DHRUV GARG 16 BCE 1190

LARGE SCALE DATA PROCESSING - LAB

LAB EXPERIMENT 1: BASIC JAVA PROGRAMS (7th December 18)

```
public class P1_add {
   public static void main(String args[]){
      int a, b, c;
      System.out.println("Addition using Scanner");
      Scanner s = new Scanner(System.in);

      for( b = 1; b <= 10; b++)
      {
         System.out.println("Enter a : ");
         a = s.nextInt();
         c = a + b;
         System.out.println("Sum : "+ c);
      }
      s.close();
   }
}
```

```
import java.util.Scanner;
public class P2_fibonacci {
   public static void main(String args[])
   {
      int newNum = 0, firstNum = 1, secNum = 1, i = 0;
      int limit;

      Scanner s = new Scanner(System.in);
      limit = s.nextInt();

      for(i = 0; i < (limit-1); i++)
      {
         if(i == 0)
            {
               System.out.print(i + " " + (i+1) + " ");
               firstNum = 0; secNum = 1;
```

```
            }

        else if(i != 0)
            {
                newNum = firstNum + secNum;
                System.out.print( newNum + " ");
                firstNum = secNum;
                secNum = newNum;
            }
      }
      System.out.println();
      System.out.println("The first "+ limit + " fibonacci numbers have been printed.");

      s.close();
   }
}
```

```
import java.util.Scanner;
public class P3_array {
   public static void main(String args[])
   {
      int a[] = new int[10];
      int i, enteredVal;

      Scanner s = new Scanner(System.in);

      for(i = 0; i < 5; i++)
      {
         System.out.println("Enter a number : ");
         enteredVal = s.nextInt();
         a[i] = enteredVal;
      }

      System.out.println("Printing the values of the array");
      for(i = 0; i < 5; i++)
      {
```

```
        System.out.println(a[i]);
      }
      s.close();
   }
}
```

```
import java.util.Scanner;
public class P4_arraySort {
   public static void main(String args[])
   {
      int a[] = new int[10];
      int i, j, n, temp, enteredVal;

      Scanner s = new Scanner(System.in);
      System.out.println("Enter size of array (max 20) : ");
      n = s.nextInt();

      for(i = 0; i < n; i++)
      {
         System.out.println("Enter a number : ");
         enteredVal = s.nextInt();
         a[i] = enteredVal;
      }

      System.out.println();
      System.out.println("Sorting using bubble sort...");
      for (i = 0; i < n-1; i++)
         for (j = 0; j < n-i-1; j++)
            if (a[j] > a[j+1])
            {
               temp = a[j];
               a[j] = a[j+1];
               a[j+1] = temp;
            }
      System.out.println("Printing array in ascending order  after sorting");
      for(i = 0; i < n; i++)
```

```
        {
            System.out.println(a[i]);
        }
        s.close();
    }
}
```

```
import java.util.Scanner;
public class P5_functionCall {
    public static void sum(int a, int b)
    {
        int c = a + b;
        System.out.println("The obtained sum is: " + c);
    }

    public static void main(String args[])
    {
        Scanner s = new Scanner(System.in);
        int a, b;

        System.out.println("Enter the first number : ");
        a = s.nextInt();
        System.out.println("Enter the second number : ");
        b = s.nextInt();

        System.out.println("Calling the function sum to compute addition.");
        P5_functionCall.sum(a, b);
        s.close();
    }
}
```

```
import java.util.Scanner;
public class P6_ReverseString {
    public static void main(String args[]) throws Exception
    {
```

```
      String str, revStr;
      System.out.println("Enter a string to reverse : ");
      Scanner s = new Scanner(System.in);

      revStr = "";
      str = s.nextLine();

      char character;
      int i = 0;
      character = str.charAt(i);
      while(character != '\0')
      {
         try {
            character = str.charAt(i);
            revStr = character + revStr;
            if(str.charAt(i)== '\0')
               break;
            else
               i++;
         }
         catch(StringIndexOutOfBoundsException exception) {
            break;
         }
      }
      System.out.println("Original string " + str);
      System.out.println("Reversed string " + revStr);
      s.close();
   }
}
```

```
import java.util.Scanner;
public class P7_vowelConsonantCount {
   public static void main(String args[]) throws Exception
   {
      String str;
      System.out.println("Enter a string : ");
```

```
    Scanner s = new Scanner(System.in);

    str = s.nextLine();

    char character;
    int i = 0;
    int vowelCount = 0, consonantCount = 0;

    for(i = 0; i < str.length(); i++)
    {
       str = str.toLowerCase();
       character = str.charAt(i);

       switch(character)
       {
       case 'a':
       case 'e':
       case 'i':
       case 'o':
       case 'u':
          {
             vowelCount += 1;
             break;
          }

       default:
          consonantCount += 1;
          }
    }
    System.out.println("Vowel count : "+ vowelCount);
    System.out.println("Consonant count : "+ consonantCount);
    s.close();
  }
}
```

DHRUV GARG 16 BCE 1190

LAB EXPERIMENT 2: BASIC HDFS COMMANDS (14th December 2018)

start-all.sh

jps

hadoop fs -ls

hadoop fs -copyFromLocal /home/ponny/Desktop/hadoop_file1 /dhruv/file1

hadoop fs -put file:/home/ponny/Desktop/hadoop_file1 hdfs:/dhruv/file2

OR

hadoop fs -put file:///home/ponny/Desktop/hadoop_file1 hdfs:///dhruv/file2

hadoop fs -copyToLocal /dhruv/file1 /home/ponny/Desktop/from_hadoop

hadoop fs -get hdfs:///dhruv/file3 file:///home/ponny/Desktop/frmHadoop_usingGet

hadoop fs -mv /dhruv/file1 /dhruv/file3

hadoop fs -rm /dhruv/file2

stop-all.sh

jps

LAB EXPERIMENT: PRACTISE JAVA PROGRAMS (21st December 2018)

```
import java.util.Scanner;
import java.util.Arrays;


public class string_letterSort {
    public static void main(String[] args) {
        System.out.println("Enter the string:");
        Scanner scan = new Scanner(System.in);
        String sentence="", sortedSentence = "";

        sentence+=scan.nextLine();
        scan.close();
        System.out.println("The sentence is :"+sentence);
        sentence = sentence.replaceAll("\\s","");
        System.out.println("The sentence is :"+sentence);
        char tempSentence[] = sentence.toCharArray();

        Arrays.sort(tempSentence);
        sortedSentence = new String(tempSentence);
        System.out.println("The sentences with sorted letters order :"+sortedSentence);
    }
}
```

```
import java.util.Scanner;


public class sortNumByFreq {

    public static void main(String args[])
    {
        int a[] = new int[7];
        int i, j, enteredVal;
        int current, count, printedNos = 0, frequency = 1;

        Scanner s = new Scanner(System.in);

        for(i = 0; i < 7; i++)
        {
            System.out.println("Enter a number : ");
            enteredVal = s.nextInt();
            a[i] = enteredVal;
        }

        System.out.println("Printing the values of the array");
        for(i = 0; i < 7; i++)
        {
            System.out.println(a[i]);
        }
        s.close();

        for(i = 0; i < 7; i++)
        {
```

```
            current = a[i];
            count = 0;
            for(j = 0; j < 7; j++)
            {
                if(a[j] == current)
                    count += 1;
            }

            //System.out.println("Current : " + current);
            //System.out.println("Count : " + count);

            if(count == frequency)
            {
                System.out.println("Printing value : " + current);
                printedNos += 1;
                //System.out.println("Printed nos : " + printedNos);
            }

            if((printedNos != 7) && (i < 6))
            {
                continue;
            }

            if((printedNos != 7)&&(i == 6))
            {
                System.out.println("Restarting loop");
                i = 0;
                frequency = frequency + 1;
                //System.out.println("Frequency : " + frequency);
            }

            if(printedNos == 7)
            {
                System.out.println("Program terminated.");
                break;
            }

            /*if((count < frequency)&&(printedNos != 7))
            {
                System.out.println("Printing value : " + current);
                printedNos += 1;
            }*/

            if(frequency > 7)
                break;
        }
    }
}
```

