```
harithaah@fedora:~/CC/exp2$ javac -classpath $HADOOP_HOME/share/hadoop/common/*:$HADOOP_HOME/share/hadoop/mapreduce/*:. -d . Mapper1.java Reducer1.java Runner1.java
harithaah@fedora:~/CC/exp2$ jar -cvf wordcount.jar -C . .
added manifest
adding: Mapper1.java(in = 1036) (out= 369)(deflated 64%)
adding: Reducer1.java(in = 871) (out= 368)(deflated 57%)
adding: Runner1.java(in = 1433) (out= 487)(deflated 66%)
adding: s.txt(in = 158) (out= 114)(deflated 27%)
adding: Mapper1.class(in = 1858) (out= 759)(deflated 59%)
adding: Reducer1.class(in = 1527) (out= 604)(deflated 60%)
adding: Runner1.class(in = 1432) (out= 709)(deflated 50%)
harithaah@fedora:~/CC/exp2$ hadoop jar wordcount.jar Runner1 /CC/s.txt /CC/output
2024-10-26 14:00:08,752 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-10-26 14:00:09,469 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2024-10-26 14:00:11,224 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application
with ToolRunner to remedy this.
2024-10-26 14:00:11,493 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/harithaah/.staging/job_1729930878688_0001
2024-10-26 14:00:13,797 INFO mapred.FileInputFormat: Total input files to process : 1
2024-10-26 14:00:15,965 INFO mapreduce.JobSubmitter: number of splits:2
2024-10-26 14:00:18,874 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1729930878688_0001
2024-10-26 14:00:18,875 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-10-26 14:00:19,491 INFO conf.Configuration: resource-types.xml not found
2024-10-26 14:00:19,495 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2024-10-26 14:00:21,006 INFO impl.YarnClientImpl: Submitted application application_1729930878688_0001
2024-10-26 14:00:21,385 INFO mapreduce.Job: The url to track the job: http://fedora:8088/proxy/application_1729930878688_0001/
2024-10-26 14:00:21,390 INFO mapreduce.Job: Running job: job_1729930878688_0001
2024-10-26 14:00:59,125 INFO mapreduce.Job: Job job_1729930878688_0001 running in uber mode : false
2024-10-26 14:00:59,130 INFO mapreduce.Job:  map 0% reduce 0%
2024-10-26 14:01:26,297 INFO mapreduce.Job:  map 100% reduce 0%
2024-10-26 14:01:46,586 INFO mapreduce.Job:  map 100% reduce 100%
2024-10-26 14:01:48,821 INFO mapreduce.Job: Job job_1729930878688_0001 completed successfully
2024-10-26 14:01:48,756 INFO mapreduce.Job: Counters: 54
```

```
harithaah@fedora:~/CC/exp2$ hdfs dfs -cat /CC/output/part-00000
B       1
CSE     1
From    1
Hello   1
Hey     1
We      1
a       1
an      1
are     1
awesome 1
be      1
ever    1
found   1
get     1
girl    2
happy   1
have    2
her     2
here    1
i       3
if      1
is      1
like    1
never   1
```

RESULT

Thus a word count program in java is implemented using Map Reduce.

Exp No: 11 **Date:**

HADOOP
IMPLEMENT THE MAX TEMPERATURE MAPREDUCE PROGRAM TO IDENTIFY THE YEAR WISE MAXIMUM TEMPERATURE FROM SENSOR DATA

AIM

To implement the Max temperature MapReduce program to identify the year-wise maximum temperature from the sensor data.

Description

Sensors senses 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:
002902907099999*1902*010720004+64333+023450
FM-12+
000599999V0202501N027819999999N0000001N9-*00331*+
99999098351ADDGF102991999999999999999999

