Exercise 03d: Earth Quack data analytics using MapReduce

Name : Annapoornima S
Roll no: 225229101

This exercise's MapReduce process is doing Earth Quack data analysis. This analysis is used to find maximum magnitude earth quack in each region. In this exercise students try to create Mapper and Reducer process using Java and Python.

Prerequisites

Ensure that Hadoop is installed, configured and is running. More details:Single Node Setup for first-time users.
Cluster Setup for large, distributed clusters.

Inputs and Outputs

- **Input file should be in : /earth/in/**

  WAData.txt

  Copy the content text from earth.csv, Which is attached in Google classroom.

- **Output file should be in /earth/out/**

Step 1:

Create and Compile `EarthQuack.java` and create an EarthQuack.jar:

- **Create `EarthQuack`.java project.**

```
import org.apache.hadoop.fs.Path;

import
org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;

import
org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
```

```
import
org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class EarthQuake
{

    public static void main(String[] args) throws
        Exception {if (args.length != 2) {
            System.err.println("Usage: hadoopex <input path> <output
            path>");System.exit(-1);
        }

        // Create the job specification object




        // Setup input and output paths



        // Set the Mapper and Reducer classes



        // Specify the type of output keys and values



        // Wait for the job to finish before terminating
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }

}
```

- **Create `EarthquakeMapper`.java project.**

```
import
org.apache.hadoop.io.DoubleWritable;
import
org.apache.hadoop.io.LongWritable;
```

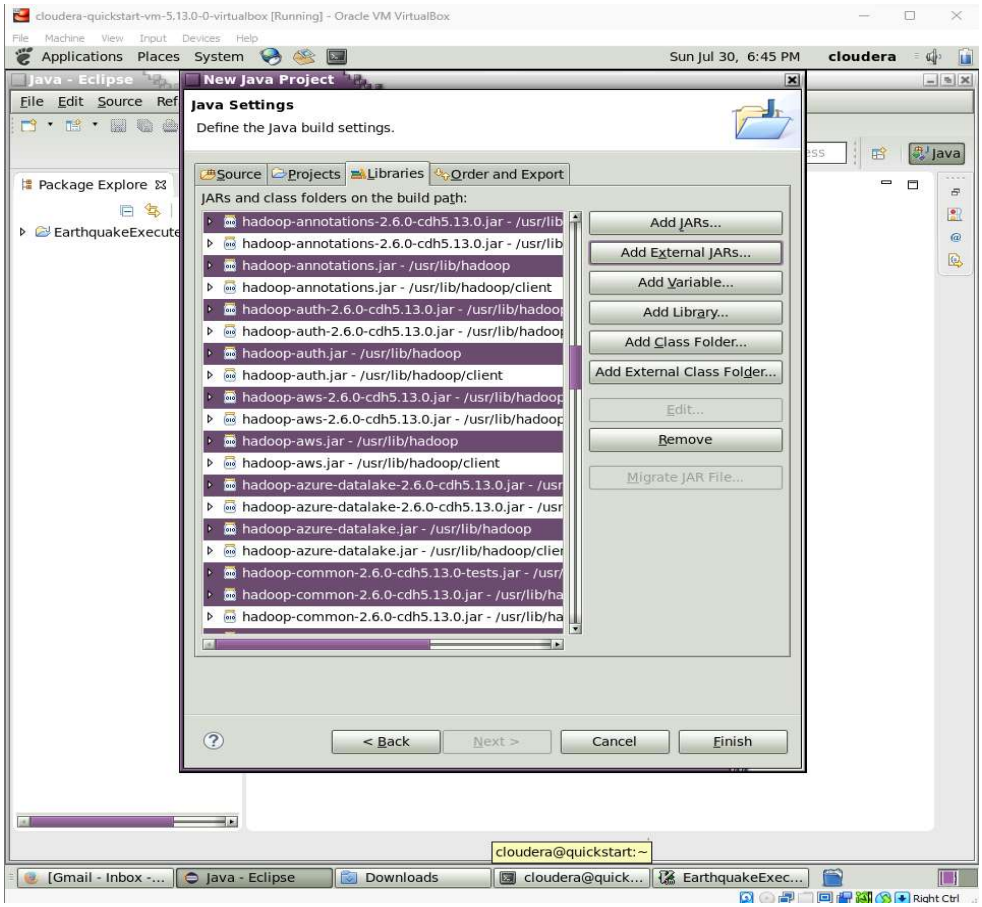
```
import org.apache.hadoop.io.Text;

import

org.apache.hadoop.mapreduce.Mapper;

import java.io.IOException;
public class EarthquakeMapper extends
        Mapper<LongWritable, Text, Text,
        DoubleWritable>
{


    @Override

    public void map(LongWritable key, Text value, Context context)
            throwsIOException, InterruptedException {

        String[] line = value.toString().split(",", 12);


        // Ignore
        invalid lines if
        (line.length !=
        12) {


            System.out.println("- " +
            line.length);return;
        }

        // The output `key` is the name of the region


        // The output `value` is the magnitude of the earthquake


        // Record the output in the Context object

    }
}
```

