EX 10: Implement the MaxTemperature MapReduce program to identify the year wise maximum temperature from sensor data

AIM:

To Implement the MaxTemperature MapReduce program to identify the year

wise maximum temperature from sensor data.

PROCEDURE:

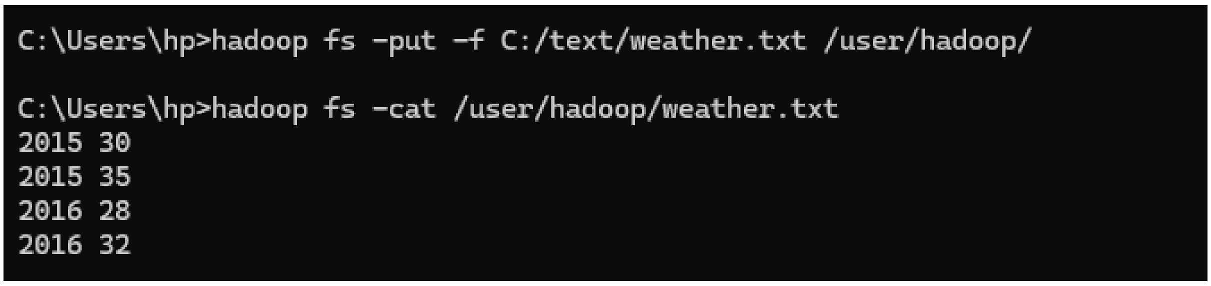
1. **Prepare the Input File**: Ensure the input file (weather.txt) contains the required data and is stored in the specified local directory (e.g., C:/text/).
2. **Upload File to HDFS**: Use the Hadoop command to upload the input file from the local system to HDFS (/user/hadoop/weather.txt).
3. **Verify File in HDFS**: Check that the file has been successfully uploaded to the desired HDFS directory.
4. **Display File Content from HDFS**: View the file’s contents directly from HDFS to confirm successful transfer and data integrity.
5. **Remove File from HDFS (if necessary)**: Optionally, delete the file from HDFS if no longer needed, using the appropriate Hadoop command.

Program:

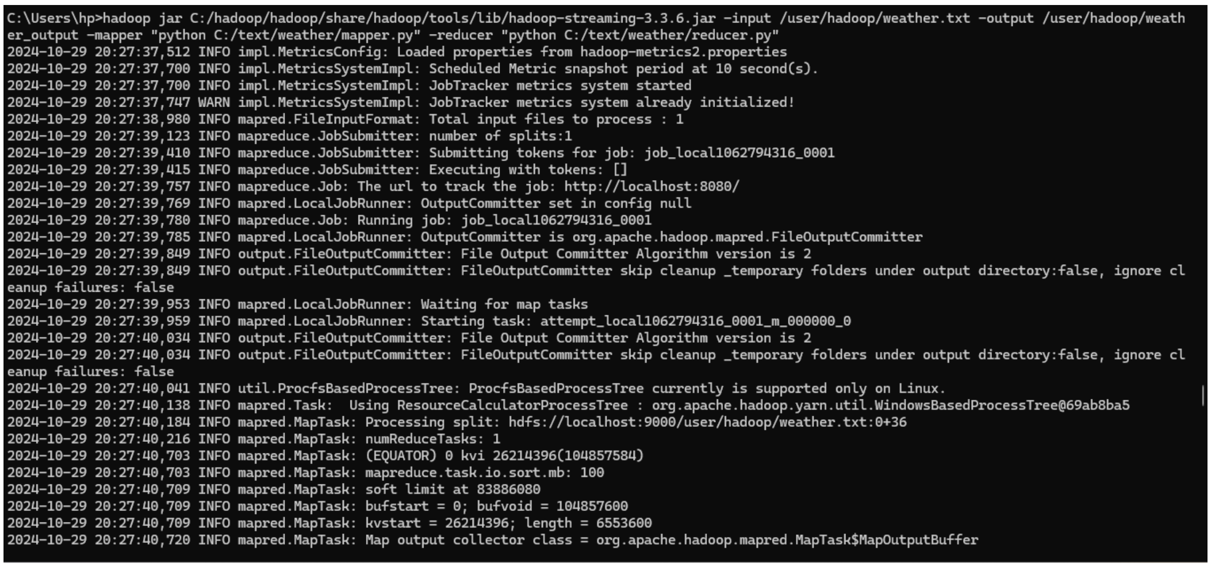
```
#!/usr/bin/env python3

import sys

# Mapper function

for line in sys.stdin:

    # Remove any leading/trailing whitespace

    line = line.strip()

    # Split the line into year and temperature

    year, temperature = line.split()

    # Output the year as the key and temperature as the value

    print(f"{year}\t{temperature}")



#!/usr/bin/env python3

import sys
```

```
current_year = None
max_temp = -float('inf')

# Reducer function
for line in sys.stdin:
    # Remove any leading/trailing whitespace
    line = line.strip()

    # Parse the input from mapper.py
    year, temperature = line.split('\t')
    temperature = int(temperature)

    # Check if we are still on the same year
    if current_year == year:
        # Update max temperature if found higher
        max_temp = max(max_temp, temperature)
    else:
        # Print the result for the previous year
        if current_year is not None:
            print(f"{current_year}\t{max_temp}")

        # Move to the next year and reset max_temp
        current_year = year
        max_temp = temperature
    # Output the max temperature for the last year
    if current_year is not None:
        print(f"{current_year}\t{max_temp}")
```

