**BDA LAB TEST 2 – CODE + OUTPUT**

1. **Mapper Code**

|  |  |
| --- | --- |
| // Importing libraries  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.mapred.MapReduceBase;  import org.apache.hadoop.mapred.Mapper;  import org.apache.hadoop.mapred.OutputCollector;  import org.apache.hadoop.mapred.Reporter;  public class EOMapper extends MapReduceBase implements Mapper<LongWritable,  Text, Text, IntWritable> {  @Override  // Map function  public void map(LongWritable key, Text value, OutputCollector<Text,  IntWritable> output, Reporter rep)  throws IOException  {  // Splitting the line into spaces  String data[] = value.toString().split(" ");  for (String num : data)  {  int number = Integer.parseInt(num);  if (number % 2 == 1)  {  // For Odd Numbers  output.collect(new Text("ODD"), new IntWritable(number));  }  else  {  // For Even Numbers  output.collect(new Text("EVEN"),  new IntWritable(number));  }  }  }  } |  |

1. **Reducer Code**

|  |
| --- |
| 1. // Importing libraries 2. import java.io.IOException; 3. import java.util.Iterator; 4. import org.apache.hadoop.io.IntWritable; 5. import org.apache.hadoop.io.Text; 6. import org.apache.hadoop.mapred.MapReduceBase; 7. import org.apache.hadoop.mapred.OutputCollector; 8. import org.apache.hadoop.mapred.Reducer; 9. import org.apache.hadoop.mapred.Reporter; 10. public class EOReducer extends MapReduceBase implements Reducer<Text, 11. IntWritable, Text, IntWritable> { 12. @Override 13. // Reduce Function 14. public void reduce(Text key, Iterator<IntWritable> value, 15. OutputCollector<Text, IntWritable> output, Reporter rep) 16. throws IOException 17. { 18. // For finding sum and count of even and odd 19. // you don't have to take different variables 20. int sum = 0, count = 0; 21. if (key.equals("ODD")) 22. { 23. while (value.hasNext()) 24. { 25. IntWritable i = value.next(); 26. // Finding sum and count of ODD Numbers 27. sum += i.get(); 28. count++; 29. } 30. } 31. else 32. { 33. while (value.hasNext()) 34. { 35. IntWritable i = value.next(); 36. // Finding sum and count of EVEN Numbers 37. sum += i.get(); 38. count++; 39. } 40. } 41. // First sum then count is printed 42. output.collect(key, new IntWritable(sum)); 43. output.collect(key, new IntWritable(count)); 44. } 45. } |

1. **Driver Code**

|  |
| --- |
| 1. // Importing libraries 2. import org.apache.hadoop.conf.Configured; 3. import org.apache.hadoop.fs.Path; 4. import org.apache.hadoop.io.IntWritable; 5. import org.apache.hadoop.io.Text; 6. import org.apache.hadoop.mapred.FileInputFormat; 7. import org.apache.hadoop.mapred.FileOutputFormat; 8. import org.apache.hadoop.mapred.JobClient; 9. import org.apache.hadoop.mapred.JobConf; 10. import org.apache.hadoop.util.Tool; 11. import org.apache.hadoop.util.ToolRunner; 12. public class EODriver extends Configured implements Tool { 13. @Override 14. public int run(String[] args) throws Exception 15. { 16. if (args.length < 2) 17. { 18. System.out.println("Please enter valid arguments"); 19. return -1; 20. } 21. JobConf conf = new JobConf(EODriver.class); 22. FileInputFormat.setInputPaths(conf, new Path(args[0])); 23. FileOutputFormat.setOutputPath(conf, new Path(args[1])); 24. conf.setMapperClass(EOMapper.class); 25. conf.setReducerClass(EOReducer.class); 26. conf.setMapOutputKeyClass(Text.class); 27. conf.setMapOutputValueClass(IntWritable.class); 28. conf.setOutputKeyClass(Text.class); 29. conf.setOutputValueClass(IntWritable.class); 30. JobClient.runJob(conf); 31. return 0; 32. } 33. // Main Method 34. public static void main(String args[]) throws Exception 35. { 36. int exitcode = ToolRunner.run(new EODriver(), args); 37. System.out.println(exitcode); 38. } 39. } |

**OUTPUT SS:**