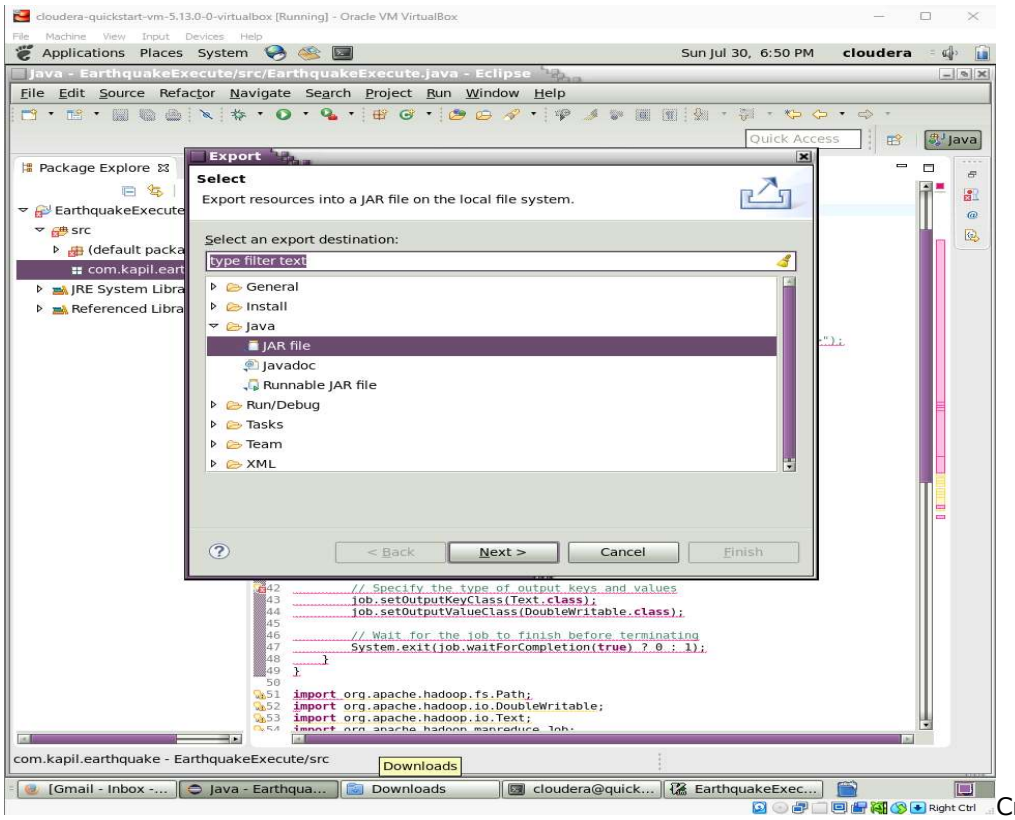
- **Create `EarthquakeMapper`.java project.**

```
import

org.apache.hadoop.io.DoubleWritable;

import

org.apache.hadoop.mapreduce.Reducer;

import java.io.IOException;

import

org.apache.hadoop.io.Text;

public class EarthquakeReducer

extends
        Reducer<Text, DoubleWritable, Text, DoubleWritable>

{

    @Override


    public void reduce(Text key, Iterable<DoubleWritable> values,

            Context context) throws IOException,

            InterruptedException {


        // Standard algorithm for finding the max value


        ______________________________________{


}
```

```
            context.write(key, new DoubleWritable(maxMagnitude));

        }

    }
```

