/*MovieTagAnalysis.java*/

/*Develop a MapReduce program to find the tags associated with each movie by analysing movie lens data. [refer tags.txt file in the teams (input file)] */

```
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.HashSet;

public class MovieTagAnalysis {

   // Mapper Class: Extracts movieId and tag
   public static class MovieTagMapper extends Mapper<Object, Text, IntWritable, Text> {
      private IntWritable movieId = new IntWritable();
      private Text tag = new Text();

      public void map(Object key, Text value, Context context) throws IOException,
InterruptedException {
         String line = value.toString();

         // Skip header line
         if (line.startsWith("userId")) {
            return;
```

```
        }

        String[] fields = line.split(",");

        if (fields.length >= 3) {
            try {
                int id = Integer.parseInt(fields[1].trim());
                movieId.set(id);
                tag.set(fields[2].trim());
                context.write(movieId, tag);
            } catch (NumberFormatException e) {
                // Ignore invalid lines
            }
        }
    }
}

// Reducer Class: Aggregates tags per movie
public static class MovieTagReducer extends Reducer<IntWritable, Text, IntWritable, Text> {
    public void reduce(IntWritable key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
        HashSet<String> uniqueTags = new HashSet<>();

        for (Text val : values) {
            uniqueTags.add(val.toString());
        }

        String result = String.join(", ", uniqueTags);
        context.write(key, new Text(result));
    }
}
```

```
    // Driver Class: Sets up and runs the MapReduce job
    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: MovieTagAnalysis <input path> <output path>");
            System.exit(-1);
        }

        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Movie Tags Analysis");

        job.setJarByClass(MovieTagAnalysis.class);
        job.setMapperClass(MovieTagMapper.class);
        job.setReducerClass(MovieTagReducer.class);

        job.setMapOutputKeyClass(IntWritable.class);
        job.setMapOutputValueClass(Text.class);
        job.setOutputKeyClass(IntWritable.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);
    }
}
```