DHRUV GARG 16 BCE 1190

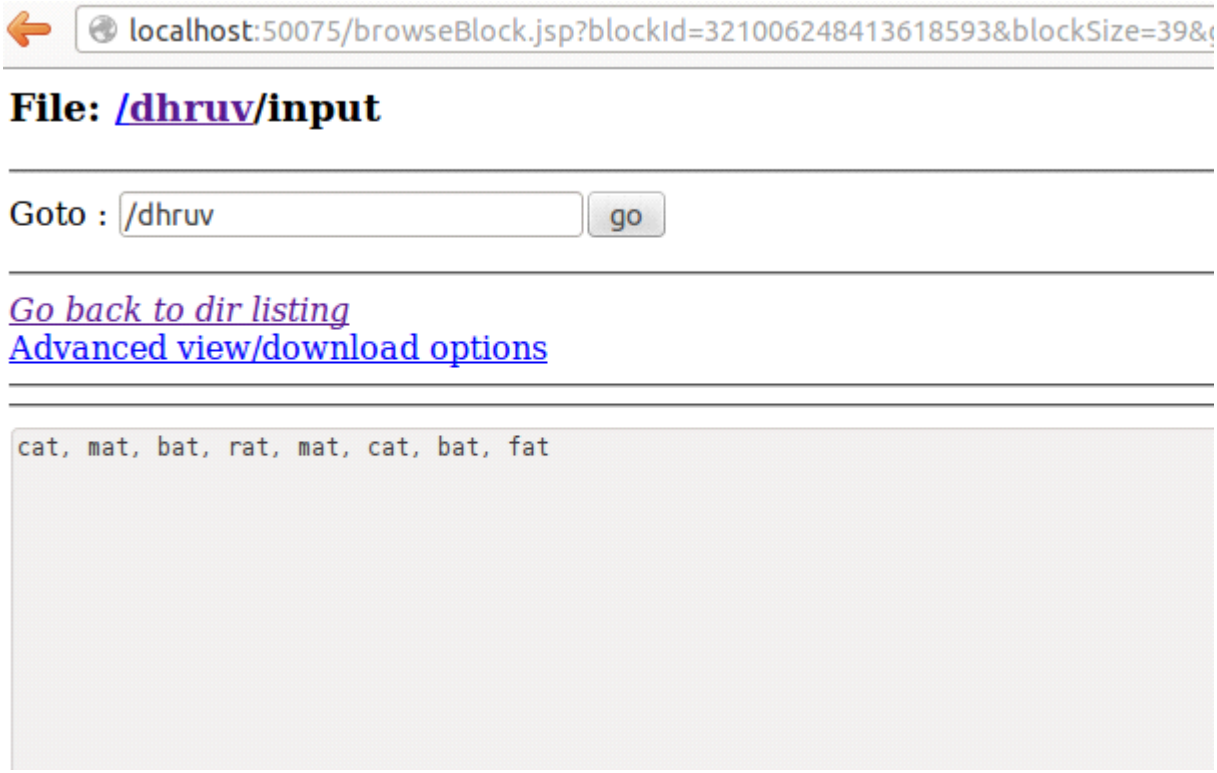
LAB EXPERIMENT 3: WORDCOUNT PROGRAM (11th January 2019)

CODE:

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
    public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>{
        //private final static IntWritable one = new IntWritable(1);
        IntWritable one = new IntWritable(1);
        //private Text word = new Text();

        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
            String[] line = value.toString().split(",");
            for(String lines:line){
                context.write(new Text(lines), one);
                }
            }
    }

public static class Reduce extends Reducer <Text,IntWritable, Text,IntWritable> {
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException{
        int sum = 0;
        for(IntWritable val : values){
            sum += val.get();
        }
        context.write(key, new IntWritable(sum));
    }
}
```

```
public static void main(String[] args) throws Exception{
        Configuration conf = new Configuration();
        Job job = new Job(conf,"wordcount");
        job.setJarByClass(WordCount.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(Map.class);
        job.setReducerClass(Reduce.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job,new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));
        job.waitForCompletion(true);
    }
}
```

INPUT FILE



OUTPUT FILE

Contents of directory /dhruv/output1

Goto : /dhruv/output1 go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
_SUCCESS	file	0 KB	1	64 MB	2019-01-12 15:34	rw-r--r--	ponny	supergroup
_logs	dir				2019-01-12 15:33	rwxr-xr-x	ponny	supergroup
part-r-00000	file	0.04 KB	1	64 MB	2019-01-12 15:34	rw-r--r--	ponny	supergroup

Go back to DFS home

File: /dhruv/output1/part-r-00000

Goto : /dhruv/output1 go

Go back to dir listing
Advanced view/download options

```
 bat    2
 cat    1
 fat    1
 mat    2
 rat    1
cat     1
```

LAB EXPERIMENT 4: MAX FREQ WORD, COUNT OF WORDS, SALARY QUESTION (12th, 18th January 2019)

Display maximum frequency word

Code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<LongWritable, Text,
Text, IntWritable>{
		//private final static IntWritable one = new IntWritable(1);
		IntWritable one = new IntWritable(1);
		//private Text word = new Text();


		public void map(LongWritable key, Text value, Context
context) throws IOException, InterruptedException{
			String[] line = value.toString().split(",");
			for(String lines : line){
				context.write(new Text(lines), one);
			}
		}
}

	public static class Reduce extends Reducer <Text,IntWritable,
Text,IntWritable>{
		Text wrd1 = new Text();
		int b = 0;
	   IntWritable res1 = new IntWritable();
		public void reduce(Text key,Iterable<IntWritable> values,
Context context) throws IOException, InterruptedException
		{
		int sum = 0;
		for(IntWritable val : values){
			sum += val.get();
		}
		if(sum>b){
			b=sum;
			wrd1.set(key);
			res1.set(b);
		}
		}

		public void cleanup(Context context) throws IOException,
InterruptedException{
			context.write(wrd1,res1);
		}
	}

	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);
```

```
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job,new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));
        job.waitForCompletion(true);
    }
}
```

Screenshot:

Count the number of words:

Code:

```
import
java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
    public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>{
        //private final static IntWritable one = new IntWritable(1);
        IntWritable one = new IntWritable(1);
        IntWritable count = new IntWritable();
        private Text word2 = new Text("Count: ");
        int counter = 0;

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

            for(String lines : line){
                context.write(new Text(lines), one);
                counter += 1;
            }
        }

        public void cleanup(Context context) throws IOException,InterruptedException{
                count.set(counter);
```

```
                context.write(word2, count);
            }
        }

    public static class Reduce extends Reducer <Text,IntWritable, Text,IntWritable>{
        Text wrd1 = new Text();
        IntWritable res1 = new IntWritable();
        int b = 0;
        public void reduce(Text key,Iterable<IntWritable> values,Context context) throws IOException, InterruptedException
        {
        int sum = 0;
        for(IntWritable val : values){
            sum += val.get();
        }
        if(sum>b){
            b=sum;
            wrd1.set(key);
            res1.set(b);
        }
        }

        public void cleanup(Context context) throws IOException,InterruptedException{
            context.write(wrd1,res1);
        }
    }


    public static void main (String[] args) throws Exception{
        // TODO Auto-generated method stub
        Configuration conf = new Configuration();
        Job job = new Job(conf,"WordCount");
        job.setJarByClass(WordCount.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(Map.class);
        job.setReducerClass(Reduce.class);
        job.setNumReduceTasks(0);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job,new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));
        job.waitForCompletion(true);
    }
}
```