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

*/

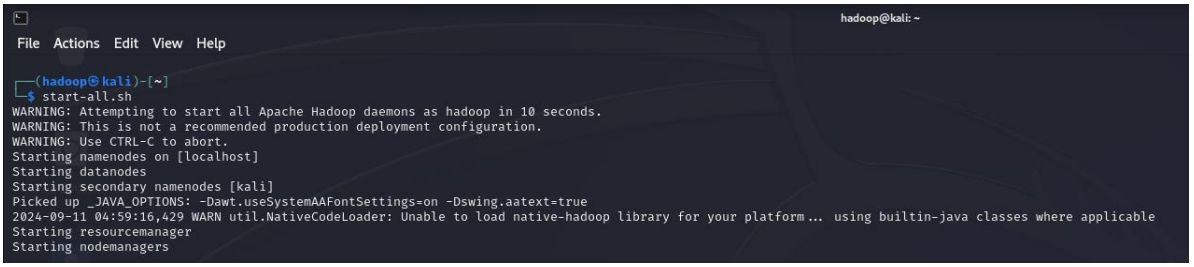
```
import  org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
importorg.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
importorg.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
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));
}
}
//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+
000599999V0202501N027819999999N0000001N9-00331+
99999098351ADDGF102991999999999999999’



```
┌──(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-unjar3830594044787382099/] [] /tmp/streamjob2158010624070613243.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 00:13:29,895 INFO impl.YarnClientImpl: 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-milliseconds taken by all map tasks=14691
                Total vcore-milliseconds taken by all reduce tasks=4696
                Total megabyte-milliseconds taken by all map tasks=15043584
                Total megabyte-milliseconds 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
```

```
┌──(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.

Sample Questions

BASIC UNDERSTANDING: Exp 1

1. What is virtualization?

Ans. Virtualization is an abstraction layer that decouples physical hardware from operating system to deliver greater IT resource utilization and flexibility.

2. What is the Difference between Full Virtualization and Para Virtualization?

Ans. Full virtualization & Para virtualization both comes under the Hardware virtualization. Some of the differences between them are listed below:

Full Virtualization: In full virtualization guest VMs (Virtual Machines) are not aware that they are in virtualized environment there-fore the guest os issues command to what it thinks as actual hardware but actually are just simulated devices created by the hosts.

Para Virtualization : In para virtualization the guest vm is aware that it is in a virtualized environment . If guest vm requires resources , it issues commands to host operating system instead of directly communicating with simulated hardware.

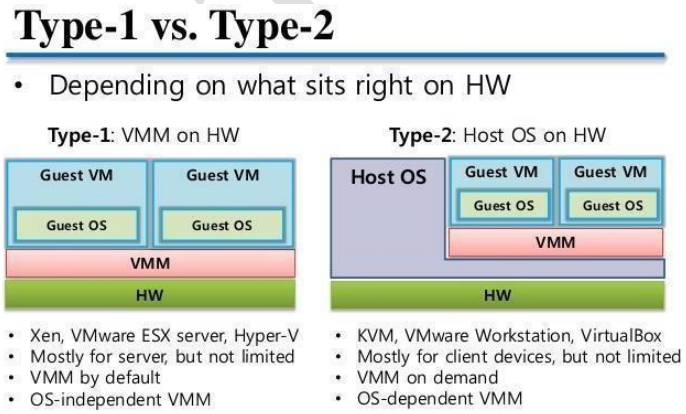
3. What is Hyper-visor ?

A **hypervisor** or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines. A computer on which a **hypervisor** runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

4. Whatare the difference between Type 1 and Type 2 Hypervisor ?

Ans. Type 1: When the Hypervisor is installed on bare metal / Physical hardware it is known as Type 1 Hypervisor . Examples are VM ware ESXi, Oracle VM, Microsoft Hyper V.

Type 2: When the Hypervisor is installed on top of an operating system it is known as Type 2 Hypervisor . Examples are Microsoft Virtual Server, VM Ware Server and workstation.



BASIC UNDERSTANDING: Exp 2

1. **What is a virtual block?**

   A virtual block device is an interface with applications that appears to the applications as a memory device, such as a standard block device.

2. **What is a virtual disk?**

   Virtual disks are stored as files on the host computer or on a network file server. It does not matter whether the physical disk that holds the files is IDE or SCSI.

   IDE (Integrated Drive Electronics) SCSI(Small Computer System Interface) SATA(Serial Advanced Technology Attachment)

3. **What is a VM clone?**

   A clone is a copy of an existing virtual machine.

4. **What is a Snapshot and a Template?**

   A snapshot is a copy of the virtual machine's disk file at a given point in time. **Snapshots** provide a change log for the virtual disk and are used to restore a VM to a particular point in time when a failure or system error occurs.

   A **template** is a master copy of a virtual machine that can be used to create many clones.

s