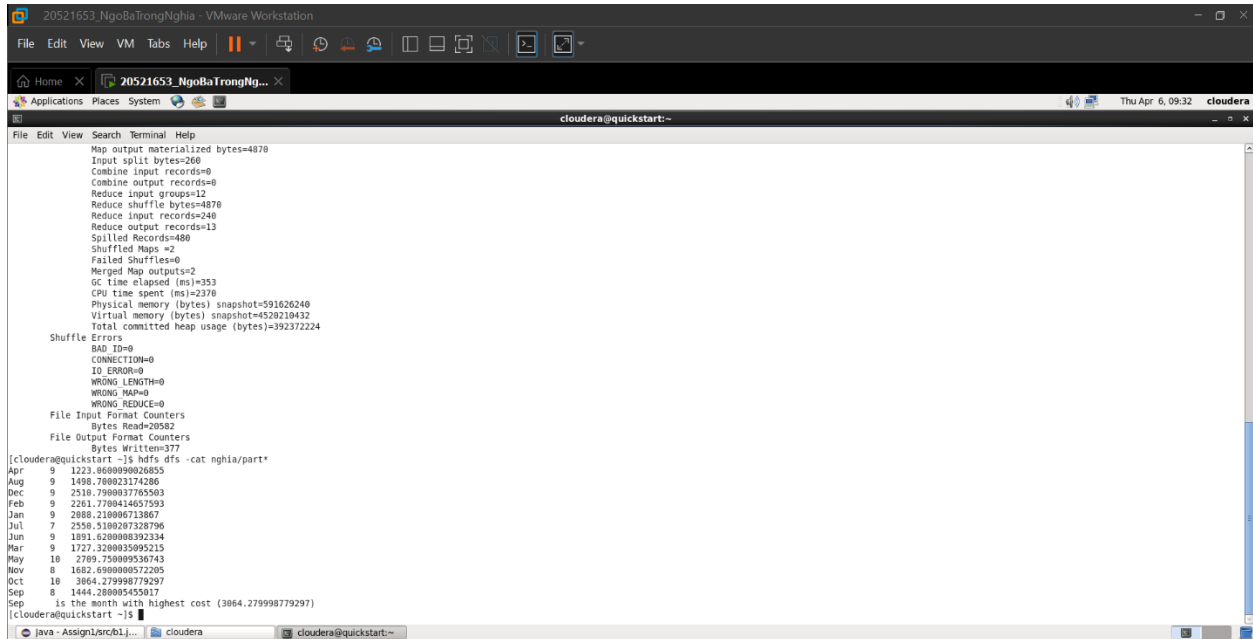


**Tên:** Ngô Bá Trọng Nghĩa

**MSSV:** 20521653

## Ex1:



The screenshot shows a terminal window titled '20521653\_NgoBaTrongNghia - VMware Workstation'. The terminal is running a Hadoop job, displaying various metrics such as 'Map output materialized bytes=4870', 'Input split bytes=260', 'Combine input records=0', 'Combine output records=0', 'Reduce input groups=12', 'Reduce shuffle bytes=4870', 'Reduce input records=240', 'Reduce output records=13', 'Spilled Records=480', 'Shuffled Maps=2', 'Failed Shuffles=0', 'Merged Map outputs=2', 'GC time elapsed (ms)=353', 'CPU time spent (ms)=2370', 'Physical memory (bytes) snapshot=591626240', 'Virtual memory (bytes) snapshot=4520210432', and 'Total committed heap usage (bytes)=392372224'. Below these metrics, there is a section for 'Shuffle Errors' which lists 'BAD ID=0', 'CONNECTION=0', 'IO ERROR=0', 'WRONG LENGTH=0', 'WRONG MAP=0', and 'WRONG REDUCE=0'. The terminal also shows 'File Input Format Counters' with 'Bytes Read=20582' and 'File Output Format Counters' with 'Bytes Written=377'. At the bottom, the terminal shows the command '[cloudera@quickstart ~]\$ hdfs dfs -cat nghia/part\*' and its output, which is a list of files and their sizes.

```
File Edit View Search Terminal Help
cloudera@quickstart:~
Map output materialized bytes=4870
Input split bytes=260
Combine input records=0
Combine output records=0
Reduce input groups=12
Reduce shuffle bytes=4870
Reduce input records=240
Reduce output records=13
Spilled Records=480
Shuffled Maps=2
Failed Shuffles=0
Merged Map outputs=2
GC time elapsed (ms)=353
CPU time spent (ms)=2370
Physical memory (bytes) snapshot=591626240
Virtual memory (bytes) snapshot=4520210432
Total committed heap usage (bytes)=392372224

Shuffle Errors
BAD ID=0
CONNECTION=0
IO ERROR=0
WRONG LENGTH=0
WRONG MAP=0
WRONG REDUCE=0

File Input Format Counters
Bytes Read=20582
File Output Format Counters
Bytes Written=377
[cloudera@quickstart ~]$ hdfs dfs -cat nghia/part*
Apr  9 1223.8600990026855
Aug  9 1495.790023174286
Dec  9 2518.7900037765503
Feb  9 2261.7700414657593
Jan  9 2080.210096713867
Jul  7 2550.5100207328796
Jun  9 1891.6200008392334
Mar  9 1727.320035595215
May 10 2769.750009536743
Nov  8 1862.690000572205
Oct 10 3864.279998779297
Sep  8 1444.280005455017
Sep  is the month with highest cost (3864.279998779297)
[cloudera@quickstart ~]$
```