Screenshot:

File: /dhruv/output12/part-m-00000

Goto : /dhruv/output12 go

Go back to dir listing
Advanced view/download options

```
cat     1
mat     1
rat     1
fat     1
cat     1
cat     1
bat     1
rat     1
sat     1
fat     1
cat     1
Count:  11
```

SALARY QUESTION

Code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
    public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>{
        //private final static IntWritable one = new IntWritable(1);
        IntWritable one = new IntWritable(1);
        IntWritable count = new IntWritable();
        private Text word2 = new Text("Number of records: ");
        int counter = 0;

        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
            String[] employee = value.toString().split(",");
            float exp = Float.parseFloat(employee[0]);
            int salary = Integer.parseInt(employee[1]);
            if((exp > 15) && (salary > 60000))
            {
                counter += 1;
                context.write(new Text(employee[3]+" "+exp), new IntWritable(salary));
            }
        }

       public void cleanup(Context context) throws IOException,InterruptedException{
                count.set(counter);
               context.write(word2, count);
          }
        }

    public static class Reduce extends Reducer <Text,IntWritable, Text,IntWritable>{
      Text wrd1 = new Text();
      IntWritable res1 = new IntWritable();
      int b = 0;
        public void reduce(Text key,Iterable<IntWritable> values,Context context) throws IOException, InterruptedException
        {
        int sum = 0;
        for(IntWritable val : values){
            sum += val.get();
        }
        if(sum>b){
            b=sum;
            wrd1.set(key);
            res1.set(b);
        }
        }

        public void cleanup(Context context) throws IOException,InterruptedException{
            context.write(wrd1,res1);
        }
    }


    public static void main (String[] args) throws Exception{
        // TODO Auto-generated method stub
        Configuration conf = new Configuration();
        Job job = new Job(conf,"WordCount");
```

```
        job.setJarByClass(WordCount.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(Map.class);
        job.setReducerClass(Reduce.class);
        job.setNumReduceTasks(0);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job,new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));
        job.waitForCompletion(true);
    }
}
```

File: /dhruv/output14/part-m-00000

Goto : /dhruv/output14 go

Go back to dir listing
Advanced view/download options

```
United Kingdom 29.0     65000
United Kingdom 16.0     210000
Germany 20.0    70000
Germany 17.0    70000
Ireland 20.0    100000
Netherlands 17.0        90000
France 20.0     220000
Denmark 21.0    135380
United Kingdom 18.0     185000
United Kingdom 27.0     181339
Denmark 20.0    75665
Estonia 16.0    65000
Denmark 23.0    84703
Belgium 16.0    102000
Denmark 35.0    140000
Norway 22.0     820000
France 19.0     80000
Ireland 18.0    90000
Sweden 19.0     76600
United Kingdom 25.0     130000
Netherlands 20.0        82000
Switzerland 20.0        113996
Ireland 20.0    80000
United Kingdom 20.0     150000
United Kingdom 20.0     100000
Sweden 20.0     63000
```

```
Norway 22.0     820000
France 19.0     80000
Ireland 18.0    90000
Sweden 19.0     76600
United Kingdom 25.0     130000
Netherlands 20.0        82000
Switzerland 20.0        113996
Ireland 20.0    80000
United Kingdom 20.0     150000
United Kingdom 20.0     100000
Sweden 20.0     63000
United Kingdom 18.0     70915
Sweden 20.0     120000
Ireland 20.0    80000
Finland 20.0    72000
Norway 20.0     75653
Germany 18.0    70000
Norway 25.0     110000
Czech Republic 17.0     63000
Germany 17.0    75000
Czech Republic 21.0     65000
Germany 20.0    120000
Norway 18.0     81000
Ireland 16.0    103000
Number of records:      39
```

LAB EXPERIMENT 5: WORDCOUNT USING COMBINER, PARTITIONING (25th January 2019)

Wordcount using combiner

CODE:

```
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>{
		//private final static IntWritable one = new IntWritable(1);
		IntWritable one = new IntWritable(1);
		//private Text word = new Text();

		public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
			String[] line = value.toString().split(",");
			for(String lines:line){
				context.write(new Text(lines), one);
				}
			}
	}

public static class Reduce extends Reducer <Text,IntWritable, Text,IntWritable> {
	public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException{
		int sum = 0;
		for(IntWritable val : values){
			sum += val.get();
		}
		context.write(key, new IntWritable(sum));
	}
}

public static void main(String[] args) throws Exception{
		Configuration conf = new Configuration();
		Job job = new Job(conf,"wordcount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);
		job.setCombinerClass(Reduce.class);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

File: /dhruv/output16/part-r-00000

Goto : /dhruv/output16 go

Go back to dir listing
Advanced view/download options

```
bat     1
cat     4
fat     2
mat     1
rat     2
sat     1
```

```
19/01/25 15:19:39 INFO util.NativeCodeLoader: Loaded the native-hadoop library
19/01/25 15:19:39 WARN snappy.LoadSnappy: Snappy native library not loaded
19/01/25 15:19:40 INFO mapred.JobClient: Running job: job_201901251506_0004
19/01/25 15:19:41 INFO mapred.JobClient:  map 0% reduce 0%
19/01/25 15:19:55 INFO mapred.JobClient:  map 100% reduce 0%
19/01/25 15:20:10 INFO mapred.JobClient:  map 100% reduce 100%
19/01/25 15:20:15 INFO mapred.JobClient: Job complete: job_201901251506_0004
19/01/25 15:20:15 INFO mapred.JobClient: Counters: 29
19/01/25 15:20:15 INFO mapred.JobClient:   Job Counters
19/01/25 15:20:15 INFO mapred.JobClient:     Launched reduce tasks=1
19/01/25 15:20:15 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=11166
19/01/25 15:20:15 INFO mapred.JobClient:     Total time spent by all reduces wai
ting after reserving slots (ms)=0
19/01/25 15:20:15 INFO mapred.JobClient:     Total time spent by all maps waitin
g after reserving slots (ms)=0
19/01/25 15:20:15 INFO mapred.JobClient:     Launched map tasks=1
19/01/25 15:20:15 INFO mapred.JobClient:     Data-local map tasks=1
19/01/25 15:20:15 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=12658
19/01/25 15:20:15 INFO mapred.JobClient:   File Output Format Counters
19/01/25 15:20:15 INFO mapred.JobClient:     Bytes Written=36
19/01/25 15:20:15 INFO mapred.JobClient:   FileSystemCounters
19/01/25 15:20:15 INFO mapred.JobClient:     FILE_BYTES_READ=66
19/01/25 15:20:15 INFO mapred.JobClient:     HDFS_BYTES_READ=144
19/01/25 15:20:15 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=43367
19/01/25 15:20:15 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=36
19/01/25 15:20:15 INFO mapred.JobClient:   File Input Format Counters
19/01/25 15:20:15 INFO mapred.JobClient:     Bytes Read=44
19/01/25 15:20:15 INFO mapred.JobClient:   Map-Reduce Framework
19/01/25 15:20:15 INFO mapred.JobClient:     Map output materialized bytes=66
19/01/25 15:20:15 INFO mapred.JobClient:     Map input records=1
19/01/25 15:20:15 INFO mapred.JobClient:     Reduce shuffle bytes=66
19/01/25 15:20:15 INFO mapred.JobClient:     Spilled Records=12
19/01/25 15:20:15 INFO mapred.JobClient:     Map output bytes=88
19/01/25 15:20:15 INFO mapred.JobClient:     Total committed heap usage (bytes)=
177016832
19/01/25 15:20:15 INFO mapred.JobClient:     CPU time spent (ms)=690
19/01/25 15:20:15 INFO mapred.JobClient:     Combine input records=11
19/01/25 15:20:15 INFO mapred.JobClient:     SPLIT_RAW_BYTES=100
19/01/25 15:20:15 INFO mapred.JobClient:     Reduce input records=6
19/01/25 15:20:15 INFO mapred.JobClient:     Reduce input groups=6
19/01/25 15:20:15 INFO mapred.JobClient:     Combine output records=6
19/01/25 15:20:15 INFO mapred.JobClient:     Physical memory (bytes) snapshot=18
8874752
19/01/25 15:20:15 INFO mapred.JobClient:     Reduce output records=6
19/01/25 15:20:15 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=767
225856
19/01/25 15:20:15 INFO mapred.JobClient:     Map output records=11
ponny@ubuntu:~$
```