- Import external .jar files



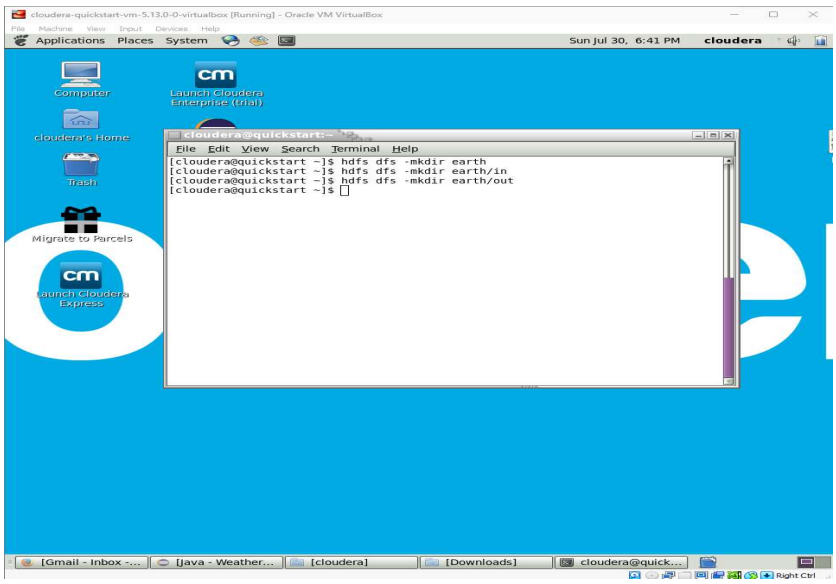
- 

Create EarthQuake.jar file

Step 2:

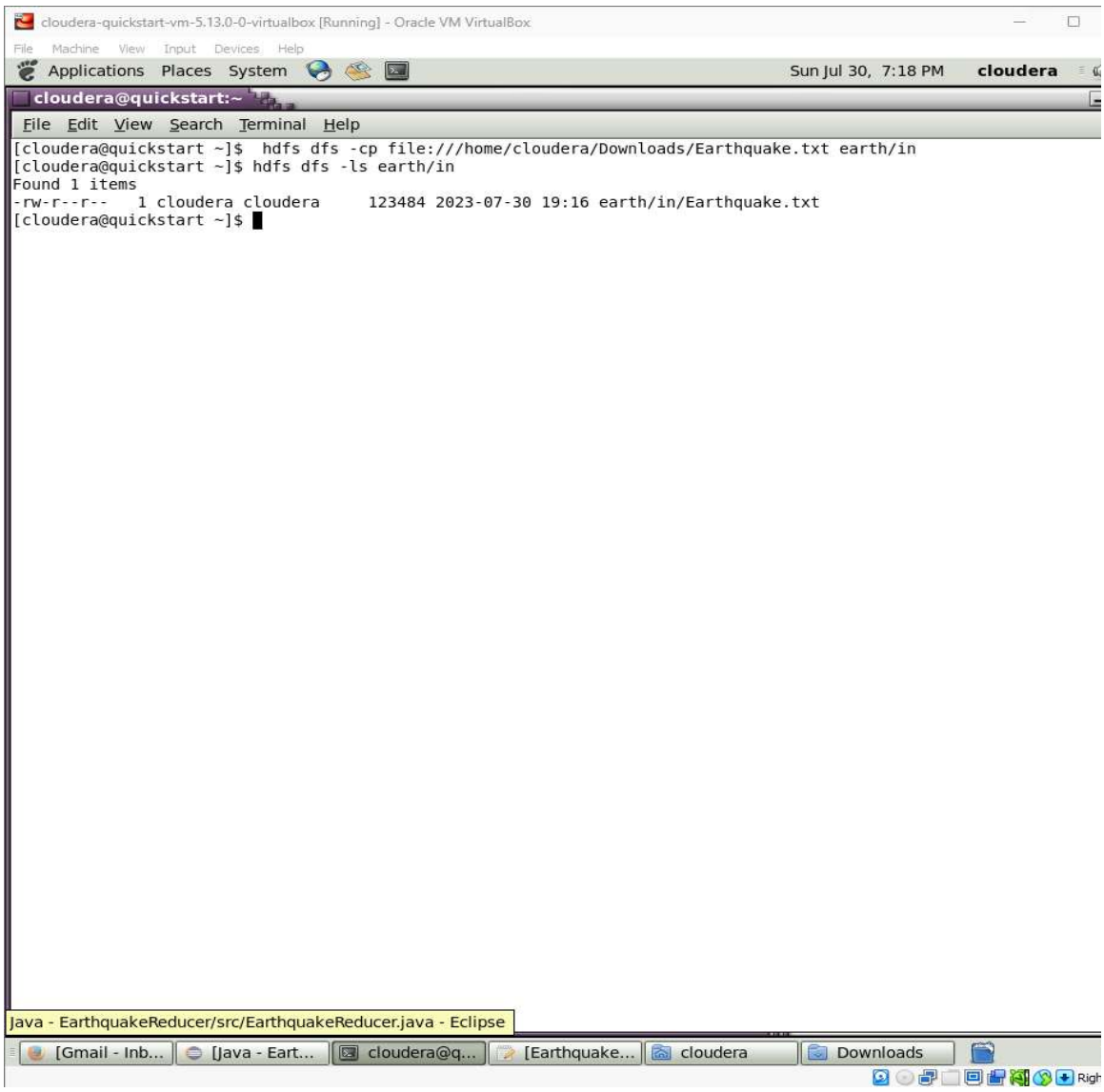
Create following folders in HDFS:

- `/earth/in` - input directory in HDFS
- `/earth/out` - output directory in HDFS



Step 3

Create and copy earth.txt-files into input folder:



[cloudera@quickstart ~]$ hdfs dfs -ls

/earth/in00/Found 1 items
-rw-r--r-- 1 cloudera supergroup 12054 2021-08-26 15:48 /earth/in/earth.txt

Step 4:

Run the MapReduce application :

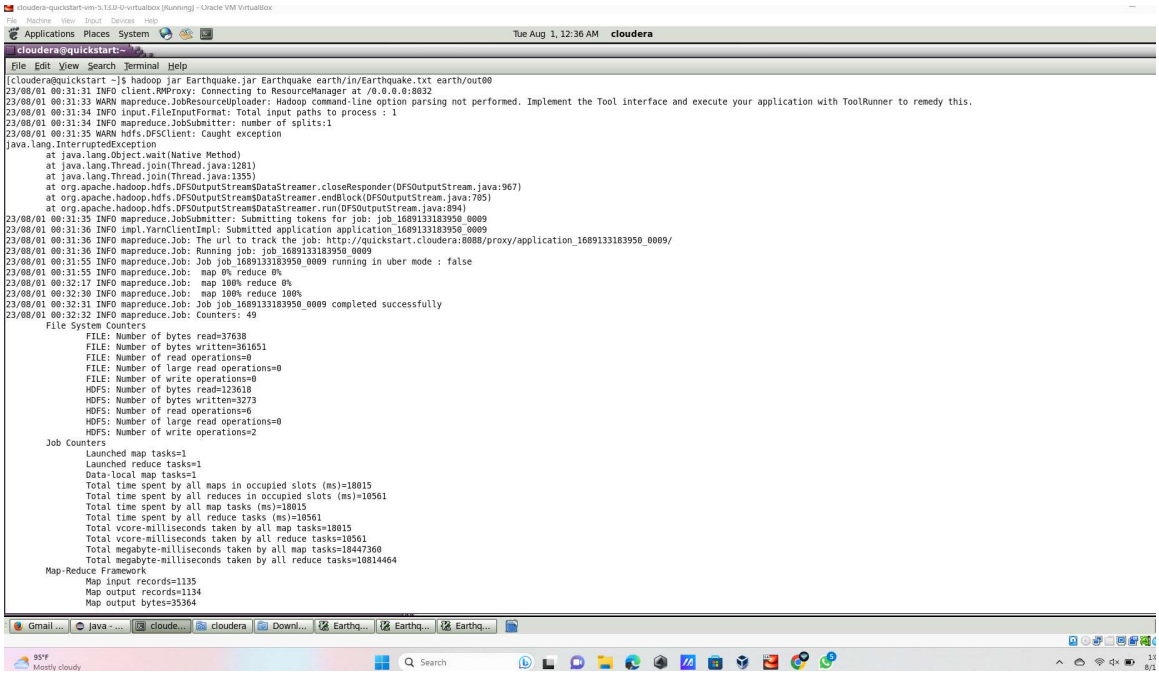
Java

[cloudera@quickstart ~]$ hadoop jar EarthQuake.jar EarthQuake /earth/in/earth.txt /earth/out/

