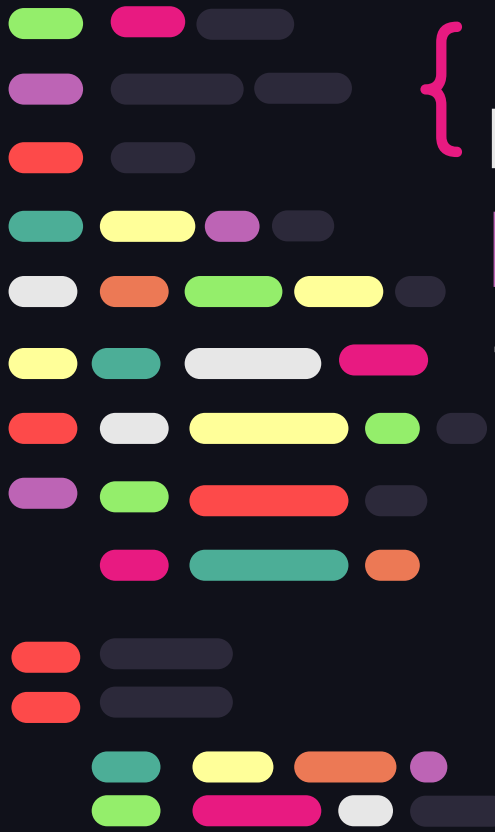
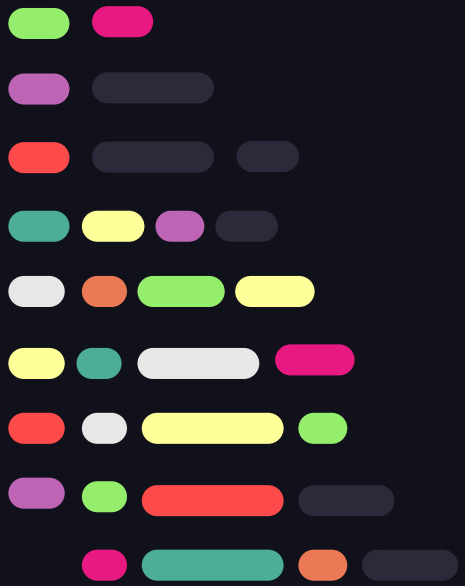


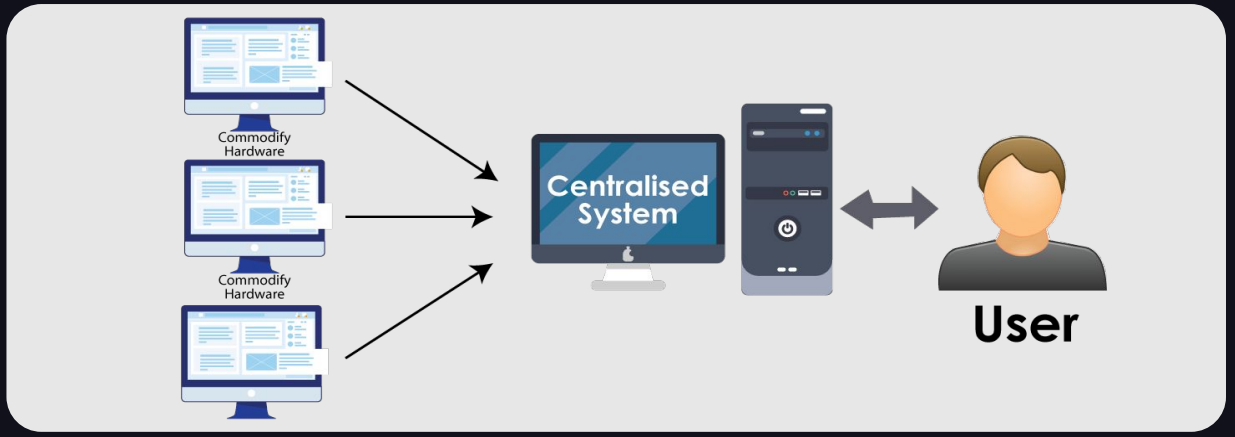
MapReduce program to find Maximum Temperature from a given dataset.