Partitioning

Code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<LongWritable, Text, Text, IntWritable>{
		//private final static IntWritable one = new IntWritable(1);
		IntWritable one = new IntWritable(1);
		IntWritable count = new IntWritable();
		//private Text word2 = new Text("Number of records: ");
		//int counter = 0;

		public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
			String[] employee = value.toString().split(",");
			//float exp = Float.parseFloat(employee[0]);
			int salary = Integer.parseInt(employee[1]);

			context.write(new Text(employee[3]), new IntWritable(salary));
			}
		}

	public static class Reduce extends Reducer <Text,IntWritable, Text,IntWritable>{

		public void reduce(Text key,IntWritable value,Context context) throws IOException, InterruptedException
		{
			context.write(key,value);
		}
	}

	public static class dpart extends Partitioner<Text,IntWritable>{
	    public int getPartition(Text key,IntWritable value,int nr)
		{
			if(value.get()<30000)
				return 0;
			else if(value.get() < 50000)
				return 1;
			else
				return 2;
			 } }


	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);
		job.setPartitionerClass(dpart.class);
		job.setNumReduceTasks(3);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
```

```
        job.waitForCompletion(true);
    }
}
```

File: /dhruv/output18/part-r-00000

Goto : /dhruv/output18 go

Go back to dir listing
Advanced view/download options

```
Austria 28000
Austria 27930
Belgium 24000
Belgium 27000
Belgium 27000
Bulgaria        27600
Bulgaria        28000
Bulgaria        20000
Bulgaria        26000
Croatia 19579
Croatia 21000
Croatia 13800
Croatia 28000
Croatia 17180
Cyprus  11400
Cyprus  18880
Czech Republic  24745
Czech Republic  12450
Czech Republic  22000
Czech Republic  19100
Estonia 29000
Estonia 24000
Finland 24000
Finland 29000
Finland 26400
Finland 27000
```

File: /dhruv/output18/part-r-00001

Goto : /dhruv/output18 go

Go back to dir listing
Advanced view/download options

```
Austria 45000
Austria 48000
Austria 40000
Austria 41000
Austria 39200
Austria 49000
Austria 45500
Austria 39200
Austria 47000
Belgium 37000
Belgium 49000
Belgium 38400
Belgium 45600
Belgium 41000
Belgium 34104
Belgium 41760
Belgium 36600
Belgium 49400
Belgium 30000
Belgium 30000
Bulgaria        46000
Bulgaria        49000
Bulgaria        30000
Bulgaria        42000
Bulgaria        48000
Bulgaria        39000
```

File: /dhruv/output18/part-r-00002

Goto : /dhruv/output18 go

Go back to dir listing
Advanced view/download options

```
Austria 58000
Austria 65000
Austria 57400
Austria 60000
Austria 56000
Austria 52000
Austria 60000
Austria 65000
Austria 60000
Austria 60000
Austria 69000
Austria 57400
Austria 90000
Austria 75000
Belgium 52000
Belgium 72000
Belgium 55000
Belgium 110000
Belgium 102000
Belgium 220000
Belgium 50000
Bulgaria        98000
Bulgaria        57000
Croatia 50000
Czech Republic  75000
Czech Republic  65000
```

LAB EXPERIMENT 6: TOP K RECORDS, COUNTERS (8th February 2019)

PROGRAM 1: TOP K Records (K=10)

Code:

```
import java.io.IOException;
import java.util.TreeMap;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class TopRec{

	public static class Map extends
			Mapper<LongWritable, Text, NullWritable, Text> {
		private TreeMap<Integer, Text> salary = new TreeMap<Integer, Text>();

		public void map(LongWritable key, Text value, Context context)
				throws IOException, InterruptedException {
			String[] employee = value.toString().split(",");
			int salary_emp = Integer.parseInt(employee[1]);
			// float exp_emp = Float.parseFloat(employee[0]);

			salary.put(salary_emp, new Text(value));
			if (salary.size() > 10) {
				salary.remove(salary.firstKey());
			}
		}

		protected void cleanup(Context context) throws IOException,
				InterruptedException {
			for (Text name : salary.values()) {
				context.write(NullWritable.get(), name);
			}
		}
	}

	public static class Reduce extends Reducer<NullWritable, Text, NullWritable, Text> {
		public void reduce(NullWritable key, Iterable<Text> values,Context context) throws IOException,
InterruptedException {
			TreeMap<Integer, Text> salary = new TreeMap<Integer, Text>();
			for (Text value : values) {
				String line = value.toString();
				String[] elements = line.split(",");
				int i = Integer.parseInt(elements[1]);
				salary.put(i, new Text(value));
				if (salary.size() > 10) {
					salary.remove(salary.firstKey());
				}
			}
			for (Text t : salary.values()) {
				context.write(NullWritable.get(), t);
			}
		}
	}

	public static void main(String[] args) throws Exception {
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf, "TopRec");
```

```
                job.setJarByClass(TopRec.class);
                // job.setOutputKeyClass(Text.class);
                // job.setOutputValueClass(IntWritable.class);
                job.setOutputKeyClass(NullWritable.class);
                job.setOutputValueClass(Text.class);
                job.setMapperClass(Map.class);
                job.setReducerClass(Reduce.class);
                job.setNumReduceTasks(1);
                job.setInputFormatClass(TextInputFormat.class);
                job.setOutputFormatClass(TextOutputFormat.class);
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.waitForCompletion(true);
        }
}
```

SCREENSHOT

```
ponny@ubuntu:~$ hadoop jar /home/ponny/TopKRec.jar TopRec /dhruv/salary1.csv /dh
ruv/output24
Warning: $HADOOP_HOME is deprecated.
```

File: /dhruv/output24/part-r-00000

Goto : /dhruv/output24 go

Go back to dir listing
Advanced view/download options

```
15,220000,2,Belgium
7,250000,1,Sweden
10,261546,2,United Kingdom
5,444000,0,Sweden
4,500000,0,United Kingdom
5,550000,0,Germany
6,625000,2,United Kingdom
10,680000,0,Germany
22,820000,0,Norway
12,850000,0,Sweden
```

COUNTERS

Code:

```
import java.io.IOException;

import org.apache.hadoop.conf.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;
import org.apache.hadoop.mapreduce.*;

public class CountRec {
      public enum ct {
            cnt, nt
      };

      public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {
```

```
		IntWritable one = new IntWritable(1);
		IntWritable counter = new IntWritable(0);
		private Text wrd2 = new Text("Num of records: ");
		int count = 0;

		public void map(LongWritable key, Text value, Context context)
				throws IOException, InterruptedException {
			String[] employee = value.toString().split(",");
			String country_emp = employee[3];
			float exp_emp = Float.parseFloat(employee[0]);
			int salary_emp = Integer.parseInt(employee[1]);
			if (exp_emp == 0.0) {
				context.getCounter(ct.cnt).increment(1);
				context.write(new Text(country_emp),
						new IntWritable(salary_emp));
				count++;
			}
			if (salary_emp > 50000) {
				context.getCounter(ct.nt).increment(1);
			}
		}
		public void cleanup(Context context) throws IOException,
InterruptedException{
			counter.set(count);
			context.write(wrd2,  counter);
		}
	}

	public static void main(String[] args) throws Exception {
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf, "CountRec");
		job.setJarByClass(CountRec.class);
		// job.setOutputKeyClass(Text.class);
		// job.setOutputValueClass(IntWritable.class);
		job.setOutputKeyClass(NullWritable.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(Map.class);
		job.setNumReduceTasks(0);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job, new Path(args[0]));
		FileOutputFormat.setOutputPath(job, new Path(args[1]));
		job.waitForCompletion(true);
		Counters cn = job.getCounters();
		cn.findCounter(ct.cnt).getValue();
		cn.findCounter(ct.nt).getValue();
	}
}
```

