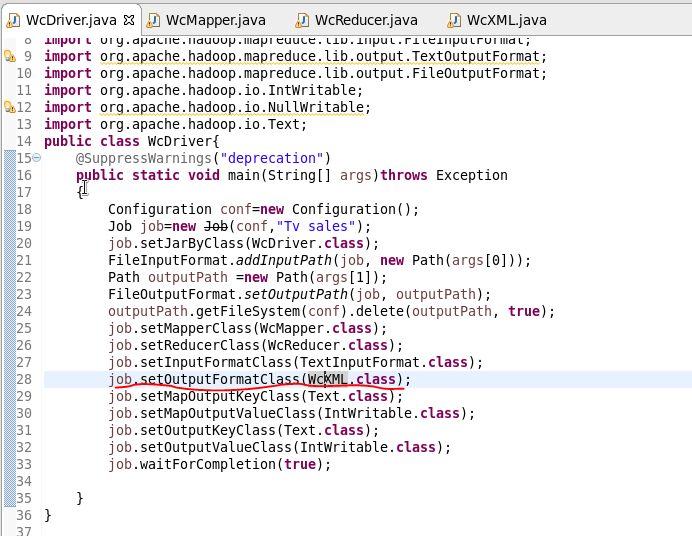
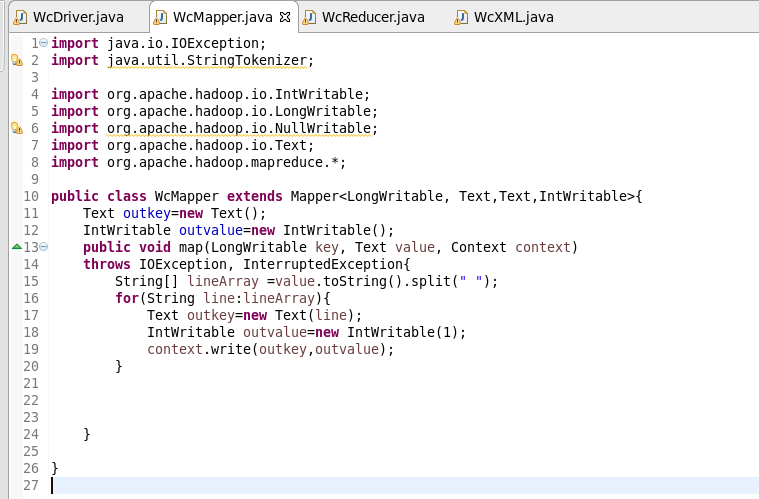
**Assignment 7.6**

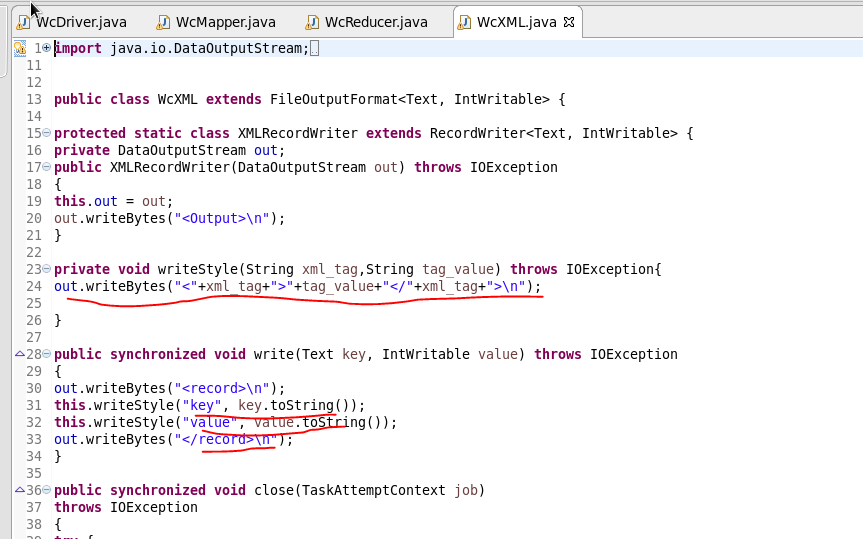
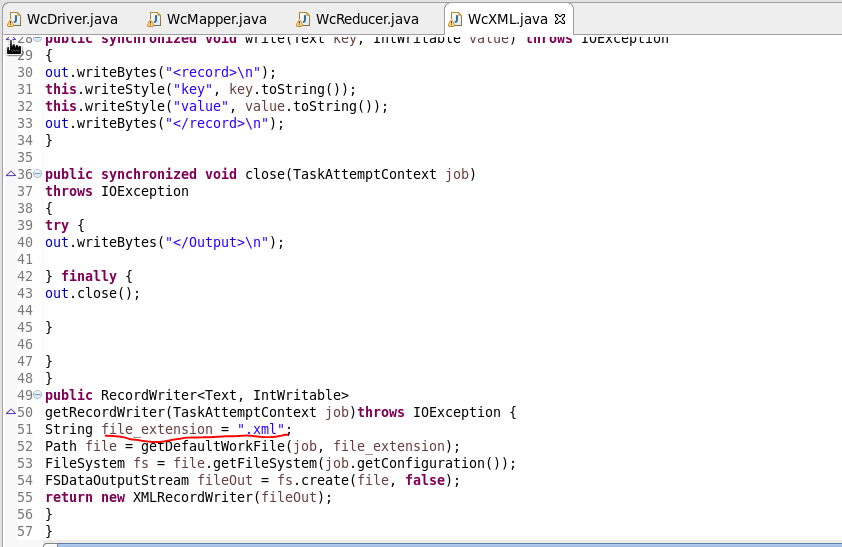
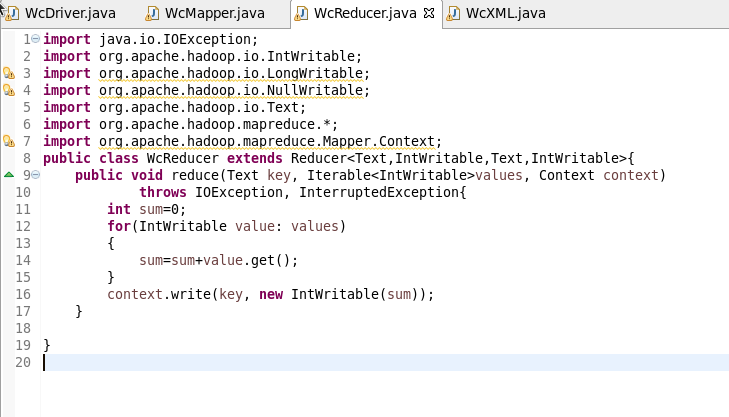
**Question:** Write a map reduce program save the output of the word count program in XML format. You need to write a custom writable to save the output in XML format.