< Team Members –
Parth Madaan, BT20HCS059
Palak Sahu, BT20HCS219 >

Table of contents

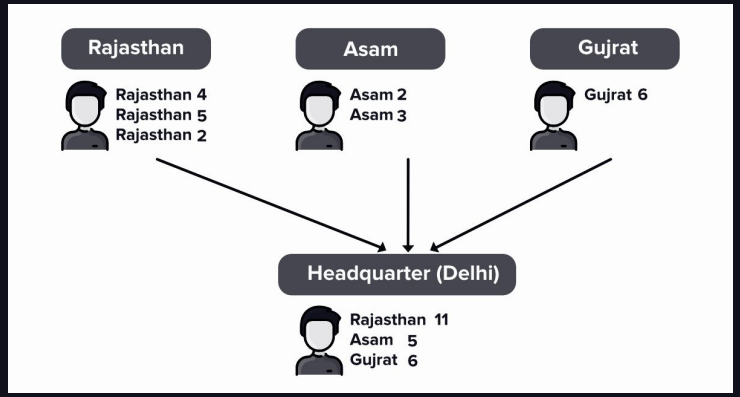


Introduction to MapReduce

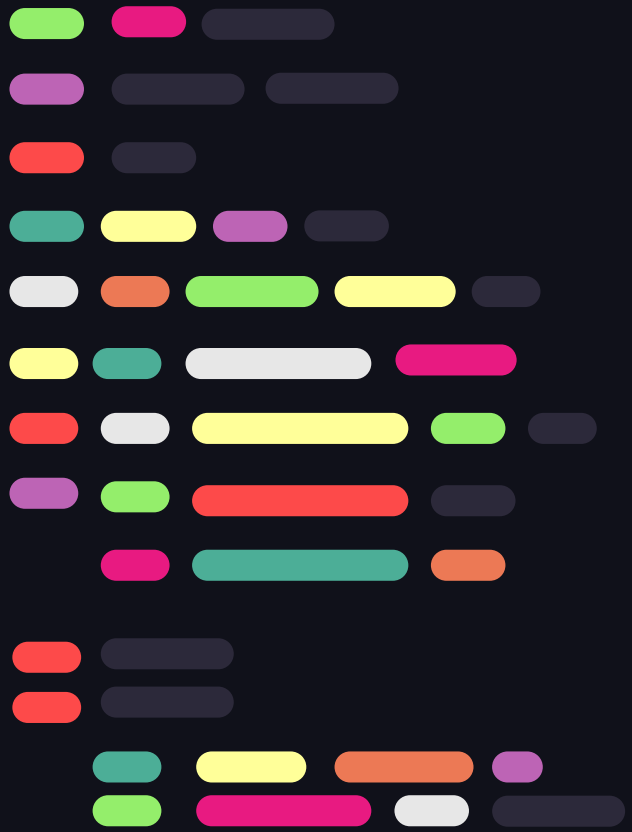


Map reduce is an application programming model to process data in multiple parallel nodes. It divides a task into smaller parts and assigns them to many devices.

Features of MapReduce



- Parallel Computing
- Fault Tolerance
- Scalability

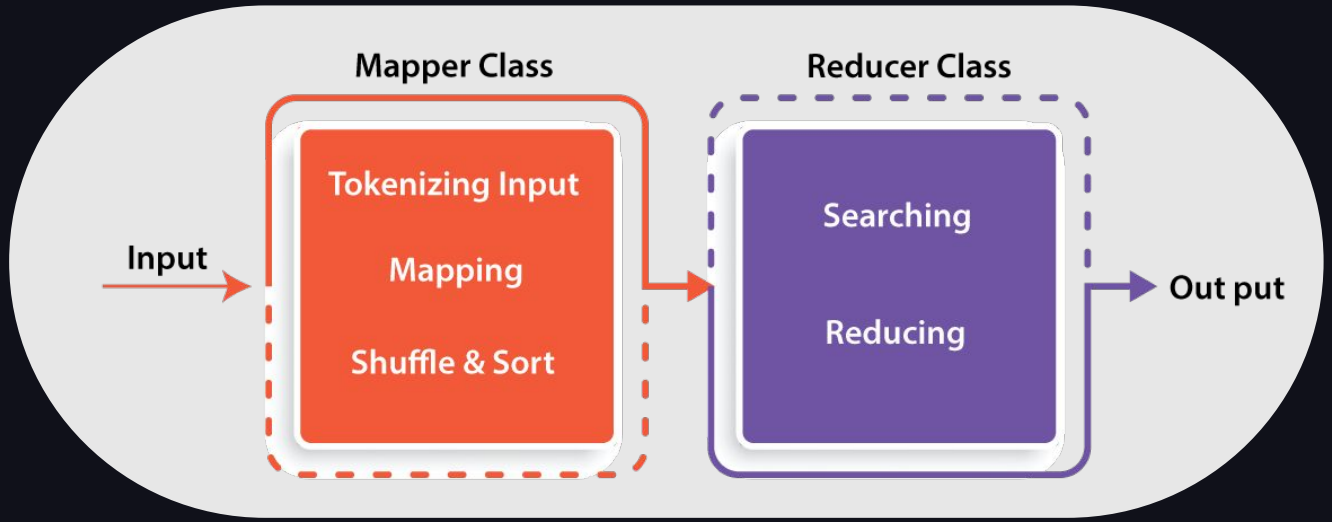


{ ..