Python

[cloudera@quickstart ~]$ hadoop jar /usr/lib/hadoop-0.20-mapreduce/contrib/streaming/hadoop- streaming-2.6.0-mr1-cdh5.13.0.jar -file /home/cloudera/map.py /home/cloudera/reduce.py - mapper "python map.py" -reducer "python reduce.py" -input /earth/in/earth.txt -output /earth/out

Show MapReduce Framework



```
[cloudera@quickstart ~]$ hadoop jar Earthquake.jar Earthquake earth/in/Earthquake.txt earth/out00
23/08/01 00:31:31 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
23/08/01 00:31:33 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
23/08/01 00:31:34 INFO input.FileInputFormat: Total input paths to process : 1
23/08/01 00:31:34 INFO mapreduce.JobSubmitter: number of splits:1
23/08/01 00:31:35 WARN hdfs.DFSClient: Caught exception
java.lang.InterruptedException
        at java.lang.Object.wait(Native Method)
        at java.lang.Thread.join(Thread.java:1281)
        at java.lang.Thread.join(Thread.java:1355)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.closeResponder(DFSOutputStream.java:967)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.endBlock(DFSOutputStream.java:705)
        at org.apache.hadoop.hdfs.DFSOutputStream$DataStreamer.run(DFSOutputStream.java:894)
23/08/01 00:31:35 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1689133183950_0009
23/08/01 00:31:36 INFO impl.YarnClientImpl: Submitted application application_1689133183950_0009
23/08/01 00:31:36 INFO mapreduce.Job: The url to track the job: http://quickstart.cloudera:8088/proxy/application_1689133183950_0009/
23/08/01 00:31:36 INFO mapreduce.Job: Running job: job_1689133183950_0009
23/08/01 00:31:55 INFO mapreduce.Job: Job job_1689133183950_0009 running in uber mode : false
23/08/01 00:31:55 INFO mapreduce.Job:  map 0% reduce 0%
23/08/01 00:32:17 INFO mapreduce.Job:  map 100% reduce 0%
23/08/01 00:32:30 INFO mapreduce.Job:  map 100% reduce 100%
23/08/01 00:32:31 INFO mapreduce.Job: Job job_1689133183950_0009 completed successfully
23/08/01 00:32:32 INFO mapreduce.Job: Counters: 49
        File System Counters
                FILE: Number of bytes read=37638
                FILE: Number of bytes written=361651
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=123618
                HDFS: Number of bytes written=3273
                HDFS: Number of read operations=6
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=18015
                Total time spent by all reduces in occupied slots (ms)=10561
                Total time spent by all map tasks (ms)=18015
                Total time spent by all reduce tasks (ms)=10561
                Total vcore-milliseconds taken by all map tasks=18015
                Total vcore-milliseconds taken by all reduce tasks=10561
                Total megabyte-milliseconds taken by all map tasks=18447360
                Total megabyte-milliseconds taken by all reduce tasks=10814464
        Map-Reduce Framework
                Map input records=1135
                Map output records=1134
                Map output bytes=35364
```