```
19/02/15 16:08:58 INFO mapred.JobClient:     nt=476
19/02/15 16:08:58 INFO mapred.JobClient:     cnt=26
19/02/15 16:08:58 INFO mapred.JobClient:   FileSystemCounters
19/02/15 16:08:58 INFO mapred.JobClient:     HDFS_BYTES_READ=23839
19/02/15 16:08:58 INFO mapred.JobClient:     FILE_BYTES_WRITTEN=21352
19/02/15 16:08:58 INFO mapred.JobClient:     HDFS_BYTES_WRITTEN=431
19/02/15 16:08:58 INFO mapred.JobClient:   File Input Format Counters
19/02/15 16:08:58 INFO mapred.JobClient:     Bytes Read=23734
19/02/15 16:08:58 INFO mapred.JobClient:   Map-Reduce Framework
19/02/15 16:08:58 INFO mapred.JobClient:     Map input records=1147
19/02/15 16:08:58 INFO mapred.JobClient:     Physical memory (bytes) snapshot=41222144
19/02/15 16:08:58 INFO mapred.JobClient:     Spilled Records=0
19/02/15 16:08:58 INFO mapred.JobClient:     CPU time spent (ms)=130
19/02/15 16:08:58 INFO mapred.JobClient:     Total committed heap usage (bytes)=16252928
19/02/15 16:08:58 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=382939136
19/02/15 16:08:58 INFO mapred.JobClient:     Map output records=27
19/02/15 16:08:58 INFO mapred.JobClient:     SPLIT_RAW_BYTES=105
```

File: /dhruv/output25/part-m-00000

Goto : /dhruv/output25 go

Go back to dir listing
Advanced view/download options

```
Poland  30819
Norway  54022
Switzerland     90000
Germany 31200
United Kingdom  61500
Spain   42000
United Kingdom  34873
Germany 36000
Netherlands     42000
Austria 39200
France  53000
Switzerland     85000
France  45000
Hungary 20000
Slovakia        12000
Finland 36000
Belgium 24000
Finland 36000
United Kingdom  70000
France  20000
United Kingdom  34039
United Kingdom  28334
United Kingdom  101349
Spain   16000
Num of records:         26
```

LAB EXPERIMENT 7: KEY VALUE, NLINE, SEQUENCE FILE OUTPUT, SEQUENCE FILE INPUT (15th February 2019)

PROGRAM 1: KeyValue input format

Code

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<Text, Text, Text, Text>{
		//private final static IntWritable one = new IntWritable(1);
		//IntWritable one = new IntWritable(1);
		//IntWritable count = new IntWritable();
		//private Text word2 = new Text("Number of records: ");
		//int counter = 0;

		String c = "Dhruv";
		public void map(Text key, Text value, Context context) throws IOException, InterruptedException
		{
			String line=key.toString();
			if(c.equalsIgnoreCase(line))
				context.write(key,value);
		}
	}


	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");

		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);
		job.setMapperClass(Map.class);
		job.setNumReduceTasks(0);
		job.setInputFormatClass(KeyValueTextInputFormat.class);
		//job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

File: /dhruv/data.txt

Goto : /dhruv go

Go back to dir listing
Advanced view/download options

```
Dhruv,Mumbai,9969017594
Aakash,Raipur,9284758472
Kunal,Mumbai,8375846759
Aditya,Jodhpur,4857348573
Rachit,Mumbai,4760486638
Arush,Nepal,8564867293
Vishnu,Delhi,8762037853
Chahat,Coimbatore,4857385763
Shantanu,Pune,8674866273
Swaraj,Pune,8574927365
```

```
ponny@ubuntu:~$ hadoop jar /home/ponny/KeyValue.jar WordCount /dhruv/data.txt /d
hruv/output19
Warning: $HADOOP_HOME is deprecated.
```

File: /dhruv/output19/part-m-00000

Goto : /dhruv/output19 go

Go back to dir listing
Advanced view/download options

```
Dhruv	Mumbai,9969017594
```

PROGRAM 2: Nline input format

Code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
```

```
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<LongWritable, Text, LongWritable, Text>{
		//private final static IntWritable one = new IntWritable(1);
		//IntWritable one = new IntWritable(1);
		//IntWritable count = new IntWritable();
		//private Text word2 = new Text("Number of records: ");
		//int counter = 0;


		public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException
		{
			context.write(key,value);
		}
	}

	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		conf.setInt(NLineInputFormat.LINES_PER_MAP, 4);

		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(IntWritable.class);
		job.setMapperClass(Map.class);
		job.setNumReduceTasks(0);
		job.setInputFormatClass(NLineInputFormat.class);
		//job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

Input:

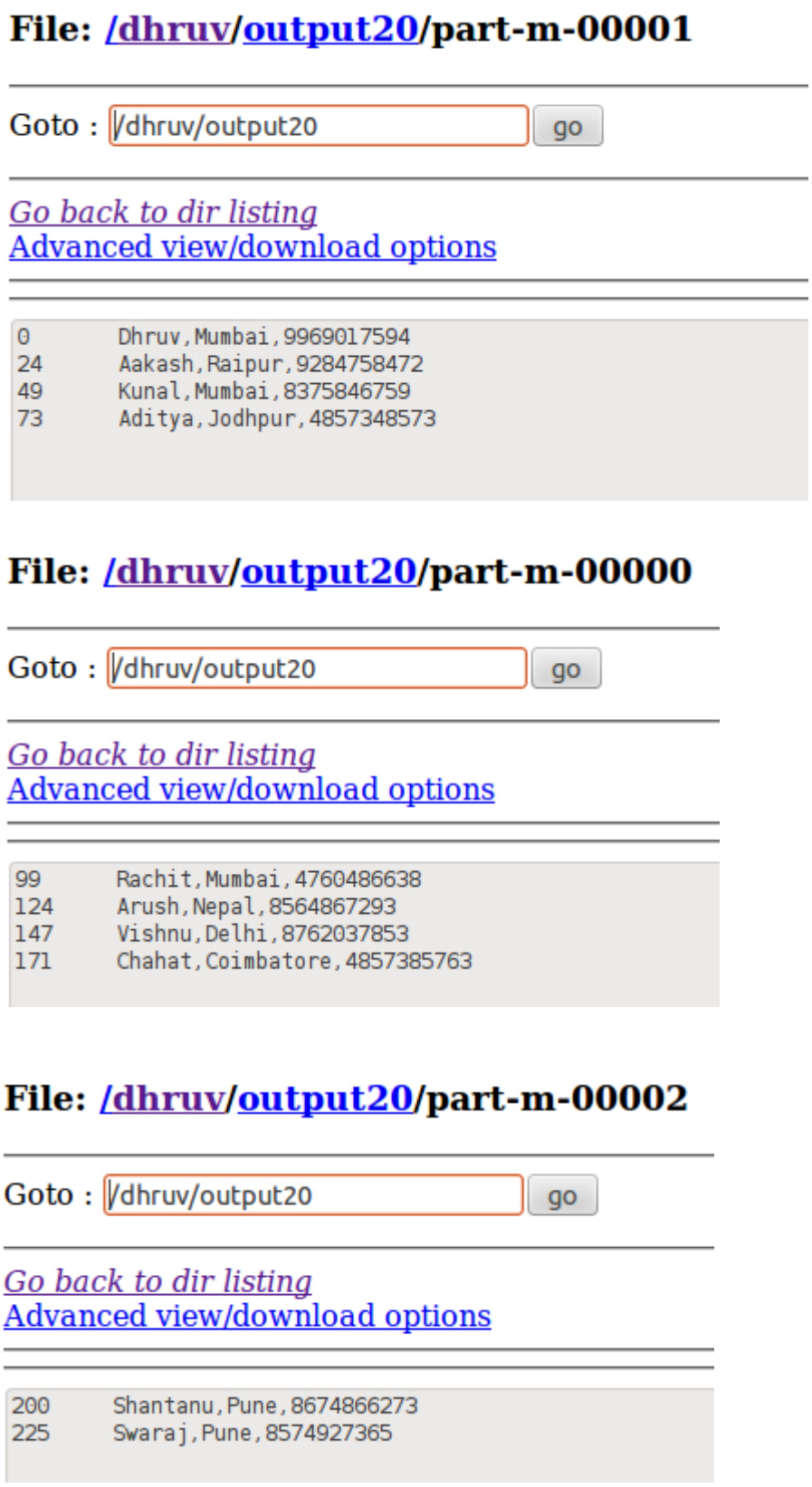
File: /dhruv/data.txt

Goto : /dhruv go

Go back to dir listing
Advanced view/download options