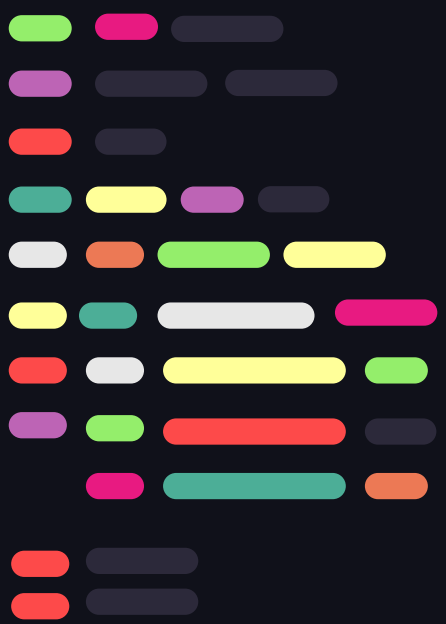
Overview of MapReduce

} ..

Key Algorithms



Here are four stages



Mapper

Map is a kind of user-defined function; this consists of series of key-value pairs and processes each key-value pair to generate more tuples data sets.

Reducer

The reducer takes the key-value paired group as an input value and runs them using Reducer functions, data is then aggregated/ integrated into one data set.

Combiner

A combiner is a kind of local reducer that helps to group similar data sets from the different map phases into identified sets.

Shuffle and Sort

The output of various mappers (k', v'), then goes into Shuffle and Sort phase. All the duplicate values are removed

Phases of MapReduce

Mapper

```java
public static class TempMapper extends Mapper<LongWritable, Text, Text, IntWritable> {

    private final static int MISSING = 9999;

    public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
        String line = value.toString();
        String[] fields = line.split(",");
        String year = fields[0].trim();
        String station = fields[1].trim();
        String tempStr = fields[2].trim();
        if (!tempStr.equals("Max Temperature")) {
            int temperature = Integer.parseInt(tempStr.substring(0, tempStr.indexOf("°F")).trim());
            context.write(new Text(year), new IntWritable(temperature));
        }
    }
}
```

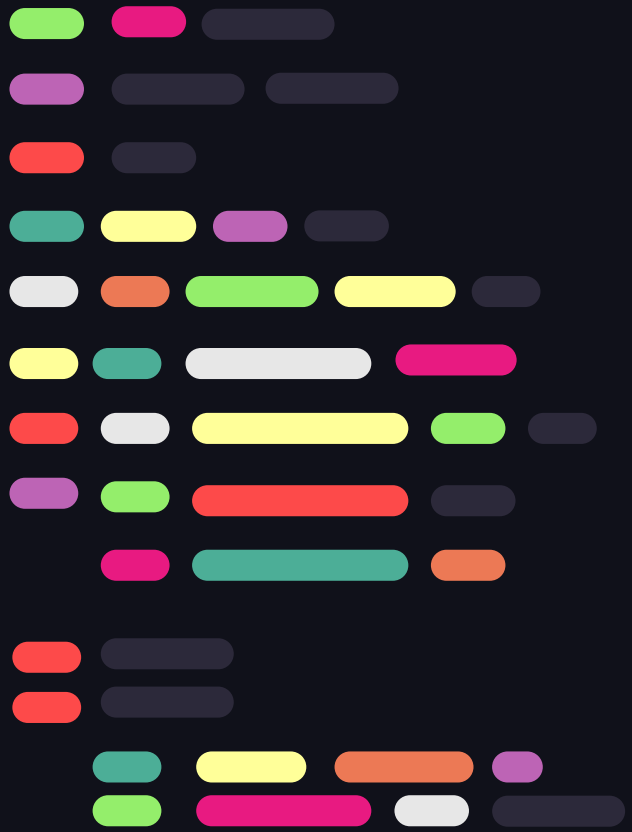
Reducer

```java
public static class TempReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {
        int maxTemp = Integer.MIN_VALUE;
        for (IntWritable value : values) {
            int temp = value.get();
            if (temp > maxTemp) {
                maxTemp = temp;
            }
        }
        context.write(key, new IntWritable(maxTemp));
    }
}
```

DEMONSTRATION

< It will utilize Eclipse, which runs on an Ubuntu virtual machine, to showcase the functionality of the application >



Thank You!