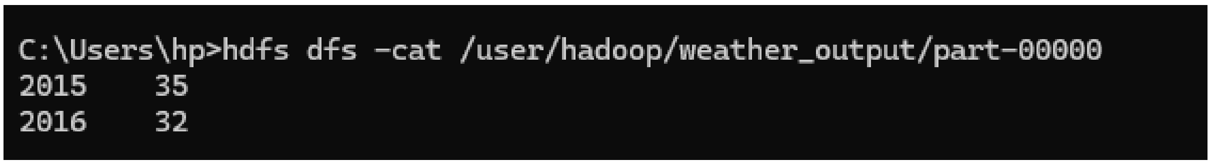
OUTPUT:



```
C:\Users\hp>hadoop fs -put -f C:/text/weather.txt /user/hadoop/

C:\Users\hp>hadoop fs -cat /user/hadoop/weather.txt
2015 30
2015 35
2016 28
2016 32
```



```
C:\Users\hp>hadoop jar C:/hadoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.3.6.jar -input /user/hadoop/weather.txt -output /user/hadoop/weath
er_output -mapper "python C:/text/weather/mapper.py" -reducer "python C:/text/weather/reducer.py"
2024-10-29 20:27:37,512 INFO impl.MetricsConfig: Loaded properties from hadoop-metrics2.properties
2024-10-29 20:27:37,700 INFO impl.MetricsSystemImpl: Scheduled Metric snapshot period at 10 second(s).
2024-10-29 20:27:37,700 INFO impl.MetricsSystemImpl: JobTracker metrics system started
2024-10-29 20:27:37,747 WARN impl.MetricsSystemImpl: JobTracker metrics system already initialized!
2024-10-29 20:27:38,980 INFO mapred.FileInputFormat: Total input files to process : 1
2024-10-29 20:27:39,123 INFO mapreduce.JobSubmitter: number of splits:1
2024-10-29 20:27:39,410 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1062794316_0001
2024-10-29 20:27:39,415 INFO mapreduce.JobSubmitter: Executing with tokens: []
2024-10-29 20:27:39,757 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
2024-10-29 20:27:39,769 INFO mapred.LocalJobRunner: OutputCommitter set in config null
2024-10-29 20:27:39,780 INFO mapreduce.Job: Running job: job_local1062794316_0001
2024-10-29 20:27:39,785 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapred.FileOutputCommitter
2024-10-29 20:27:39,849 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-10-29 20:27:39,849 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cl
eanup failures: false
2024-10-29 20:27:39,953 INFO mapred.LocalJobRunner: Waiting for map tasks
2024-10-29 20:27:39,959 INFO mapred.LocalJobRunner: Starting task: attempt_local1062794316_0001_m_000000_0
2024-10-29 20:27:40,034 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 2
2024-10-29 20:27:40,034 INFO output.FileOutputCommitter: FileOutputCommitter skip cleanup _temporary folders under output directory:false, ignore cl
eanup failures: false
2024-10-29 20:27:40,041 INFO util.ProcfsBasedProcessTree: ProcfsBasedProcessTree currently is supported only on Linux.
2024-10-29 20:27:40,138 INFO mapred.Task:  Using ResourceCalculatorProcessTree : org.apache.hadoop.yarn.util.WindowsBasedProcessTree@69ab8ba5
2024-10-29 20:27:40,184 INFO mapred.MapTask: Processing split: hdfs://localhost:9000/user/hadoop/weather.txt:0+36
2024-10-29 20:27:40,216 INFO mapred.MapTask: numReduceTasks: 1
2024-10-29 20:27:40,703 INFO mapred.MapTask: (EQUATOR) 0 kvi 26214396(104857584)
2024-10-29 20:27:40,703 INFO mapred.MapTask: mapreduce.task.io.sort.mb: 100
2024-10-29 20:27:40,709 INFO mapred.MapTask: soft limit at 83886080
2024-10-29 20:27:40,709 INFO mapred.MapTask: bufstart = 0; bufvoid = 104857600
2024-10-29 20:27:40,709 INFO mapred.MapTask: kvstart = 26214396; length = 6553600
2024-10-29 20:27:40,720 INFO mapred.MapTask: Map output collector class = org.apache.hadoop.mapred.MapTask$MapOutputBuffer
```



```
C:\Users\hp>hdfs dfs -cat /user/hadoop/weather_output/part-00000
2015    35
2016    32
```

Result:

Thus to Implement the MaxTemperature MapReduce program to identify the year wise maximum temperature from sensor data was completed successfully.