14/06/21 BDA Lab 2 Mohammad Tanvir
1BM18CS054

1. Map Reduce Program to add two numbers

Mapper:

```
import org.apache.hadoop.mapred.MapReduceBase ;
import org.apache.hadoop.mapred.Mapper ;

public class EOMapper extends MapReduceBase implements
        Mapper< LongWritable, Text, Text, IntWritable >
{
    public void map ( LongWritable key, Text value,
                      OutputCollector< Text, IntWritable > output,
                      Reporter rep )
    {
        String data[] = value.toString.split(" "); count = 0;
        for ( String num : data )
        {
            int number = Integer.parseInt(num);
            if ( number == NULL )
            {
                continue;
            }
            else
            {
                count += 1;
                output.collect( new Text("Number"),
                                new IntWritable(number) );
            }
        }
    }
}
```


Tanvir

Reducer:

```
public void Reduce (Text key, Iterator < IntWritable> value,
                    OutputCollector <Text, IntWritable> output,
                    Reporter rep)
{   int sum = 0;
    if (key.Equals ("Number")
    {   while (value.hasNext())
        {   IntWritable i = value.next();
            sum += i.get();
        }
    Output.collect ( key, new Text ("Sum"), new
                                IntWritable (sum));
```

Driver:

```
public int run (String [] args) throws Exception
{   if (args.length < 2)
    {   System.out.println ("Invalid arguments");
        return -1;
    }
    JobConf conf = new JobConf (EODriver.class);
    FileInputFormat.setInputPaths (conf, new Path (args[0]));
    FileOutputFormat.setOutputPath (conf, new Path (args[1]));
    Conf.setMapperClass (EOMapper.class);
    conf.setReducerClass (EOReducer.class);
```

Mohammad Tanvir
1BM16CS058

```
conf.setMapOutputValueClass(IntWritable.class);
conf.setMapOutputKeyClass(Text.class);
conf.OutputKeyClass(Int.class);
conf.OutputValueClass(IntWritable.class);

JobClient.runJob(conf);
return 0;
}
```

Main

```
public static void main(String args[])
{
    int exitcode = ToolRunner.run(new EODriver(), args);
    System.out.println("Driver exited with code" + exitcode);
    return 0;
}
}
```

Scala / Spark

UDF to return largest of 2 numbers

```
Object LargestofTwo {
    def getLarge(x:Int, y:Int):Int ={
        var largenum : Int = 0;
        if (x > y)
            largenum = x;
        else
            largenum = y;
```


Tanvir

```
    return largenum;
}

def main (args: Array[String])
{
    var a: Int = 30;
    var b: Int = 10;
    println("Largest number between" +a+ "and" + b + " is: "
        + getLarge(a,b));
}
}
```

VIVA