***Codes:***

import org.apache.hadoop.fs.Path;  
import java.io.IOException;  
import org.apache.hadoop.conf.\*;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
public class WcDriver{  
    @SuppressWarnings("deprecation")  
    public static void main(String[] args)throws Exception  
    {  
        Configuration conf=new Configuration();  
        Job job=new Job(conf,"Tv sales");  
        job.setJarByClass(WcDriver.class);  
        FileInputFormat.addInputPath(job, new Path(args[0]));  
        Path outputPath =new Path(args[1]);  
        FileOutputFormat.setOutputPath(job, outputPath);  
        outputPath.getFileSystem(conf).delete(outputPath, true);  
        job.setMapperClass(WcMapper.class);  
        job.setReducerClass(WcReducer.class);  
        job.setNumReduceTasks(3);  
        job.setPartitionerClass(WcPartitioner.class);  
        job.setInputFormatClass(TextInputFormat.class);  
        job.setOutputFormatClass(TextOutputFormat.class);  
        job.setMapOutputKeyClass(Text.class);  
        job.setMapOutputValueClass(IntWritable.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
        job.waitForCompletion(true);  
    }  
}  
import java.io.IOException;  
import java.util.StringTokenizer;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
public class WcMapper extends Mapper<LongWritable, Text,Text,IntWritable>{  
    Text outkey=new Text();  
    IntWritable outvalue=new IntWritable();  
    public void map(LongWritable key, Text value, Context context)  
    throws IOException, InterruptedException{  
        String[] lineArray =value.toString().split(" ");  
        for(String line:lineArray){  
            Text outkey=new Text(line);  
            IntWritable outvalue=new IntWritable(1);  
            context.write(outkey,outvalue);  
        }  
    }  
}  
  
  
  
import java.io.IOException;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.LongWritable;  
import org.apache.hadoop.io.NullWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.\*;  
import org.apache.hadoop.mapreduce.Mapper.Context;  
public class WcReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
    public void reduce(Text key, Iterable<IntWritable>values, Context context)  
            throws IOException, InterruptedException{  
        int sum=0;  
        for(IntWritable value: values)  
        {  
            sum=sum+value.get();  
        }  
        context.write(key, new IntWritable(sum));  
    }  
}

import java.io.DataOutputStream;  
import java.io.IOException;  
import org.apache.hadoop.fs.FSDataOutputStream;  
import org.apache.hadoop.fs.FileSystem;  
import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.\*;  
import org.apache.hadoop.mapreduce.RecordWriter;  
import org.apache.hadoop.mapreduce.TaskAttemptContext;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
public class XMLOutputFormat extends FileOutputFormat<Text, IntWritable> {  
protected static class XMLRecordWriter extends RecordWriter<Text, IntWritable> {  
private DataOutputStream out;  
public XMLRecordWriter(DataOutputStream out) throws IOException  
{  
this.out = out;  
out.writeBytes("<Output>\n");  
}  
private void writeStyle(String xml\_tag,String tag\_value) throws IOException{  
out.writeBytes("<"+xml\_tag+">"+tag\_value+"</"+xml\_tag+">\n");  
}  
public synchronized void write(Text key, IntWritable value) throws IOException  
{  
out.writeBytes("<record>\n");  
this.writeStyle("key", key.toString());  
this.writeStyle("value", value.toString());  
out.writeBytes("</record>\n");  
}  
public synchronized void close(TaskAttemptContext job)  
throws IOException  
{  
try   
{  
out.writeBytes("</Output>\n");  
}  
 finally   
{  
out.close();  
 }  
 }  
}  
public RecordWriter<Text, IntWritable>  
getRecordWriter(TaskAttemptContext job)throws IOException  
 {  
String file\_extension = ".xml";  
Path file = getDefaultWorkFile(job, file\_extension);  
FileSystem fs = file.getFileSystem(job.getConfiguration());  
FSDataOutputStream fileOut = fs.create(file, false);  
return new XMLRecordWriter(fileOut);  
 }  
}

**OUTPUT:**   
  