;

  public void map(LongWritable key).
     String line. = value. toString();
     if (line. length()<93){
           return;
     }
     String year= line. substring(15,19);

     int temperature;
     if (line.charAt(87)==-'+'){
       temperature = Integer. parseInt(
        line.substring(87,92));
     } else {
       temperature=Integer. parseInt(
      line. substring (87, 92));
     }
     String quality= line. substring
            (92,93);
```

Date __/__/__
Page _____

```
if (temperature != MISSING && quality
        matches("[01459]")) {
    context.write(new Text(year), new
    IntWritable(temperature));
}}.
```

Average Reducer.

```
public class AverageReducer {.
    protected void reduce {
        int sum=0;
        int count=0;

        for (IntWritable value: values) {.
            sum+= value.get();
            count++;
        }

        int average=(count==0)?0:sum/count;

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

    }

}.
```

Output

01	-13
02	-66
03	-15
04	13
05	100
06	168
07	219
08	198

09	144
10	100
11	1
12	-61

Mean Max Temperature

MeanMaxMapper.

```
public class MeanMaxMapper.

    public void map(Text value){
        string line = value.toString();

        if (line.length()<93){
            return;
        }
        string month = line.substring(19,81);

        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);
    }
}
```

Date __/__/__
Page _____

MeanMaxReducer.

```
public class MeanMaxReducer.

    public void reduce {.
      int MaxTemp = Integer.MIN_VALUE;
      int totalTemp = 0;
      int count = 0;
      int days = 0;

      for (IntWritable value : values) {
          int temp = value.get();

          if (temp > maxTemp) {
              maxTemp = temp;
          }
          count ++;

          if (count == 3) {
              totalTemp += maxTemp;
              maxTemp = Integer.MIN_VALUE;
              count = 0;
              days ++
          }
      }
    }
```

Output
190 1 24 6.

Date
Page

Top 10 maximum occurrences of words.

Mapper

```
import sys
import re

for line in sys.stdin
    words = re.findall(r'\w+', line.lower())
    for words in words:
        print(f"{word}\t1")
```

Reducer

```
from collections import defaultdict

count_map = defaultdict(int)

for line in sys.stdin:
    word, count = line.strip().split('\t')
    count_map[word] += int(count)

sorted_words = sorted(count_map.items())

for word, count in sorted_words[:10]:
    print(f"{word}\t{count}")
```

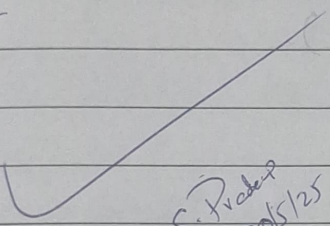
```
hadoop jar /home/hadoop/hadoop/share/
hadoop/tools/lib/hadoop-streaming-
3.3.6.jar
-input /word/input.txt
-output /word/output
```

Date __/__/__
Page _____

- mapper mapperword.py
- reducer reducerword.py

Output

12
11
7
6.
6
5
3
3
3
2.



S. Prabu
20/5/25