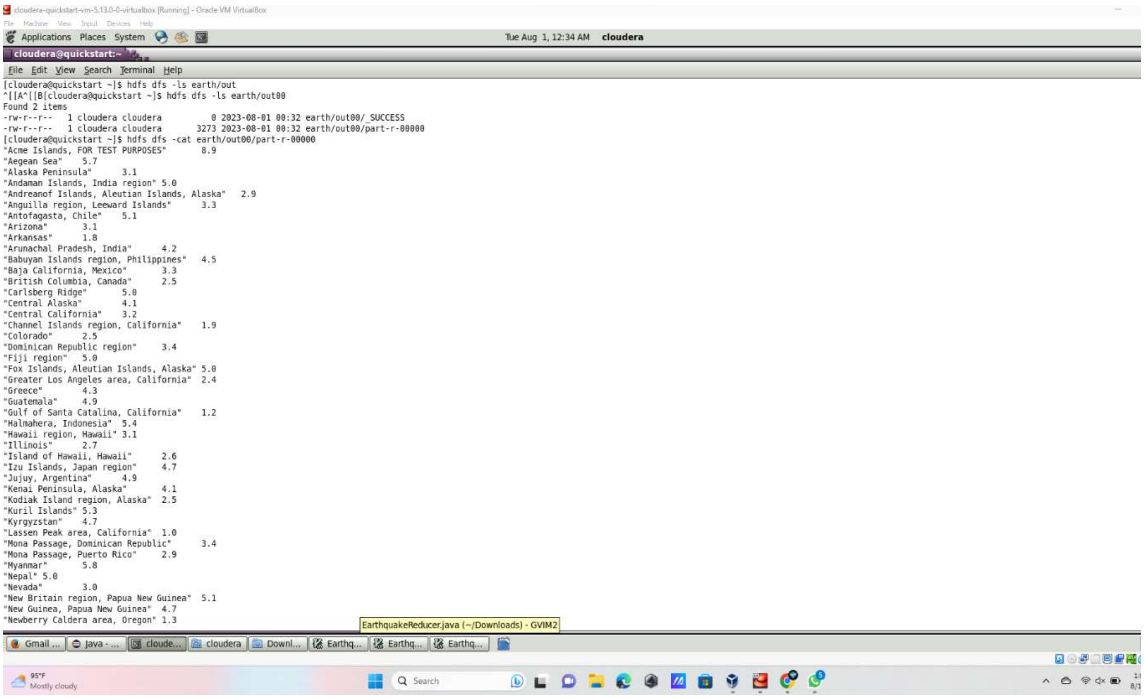
Step 5:

Output:

[cloudera@quickstart ~]$ hdfs dfs -ls

/earth/out/Found 2 items
-rw-r--r-- 1 cloudera supergroup 0 2021-08-26 15:50 /weather/out00/_SUCCESS

-rw-r--r-- 1 cloudera supergroup 228 2021-08-26 15:50 /weather/out00/part-r-00000[cloudera@quickstart ~]$ hdfs dfs -cat /earth/out/part-r-00000


cloudera@quickstart:~
File Edit View Search Terminal Help
[cloudera@quickstart ~]$ hdfs dfs -ls earth/out
^[[A^[[B[cloudera@quickstart ~]$ hdfs dfs -ls earth/out00
Found 2 items
-rw-r--r-- 1 cloudera cloudera 0 2023-08-01 00:32 earth/out00/_SUCCESS
-rw-r--r-- 1 cloudera cloudera 3273 2023-08-01 00:32 earth/out00/part-r-00000
[cloudera@quickstart ~]$ hdfs dfs -cat earth/out00/part-r-00000
"Acme Islands, FOR TEST PURPOSES" 8.9
"Aegean Sea" 5.7
"Alaska Peninsula" 3.1
"Andaman Islands, India region" 5.0
"Andreanof Islands, Aleutian Islands, Alaska" 2.9
"Anguilla region, Leeward Islands" 3.3
"Antofagasta, Chile" 5.1
"Arizona" 3.1
"Arkansas" 1.8
"Arunachal Pradesh, India" 4.2
"Babuyan Islands region, Philippines" 4.5
"Baja California, Mexico" 3.3
"British Columbia, Canada" 2.5
"Carlsberg Ridge" 5.0
"Central Alaska" 4.1
"Central California" 3.2
"Channel Islands region, California" 1.9
"Colorado" 2.5
"Dominican Republic region" 3.4
"Fiji region" 5.0
"Fox Islands, Aleutian Islands, Alaska" 5.0
"Greater Los Angeles area, California" 2.4
"Greece" 4.3
"Guatemala" 4.9
"Gulf of Santa Catalina, California" 1.2
"Halmahera, Indonesia" 5.4
"Hawaii region, Hawaii" 3.1
"Illinois" 2.7
"Island of Hawaii, Hawaii" 2.6
"Izu Islands, Japan region" 4.7
"Jujuy, Argentina" 4.9
"Kenai Peninsula, Alaska" 4.1
"Kodiak Island region, Alaska" 2.5
"Kuril Islands" 5.3
"Kyrgyzstan" 4.7
"Lassen Peak area, California" 1.0
"Mona Passage, Dominican Republic" 3.4
"Mona Passage, Puerto Rico" 2.9
"Myanmar" 5.8
"Nepal" 5.0
"Nevada" 3.0
"New Britain region, Papua New Guinea" 5.1
"New Guinea, Papua New Guinea" 4.7
"Newberry Caldera area, Oregon" 1.3
EarthquakeReducer.java (~/Downloads) - GVIM2