## Code java:

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
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.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

public class Ex1 {

    //read trans file and show the list of: Game_type Total_amount
```

```

public static class TransMapper extends Mapper <Object, Text, Text, Text>
{
    public void map(Object key, Text value, Context context)
    throws IOException, InterruptedException
    {
        String record = value.toString().trim();
        String[] parts = record.split(",");

        String date = parts[1];
        String id = parts[2];
        String amount = parts[3];

        String month = date.split("-")[0];

        context.write(new Text(month), new Text(id + " " + amount));
    }
}

```

```

public static class TransReducer extends Reducer <Text, Text, Text, Text>
{

    public Double maxCost = 0.0;
    public Text maxMonth;
    public void reduce(Text key, Iterable<Text> values, Context context)
    throws IOException, InterruptedException
    {
        List<String> IDs = new ArrayList<String>();
        double total = 0.0;

        for (Text t : values)
        {
            String[] parts = t.toString().trim().split(" ");

            total += Float.parseFloat(parts[1]);
            String id = parts[0];

            if (!IDs.contains(id))
            {
                IDs.add(id);
            }

            if (total > maxCost) {
                maxMonth = key;
            }
        }
    }
}

```

```

        maxCost = total;
    }
}
context.write(key, new Text(IDs.size() + " " + Double.toString(total)));
}

```

@Override

```

public void cleanup(Context context) throws IOException, InterruptedException {
    //write the month with highest cost
    context.write(new Text(maxMonth), new Text(" is the month with highest cost (" +
String.valueOf(maxCost) + ")"));
}
}

```

```

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "Trans analysis 3");
    job.setJarByClass(Ex1.class);
    job.setMapperClass(TransMapper.class);
    job.setReducerClass(TransReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    //job.setNumReduceTasks(0);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

## Ex2:



The screenshot shows a terminal window titled '20521653\_NgoBaTrongNghia - VMware Workstation'. The terminal displays the output of a Hadoop MapReduce job. The output includes statistics such as 'Map output bytes=4611', 'Input split bytes=260', 'Combine input records=0', 'Reduce input groups=12', 'Reduce shuffle bytes=5183', 'Reduce input records=248', 'Reduce output records=12', 'Spilled Records=480', 'Shuffled Maps=2', 'Failed Shuffles=0', 'Merged Map outputs=2', 'GC time elapsed (ms)=481', 'CPU time spent (ms)=2430', 'Physical memory (bytes) snapshot=596484996', 'Virtual memory (bytes) snapshot=4528079368', and 'Total committed heap usage (bytes)=392372224'. It also lists 'Shuffle Errors' and 'File Input Format Counters'. At the bottom, there is a list of files and their sizes, such as 'Gretchen-2 Karen-2 Kristina-2 Elsie-3 Dolores-1 Paige-1 Hazel-2 Malcolm-1 Sherri-1'.

## Code java:

```
import java.io.*;
import java.util.*;
import java.net.URI;

import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Reducer.Context;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

public class Assignment2 {

    public static class TransMapper extends Mapper<Object, Text, Text, Text> {
        // user map to keep the userId-userName
        private Map<Integer, String> userMap = new HashMap<>();

        public void setup(Context context) throws IOException,
            InterruptedException {
            try (BufferedReader br = new BufferedReader(new FileReader(
```

```

        "cust.txt")))) {
    String line;
    while ((line = br.readLine()) != null) {
        String columns[] = line.split(",");
        String id = columns[0];
        String name = columns[1];
        userMap.put(Integer.parseInt(id), name);
    }
} catch (IOException e) {
    e.printStackTrace();
}
}

public void map(Object key, Text value, Context context)
    throws IOException, InterruptedException {
    String record = value.toString().trim();
    String[] parts = record.split(",");

    String date = parts[1];
    String id = parts[2];
    String name = userMap.get(Integer.parseInt(id));

    String month = date.split("-")[0];

    context.write(new Text(month), new Text(id + "," + name));
}

}

public static class TransReducer extends Reducer<Text, Text, Text, Text> {

    public void reduce(Text key, Iterable<Text> values, Context context)
        throws IOException, InterruptedException {
        List<String> Names = new ArrayList<String>();
        Map<String, Integer> userCount = new HashMap<String,
Integer>();

        for (Text t : values) {
            String[] parts = t.toString().trim().split(",");
            String name = parts[1];

            if (!Names.contains(name)) {
                Names.add(name);
                userCount.put(name, 1);
            }
        }
    }
}

```

```

        } else {
            userCount.put(name, userCount.get(name) + 1);
        }

    }

    String str = "";

    for (String name : Names) {
        str += name + "-" + userCount.get(name) + " ";
    }

    context.write(key, new Text(str));
}
}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = new Job(conf, "q2");
    job.setJarByClass(Assignment2.class);
    job.setMapperClass(TransMapper.class);
    job.setReducerClass(TransReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(Text.class);
    // Setting reducer to zero
    // job.setNumReduceTasks(0);

    try {

        job.addCacheFile(new URI(
            "hdfs://localhost:8020/user/cloudera/cust.txt"));
    } catch (Exception e) {
        System.out.println("File Not Added");
        System.exit(1);
    }

    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```