

## AIM

### Description

Sensors sense weather data in big text format containing station ID, year, date, time, temperature, quality etc. from each sensor and store it in a single line. Suppose thousands of data sensors are there, then we have thousands of records with no particular order. We require only a year and maximum temperature of particular quality in that year.

For example:

Input string from sensor:

0029029070999991902010720004+64333+023450

FM-12+

000599999V0202501N027819999999N0000001N9-00331+

99999098351ADDGF1029919999999999999999999

Here: 1902 is year

0033 is temperature

1 is measurement quality (Range between 0 or 1 or 4 or 5 or 9)

Here each mapper takes the input **key** as "byte offset of line" and **value** as "one weather sensor read i.e one line". and parse each line and produce an intermediate **key** "year" and **intermediate value** as "temperature of certain measurement qualities" for that year.

The combiner will form set values of temperature. Year and set of values of temperatures is given as input <key, value> to reducer and Reducer will produce year and maximum temperature for that year from the set of temperature values.

## PROGRAM

\*/

```
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;

//Mapper class

class MaxTemperatureMapper
extends Mapper<LongWritable, Text, Text, IntWritable> { private static final int MISSING

= 9999;

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

String line = value.toString(); String year = line.substring(15, 19); int airTemperature;
if (line.charAt(87) == '+') { // parseInt doesn't like leading plus signs airTemperature =
Integer.parseInt(line.substring(88, 92));
} else {
airTemperature = Integer.parseInt(line.substring(87, 92));
}
String quality = line.substring(92, 93);
if (airTemperature != MISSING && quality.matches("[01459]")) { context.write(new
Text(year), new IntWritable(airTemperature));
}
}
}

//Reducer class
class MaxTemperatureReducer
extends Reducer<Text, IntWritable, Text, IntWritable> {
```

@Override

NAME: JOSHITA UMANATH

```
public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
int maxVal = Integer.MIN_VALUE; for (IntWritable value : values)
{ maxVal = Math.max(maxVal, value.get());
}
context.write(key, new IntWritable(maxVal));
}
}
//Driver Class
```

```
public class MaxTemperature {
```

```
public static void main(String[] args) throws Exception { if (args.length != 2)
{ System.err.println("Usage: MaxTemperature <input path=> <output path>"); System.exit(-
1);
}
```

```
Job job = Job.getInstance(new Configuration()); job.setJarByClass(MaxTemperature.class);
job.setJobName("Max temperature");
```

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

```
job.setMapperClass(MaxTemperatureMapper.class);
job.setReducerClass(MaxTemperatureReducer.class);
```

```
job.setOutputKeyClass(Text.class); job.setOutputValueClass(IntWritable.class);
```

```
job.submit();
}
}
```

## OUTPUT:

Input for String :

0029029070999991902010720004+64333+023450FM-12+  
000599999V0202501N0278199999999N0000001N9-00331+  
99999098351ADDGF1029919999999999999999

```
File Actions Edit View Help
(hadoop@kali)-[~]
$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as hadoop in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
(hadoop@kali)-[~]
$ jps
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
14436 NodeManager
16772 Jps
13830 SecondaryNameNode
14311 ResourceManager
13597 DataNode
13471 NameNode
```

```
(hadoop@kali)-[~/hadoop/bin]
$ ./hdfs dfs -ls /exp3
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
2024-09-21 00:11:13,818 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...
Found 3 items
-rw-r--r-- 1 hadoop supergroup 79205 2024-08-29 10:50 /exp3/dataset.txt
drwxr-xr-x - hadoop supergroup 0 2024-08-29 10:52 /exp3/new_output
drwxr-xr-x - hadoop supergroup 0 2024-09-13 01:00 /exp3/output
```

```
(hadoop@kali)-[~/hadoop/bin]
$ hadoop jar $HADOOP_STREAMING -input /exp3/dataset.txt -output /exp3/output -mapper ~/DA-Lab/exp3/mapper.py -reducer ~/DA-Lab/exp3/reducer.py
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
2024-09-21 00:13:19,993 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
packageJobJar: [/tmp/hadoop-unjar3835/004787382099/] [] /tmp/streamjob2158010624078613243.jar tmpDir=null
2024-09-21 00:13:20,918 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-21 00:13:21,223 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-09-21 00:13:27,216 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hadoop/.staging/job_1726891437845_0001
2024-09-21 00:13:28,262 INFO mapred.FileInputFormat: Total input files to process : 1
2024-09-21 00:13:28,365 INFO mapreduce.JobSubmitter: number of splits:2
2024-09-21 00:13:28,613 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1726891437845_0001
2024-09-21 00:13:28,613 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-09-21 00:13:29,230 INFO conf.Configuration: resource-types.xml not found
2024-09-21 00:13:29,230 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-09-21 00:13:29,805 INFO impl.VarnClientImpl: Submitted application application_1726891437845_0001
2024-09-21 00:13:29,993 INFO mapreduce.Job: The url to track the job: http://kali:8088/proxy/application_1726891437845_0001/
2024-09-21 00:13:29,998 INFO mapreduce.Job: Running job: job_1726891437845_0001
2024-09-21 00:13:43,554 INFO mapreduce.Job: Job job_1726891437845_0001 running in uber mode : false
2024-09-21 00:13:43,560 INFO mapreduce.Job: map 0% reduce 0%
2024-09-21 00:13:52,918 INFO mapreduce.Job: map 100% reduce 0%
2024-09-21 00:14:00,992 INFO mapreduce.Job: map 100% reduce 100%
2024-09-21 00:14:01,012 INFO mapreduce.Job: Job job_1726891437845_0001 completed successfully
2024-09-21 00:14:01,189 INFO mapreduce.Job: Counters: 54
File System Counters
  FILE: Number of bytes read=102094
  FILE: Number of bytes written=1138411
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=83481
  HDFS: Number of bytes written=96
  HDFS: Number of read operations=11
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=2
  HDFS: Number of bytes read erasure-coded=0
Job Counters
  Launched map tasks=2
  Launched reduce tasks=1
  Data-local map tasks=2
  Total time spent by all maps in occupied slots (ms)=14691
  Total time spent by all reduces in occupied slots (ms)=4696
  Total time spent by all map tasks (ms)=14691
  Total time spent by all reduce tasks (ms)=4696
  Total vcore-milliseonds taken by all map tasks=14691
  Total vcore-milliseonds taken by all reduce tasks=4696
  Total megabyte-milliseonds taken by all map tasks=15043584
  Total megabyte-milliseonds taken by all reduce tasks=4808704
Map-Reduce Framework
  Map input records=365
  Map output records=10220
  Map output bytes=81648
  Map output materialized bytes=102100
  Input split bytes=180
```

NAME: JOSHITA UMANATH

```
(hadoop@kali)-[~/hadoop/bin]
$ ./hdfs dfs -cat /exp3/output/*
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
2024-09-21 00:15:38,966 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform...
01      26.5
02      26.6
03      29.1
04      30.8
05      31.1
06      33.6
07      38.5
08      40.2
09      36.5
10      36.9
11      27.6
12      25.9
```

## RESULT

Thus a java program has been implemented to identify the year-wise maximum temperature from the sensor data.

NAME: JOSHITA UMANATH