```
Dhruv,Mumbai,9969017594
Aakash,Raipur,9284758472
Kunal,Mumbai,8375846759
Aditya,Jodhpur,4857348573
Rachit,Mumbai,4760486638
Arush,Nepal,8564867293
Vishnu,Delhi,8762037853
Chahat,Coimbatore,4857385763
Shantanu,Pune,8674866273
Swaraj,Pune,8574927365
```

Output:



PROGRAM 3: SequenceFileOutput

Code

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
```

```
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<Text , Text, Text, Text>{
		//private final static IntWritable one = new IntWritable(1);
		//IntWritable one = new IntWritable(1);
		//IntWritable count = new IntWritable();
		//private Text word2 = new Text("Number of records: ");
		//int counter = 0;


		public void map(Text key, Text value, Context context) throws IOException, InterruptedException
		{
			context.write(key,value);
		}
	}



	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		conf.set("mapreduce.input.keyvaluelinerecordreader.key.value.separator",",");

		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(Map.class);
		job.setNumReduceTasks(0);
		job.setInputFormatClass(KeyValueTextInputFormat.class);
		//job.setInputFormatClass(NLineInputFormat.class);
		//job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(SequenceFileOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

File: /dhruv/output22/part-m-00000

Goto : /dhruv/output22 go

Go back to dir listing
Advanced view/download options

```
SEQorg.apache.hadoop.io.Textorg.apache.hadoop.io.Text▯▯▯▯▯▯M▯U▯▯j▯▯g▯▯▯▯▯▯▯▯Dhruv
Mumbai,9969017594▯▯▯▯▯▯AakashRaipur,9284758472▯▯▯▯▯▯KunalMumbai,8375846759▯▯▯▯▯▯Aditya
Jodhpur,4857348573▯▯▯▯▯▯RachitMumbai,4760486638▯▯▯▯▯▯ArushNepal,8564867293▯▯▯▯▯▯Vishnu
Delhi,8762037853▯▯▯▯▯▯ChahatCoimbatore,4857385763▯▯▯▯▯▯ ShantanuPune,8674866273▯▯▯▯▯▯
SwarajPune,8574927365
```

PROGRAM 4: SequnceFileInput to Text output

Code:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
	public static class Map extends Mapper<Text , Text, Text, Text>{
		//private final static IntWritable one = new IntWritable(1);
		//IntWritable one = new IntWritable(1);
		//IntWritable count = new IntWritable();
		//private Text word2 = new Text("Number of records: ");
		//int counter = 0;


		public void map(Text key, Text value, Context context) throws IOException, InterruptedException
		{
			context.write(key,value);
		}
	}



	public static void main (String[] args) throws Exception{
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf,"WordCount");
		job.setJarByClass(WordCount.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(Map.class);
		job.setNumReduceTasks(0);
		job.setInputFormatClass(SequenceFileInputFormat.class);
		//job.setInputFormatClass(NLineInputFormat.class);
		//job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job,new Path(args[0]));
		FileOutputFormat.setOutputPath(job,new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

GIVE THE SequenceFile as input

```
ponny@ubuntu:~$ hadoop jar /home/ponny/SequenceInput.jar WordCount /dhruv/output
22/part-m-00000 /dhruv/output23
Warning: $HADOOP_HOME is deprecated.
```

Input:

File: /dhruv/output22/part-m-00000

Goto : /dhruv/output22 go

Go back to dir listing
Advanced view/download options

```
SEQorg.apache.hadoop.io.Textorg.apache.hadoop.io.Text[illegible]M@U@@j@@g[illegible]Dhruv
Mumbai,9969017594[illegible]AakashRaipur,9284758472[illegible]KunalMumbai,8375846759[illegible]Aditya
Jodhpur,4857348573[illegible]RachitMumbai,4760486638[illegible]ArushNepal,8564867293[illegible]Vishnu
Delhi,8762037853[illegible]ChahatCoimbatore,4857385763[illegible] ShantanuPune,8674866273[illegible]
SwarajPune,8574927365
```

Output:

File: /dhruv/output23/part-m-00000

Goto : /dhruv/output23 go

Go back to dir listing
Advanced view/download options

```
Dhruv   Mumbai,9969017594
Aakash  Raipur,9284758472
Kunal   Mumbai,8375846759
Aditya  Jodhpur,4857348573
Rachit  Mumbai,4760486638
Arush   Nepal,8564867293
Vishnu  Delhi,8762037853
Chahat  Coimbatore,4857385763
Shantanu        Pune,8674866273
Swaraj  Pune,8574927365
```

LAB EXPERIMENT 8: SIDE DATA, REDUCE SIDE JOIN (1st March 19)

PROGRAM 1: SIDE DATA

CODE:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
    public static class Map extends Mapper<LongWritable, Text, Text, Text>{

        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
            String name1;
            name1 = context.getConfiguration().get("name");
            String[] mobile = value.toString().split(",");
            String name = mobile[1];
            if(name1.equals(name)){
                context.write(new Text(mobile[0]), new Text(mobile[1] +","+mobile[2]));
            }
        }
}

    public static void main (String[] args) throws Exception{
        // TODO Auto-generated method stub
        Configuration conf = new Configuration();
        conf.set("name",args[2]);
        Job job = new Job(conf,"WordCount");
        job.setJarByClass(WordCount.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(Map.class);
        job.setNumReduceTasks(0);
        job.setInputFormatClass(TextInputFormat.class);
```

```
        job.setOutputFormatClass(TextOutputFormat.class);
        FileInputFormat.addInputPath(job,new Path(args[0]));
        FileOutputFormat.setOutputPath(job,new Path(args[1]));
        job.waitForCompletion(true);
    }
}
```

OUTPUT:

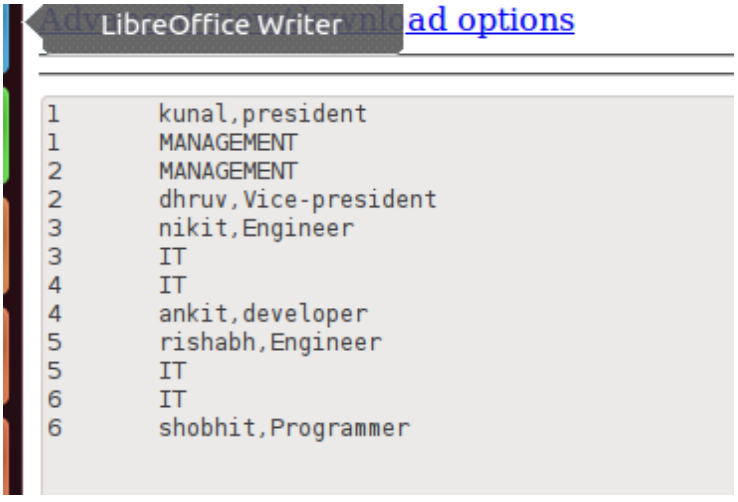
PROGRAM 2: Reduce side join

CODE:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class WordCount{
    public static class empmapper extends Mapper<LongWritable, Text, IntWritable, Text>{
        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
            String[] emp = value.toString().split(",");
            int id = Integer.parseInt(emp[0]);
            context.write(new IntWritable(id),new Text(emp[1]+","+emp[2]));
        } }
```

```
    public static class depmapper extends Mapper<LongWritable, Text, IntWritable, Text>{
       public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException{
         String[] dept = value.toString().split(",");
          int id = Integer.parseInt(dept[0]);
          context.write(new IntWritable(id),new Text(dept[1]));
       }
}
    public static class Reduce extends Reducer <IntWritable, Text,IntWritable, Text>{
        public void reduce(IntWritable key,Text values, Context context) throws IOException, InterruptedException{
                context.write(key, values);
        }}
    public static void main (String[] args) throws Exception{
        // TODO Auto-generated method stub
        Configuration conf = new Configuration();
        Job job = new Job(conf,"WordCount");
        job.setJarByClass(WordCount.class);
        job.setOutputKeyClass(IntWritable.class);
        job.setOutputValueClass(Text.class);
        job.setReducerClass(Reduce.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        MultipleInputs.addInputPath(job,new Path(args[0]),TextInputFormat.class,  empmapper.class);
        MultipleInputs.addInputPath(job,new Path(args[1]),TextInputFormat.class, depmapper.class);
        FileOutputFormat.setOutputPath(job,new Path(args[2]));
        job.waitForCompletion(true);
    }
}
```

OUTPUT:



LAB EXPERIMENT 9: MAP SIDE JOIN AND DISTRIBUTED CACHE (15th March 19)

MAP SIDE JOIN

Input files

deptDetails

```
1,CSE
2,CSE
8,ECE
3,MECH
9,CIVIL
4,CSE
5,CSE
10,ECE
6,ECE
7,CIVIL
```

studDetails

```
1,Dhruv,9.3
2,Kunal,9.1
3,Arush,8.6
4,Rishabh,8.9
5,Aakash,8.5
6,Aditya,8.5
7,Chahat,8.5
8,Raagul,8.7
9,Shantanu,9.4
10,Vishnu,8.1
```

CODE:

```
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;
import org.apache.hadoop.filecache.*;
import java.io.*;

public class DistCache {
	public static class Map extends Mapper<LongWritable, Text, Text, Text> {

		Path[] cfile = new Path[0];
		ArrayList<Text> dep = new ArrayList<Text>();

		public void setup(Context context) {
			Configuration conf = context.getConfiguration();
			try {
				cfile = DistributedCache.getLocalCacheFiles(conf);
				BufferedReader reader = new BufferedReader(new FileReader(
						cfile[0].toString()));
				String line;
				while ((line = reader.readLine()) != null) {
					Text tt = new Text(line);
					dep.add(tt);
				}
			}
			catch (IOException e) {
				e.printStackTrace();
			}
		}

		public void map(LongWritable key, Text value, Context context)
				throws IOException, InterruptedException {
			String line2 = value.toString();
			String[] elements = line2.split(",");

			for (Text e : dep) {
				String[] line1 = e.toString().split(",");
				if (elements[0].equals(line1[0])) {
					context.write(new Text(elements[0]), new Text(elements[1]
```

```
                                                    + "," + elements[2] + "," + line1[1]));
                        }
                    }
                }
        }

        public static void main(String[] args) throws Exception {

                Configuration conf = new Configuration();
                Job job = new Job(conf, "wordcount");
                job.setJarByClass(DistCache.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);
                job.setMapperClass(Map.class);
                job.setNumReduceTasks(0);
                DistributedCache.addCacheFile(new Path(args[0]).toUri(),job.getConfiguration());
                job.setInputFormatClass(TextInputFormat.class);
                FileInputFormat.addInputPath(job, new Path(args[1]));
                FileOutputFormat.setOutputPath(job, new Path(args[2]));
                job.waitForCompletion(true);

        }

}
```

OUTPUT:

File: /dhruv/distCache_output/part-m-00000

Goto : /dhruv/distCache_output go

Go back to dir listing
Advanced view/download options

```
1       Dhruv,9.3,CSE
2       Kunal,9.1,CSE
3       Arush,8.6,MECH
4       Rishabh,8.9,CSE
5       Aakash,8.5,CSE
6       Aditya,8.5,ECE
7       Chahat,8.5,CIVIL
8       Raagul,8.7,ECE
9       Shantanu,9.4,CIVIL
10      Vishnu,8.1,ECE
```

PROGRAM 2: Reduce side join

Input:

custDetails

```
1,Dhruv
2,Kunal
3,Arush
4,Vishnu
5,Aakash
6,Aditya
7,Chahat
```

transacDetails

```
1,372,card
5,1000,card
2,945,card
4,2353,card
5,400,card
3,912,card
2,583,cash
7,114,cash
1,26,cash
6,867,cash
```

CODE:

```
import java.io.IOException;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;
//import org.apache.hadoop.filecache.*;
//import java.io.*;

public class ReduceJoin {
        public static class custmapper extends
                        Mapper<LongWritable, Text, Text, Text> {

                public void map(LongWritable key, Text value, Context context)
                                throws IOException, InterruptedException {
                        String text = value.toString();
                        String[] line = text.split(",");
                        context.write(new Text(line[0]), new Text("cust" + "," + line[1]));
                }
        }

        public static class transmapper extends
                        Mapper<LongWritable, Text, Text, Text> {

                public void map(LongWritable key, Text value, Context context)
                                throws IOException, InterruptedException {
                        String text = value.toString();
                        String[] line = text.split(",");
                        context.write(new Text(line[0]), new Text("trans" + "," + line[1]+","+line[2]));
                }
        }

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

                String st1;

                public void reduce(Text key, Iterable<Text> values, Context context )
          throws IOException, InterruptedException
          {
                    int c=0,amt=0;
          for(Text val:values)
           {
```

```
                String[] line = val.toString().split(",");

                if (line[0].equals("trans"))
                {
                        if(line[2].equals("card")){
                                amt += Integer.parseInt(line[1]);
                                c+=1;
                        }

                }
                else if (line[0].equals("cust"))
                {
                        st1 = line[1];
                                }
          }
                                context.write(new Text(st1), new Text(c+","+amt));
                }
        }

        public static void main(String[] args) throws Exception {

                Configuration conf = new Configuration();
                Job job = new Job(conf, "wordcount");
                job.setJarByClass(ReduceJoin.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(Text.class);
                job.setReducerClass(jreducer.class);
                job.setInputFormatClass(TextInputFormat.class);
                MultipleInputs.addInputPath(job, new Path(args[0]), TextInputFormat.class, custmapper.class);
                MultipleInputs.addInputPath(job, new Path(args[1]), TextInputFormat.class, transmapper.class);
                FileOutputFormat.setOutputPath(job, new Path(args[2]));
                job.waitForCompletion(true);
        }

}
```

OUTPUT:

File: /dhruv/reduceJoin_output/part-r-00000

Goto : /dhruv/reduceJoin_output go

Go back to dir listing
Advanced view/download options

```
Dhruv   1,372
Kunal   1,945
Arush   1,912
Vishnu  1,2353
Aakash  2,1400
Aditya  0,0
Chahat  0,0
```

LAB EXPERIMENT 10: BASELINE INVERTED INDEX (22nd March 19)

Code:

```
import java.io.IOException;
import java.util.HashMap;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class BaselineIndex {
	public static class Map extends Mapper<LongWritable, Text, Text, Text> {
		// private final static IntWritable one = new IntWritable(1);

		public void map(LongWritable key, Text value, Context context)
				throws IOException, InterruptedException {
			String fileName = ((FileSplit) context.getInputSplit()).getPath()
					.getName();
			String[] words = value.toString().split(",");
			for (String s : words) {
				context.write(new Text(s), new Text(fileName));
			}
		}
	}

	public static class Reduce extends Reducer<Text, Text, Text, Text> {
		public void reduce(Text key, Iterable<Text> values, Context context)
				throws IOException, InterruptedException {
			HashMap m = new HashMap();
			int count = 0;
			for (Text t : values) {
				String str = t.toString();
				if (m != null && m.get(str) != null) {
					count = (int) m.get(str);
					m.put(str, ++count);
				} else {
					m.put(str, 1);
				}
			}
			context.write(key, new Text(m.toString()));
		}
	}

	public static void main(String[] args) throws Exception {
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf, "BaselineIndex");
		job.setJarByClass(BaselineIndex.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);
		//job.setNumReduceTasks(0);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
		FileInputFormat.addInputPath(job, new Path(args[0]));
		FileOutputFormat.setOutputPath(job, new Path(args[1]));
		job.waitForCompletion(true);
	}
}
```

```
ponny@ubuntu:~$ hadoop jar /home/ponny/baselineindex.jar BaselineIndex /dhruv/in
vInput /dhruv/invIndexOutput
Warning: $HADOOP_HOME is deprecated.
```

INPUT:

File: /dhruv/invInput/text1

Goto : /dhruv/invInput go

Go back to dir listing
Advanced view/download options

```
cat,bat,rat,mat,fat,sat,hello
world,hollow,pillow,hello
pillow,bat,rat,fat,willow
fellow
```

File: /dhruv/invInput/text2

Goto : /dhruv/invInput go

Go back to dir listing
Advanced view/download options

```
ball,bat,rat,yoga,reduce,weight,trell
global,world,hello,hollow,pillow,hello
pillow,cat,rat,fat,bat,rat
fat,world,willow
fellow,intern
```

File: /dhruv/invIndexOutput/part-r-00000

Goto : /dhruv/invIndexOutput go

Go back to dir listing
Advanced view/download options

```
ball    {text2=1}
bat     {text1=2, text2=2}
cat     {text1=1, text2=1}
fat     {text1=2, text2=2}
fellow  {text1=1, text2=1}
global  {text2=1}
hello   {text1=2, text2=2}
hollow  {text1=1, text2=1}
intern  {text2=1}
mat     {text1=1}
pillow  {text1=2, text2=2}
rat     {text1=2, text2=3}
reduce  {text2=1}
sat     {text1=1}
trell   {text2=1}
weight  {text2=1}
willow  {text1=1, text2=1}
world   {text1=1, text2=2}
yoga    {text2=1}
```

LAB EXPERIMENT 11: POSITION OF WORD IN A FILE (29th March 19)

CODE:

```
import java.io.IOException;
import java.util.HashMap;
//import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.*;
import org.apache.hadoop.mapreduce.lib.output.*;

public class BaselineIndex {

	public static class Map extends Mapper<LongWritable, Text, Text, Text> {
		// private final static IntWritable one = new IntWritable(1);
		int k = 0;
		public void map(LongWritable key, Text value, Context context)
				throws IOException, InterruptedException {
			String fileName = ((FileSplit) context.getInputSplit()).getPath()
					.getName();
			String[] words = value.toString().split(",");

			for (String s : words) {
				context.write(new Text(s), new Text(fileName +" "+ k));
				k++;
			}
		}
	}

	public static class Reduce extends Reducer<Text, Text, Text, Text> {
		public void reduce(Text key, Iterable<Text> values, Context context)
				throws IOException, InterruptedException {
			//HashMap m = new HashMap();
			int count = 0;
			for (Text t : values) {
				/*String str = t.toString();
				if (m != null && m.get(str) != null) {
					count = (int) m.get(str);
					m.put(str, ++count);
				} else {
					m.put(str, 1);
				}*/
				context.write(key, t);
			}
			//context.write(key, new Text(m.toString()));
		}
	}

	public static void main(String[] args) throws Exception {
		// TODO Auto-generated method stub
		Configuration conf = new Configuration();
		Job job = new Job(conf, "BaselineIndex");
		job.setJarByClass(BaselineIndex.class);
		job.setOutputKeyClass(Text.class);
		job.setOutputValueClass(Text.class);
		job.setMapperClass(Map.class);
		job.setReducerClass(Reduce.class);

		//job.setNumReduceTasks(0);
		job.setInputFormatClass(TextInputFormat.class);
		job.setOutputFormatClass(TextOutputFormat.class);
```

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

SCREENSHOT

```
ponny@ubuntu:~$ hadoop jar /home/ponny/findpos.jar BaselineIndex /dhruv/invInput
 /dhruv/findPosOutputNewest2
Warning: $HADOOP_HOME is deprecated.

19/03/29 15:34:03 WARN mapred.JobClient: Use GenericOptionsParser for parsing th
e arguments. Applications should implement Tool for the same.
19/03/29 15:34:03 INFO input.FileInputFormat: Total input paths to process : 2
19/03/29 15:34:03 INFO util.NativeCodeLoader: Loaded the native-hadoop library
19/03/29 15:34:03 WARN snappy.LoadSnappy: Snappy native library not loaded
19/03/29 15:34:03 INFO mapred.JobClient: Running job: job_201903291516_0007
19/03/29 15:34:04 INFO mapred.JobClient:  map 0% reduce 0%
19/03/29 15:34:29 INFO mapred.JobClient:  map 50% reduce 0%
19/03/29 15:34:32 INFO mapred.JobClient:  map 100% reduce 0%
19/03/29 15:34:44 INFO mapred.JobClient:  map 100% reduce 100%
19/03/29 15:34:49 INFO mapred.JobClient: Job complete: job_201903291516_0007
19/03/29 15:34:49 INFO mapred.JobClient: Counters: 29
19/03/29 15:34:49 INFO mapred.JobClient:   Job Counters
19/03/29 15:34:49 INFO mapred.JobClient:     Launched reduce tasks=1
19/03/29 15:34:49 INFO mapred.JobClient:     SLOTS_MILLIS_MAPS=14436
19/03/29 15:34:49 INFO mapred.JobClient:     Total time spent by all reduces wai
ting after reserving slots (ms)=0
19/03/29 15:34:49 INFO mapred.JobClient:     Total time spent by all maps waitin
g after reserving slots (ms)=0
19/03/29 15:34:49 INFO mapred.JobClient:     Launched map tasks=2
19/03/29 15:34:49 INFO mapred.JobClient:     Data-local map tasks=2
19/03/29 15:34:49 INFO mapred.JobClient:     SLOTS_MILLIS_REDUCES=12657
```

File: /dhruv/findPosOutputNewest2/part-r-00000

Goto : /dhruv/findPosOutputNewest2 go

Go back to dir listing
Advanced view/download options

```
ball    text2 0
bat     text2 1
bat     text2 17
bat     text1 1
bat     text1 12
cat     text1 0
cat     text2 14
fat     text2 19
fat     text2 16
fat     text1 14
fat     text1 4
fellow  text1 16
fellow  text2 22
global  text2 7
hello   text2 12
hello   text2 9
hello   text1 6
hello   text1 10
hollow  text1 8
hollow  text2 10
intern  text2 23
mat     text1 3
pillow  text1 9
pillow  text1 11
pillow  text2 13
pillow  text2 11
```