```
import org.apache.hadoop.conf.Configuration;
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.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import java.io.IOException;
import java.util.ArrayList;

public class MatrixMultiplication {

  public static class MatrixMapper extends Mapper<Object, Text, Text, Text> {
    public void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
      String[] tokens = value.toString().split(",");
      String matrixName = tokens[0];
      int row = Integer.parseInt(tokens[1]);
      int col = Integer.parseInt(tokens[2]);
      int val = Integer.parseInt(tokens[3]);

      if (matrixName.equals("A")) {
        for (int k = 0; k < 3; k++) {  // Assuming B has 3 columns
          context.write(new Text(row + "," + k), new Text("A," + col + "," + val));
        }
      } else if (matrixName.equals("B")) {
        for (int i = 0; i < 3; i++) {  // Assuming A has 3 rows
          context.write(new Text(i + "," + col), new Text("B," + row + "," + val));
        }
      }
    }
  }

  public static class MatrixReducer extends Reducer<Text, Text, Text, IntWritable> {
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
      ArrayList<int[]> aElements = new ArrayList<>();
      ArrayList<int[]> bElements = new ArrayList<>();

      for (Text val : values) {
        String[] tokens = val.toString().split(",");
        int[] pair = {Integer.parseInt(tokens[1]), Integer.parseInt(tokens[2])};

        if (tokens[0].equals("A")) {
          aElements.add(pair);
```

```
                } else {
                    bElements.add(pair);
                }
            }

            int sum = 0;
            for (int[] a : aElements) {
                for (int[] b : bElements) {
                    if (a[0] == b[0]) {  // Multiply only if column index matches row index
                        sum += a[1] * b[1];
                    }
                }
            }

            context.write(key, new IntWritable(sum));
        }
    }

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Matrix Multiplication");
        job.setJarByClass(MatrixMultiplication.class);
        job.setMapperClass(MatrixMapper.class);
        job.setReducerClass(MatrixReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```