Experiment No.4
Experiment on Hadoop Map-Reduce
Date of Performance: 14/08/23
Date of Submission: 21/08/23

AIM: -To write a program to implement a word count program using MapReduce.

THEORY:

WordCount is a simple program which counts the number of occurrences of each word in a given text input data set. WordCount fits very well with the MapReduce programming model making it a great example to understand the Hadoop Map/Reduce programming style. The implementation consists of three main parts:

1. Mapper

2. Reducer

3. Driver

Step-1. Write a Mapper

A Mapper overrides the ―map‖ function from the Class "org.apache.hadoop.mapreduce.Mapper" which provides <key, value> pairs as the input. A Mapper implementation may output <key,value> pairs using the provided Context .

Input value of the WordCount Map task will be a line of text from the input data file and the key would be the line number <line_number, line_of_text> . Map task outputs <word, one> for each word in the line of text. Pseudo-code void Map (key, value){ for each word x in value:

output.collect(x,1);

}

Step-2. Write a Reducer

A Reducer collects the intermediate <key,value> output from multiple map tasks and assemble a single result. Here, the WordCount program will sum up the occurrence of each word to pairs as <word, occurrence>. Pseudo-code

void Reduce (keyword, <list of value>){ for each

x in <list of value>:

sum+=x;
final_output.collect(keyword, sum);

}

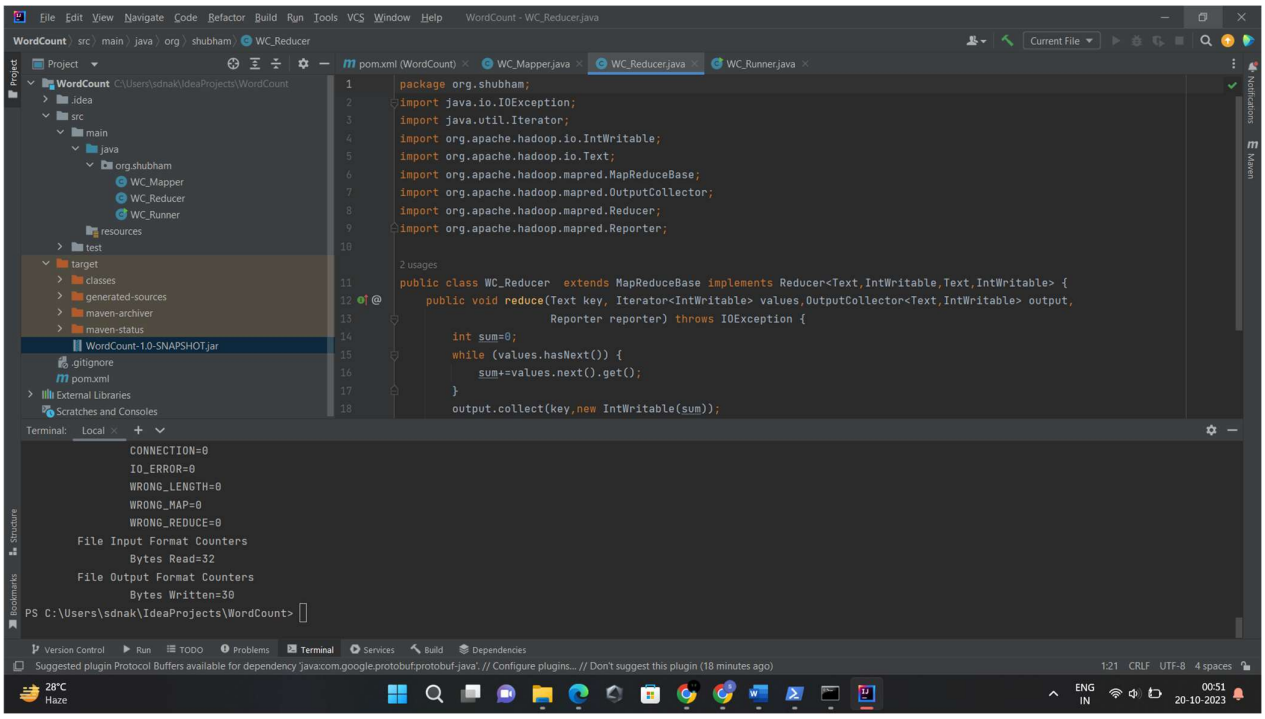
Code:

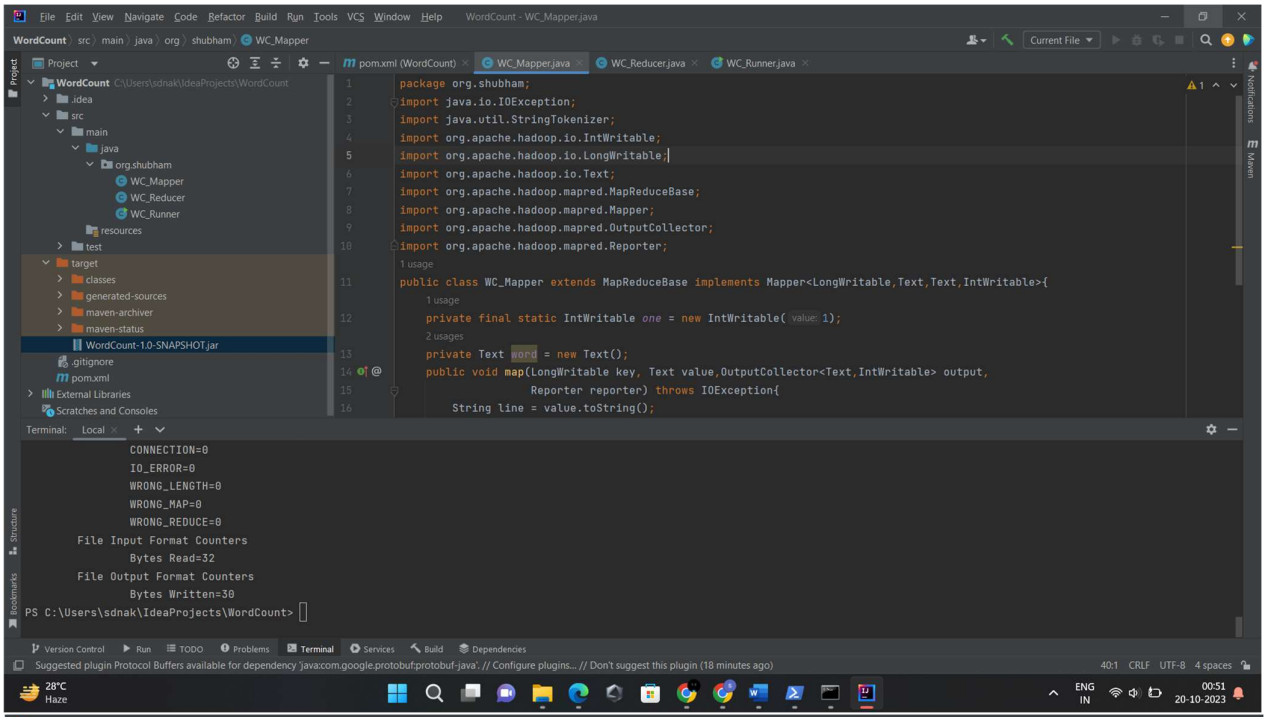
```
import java.io.IOException; import
java.util.StringTokenizer; import
org.apache.hadoop.io.IntWritable; import
org.apache.hadoop.io.LongWritable; import
org.apache.hadoop.io.Text; import
org.apache.hadoop.mapreduce.Mapper; import
org.apache.hadoop.mapreduce.Reducer; import
org.apache.hadoop.conf.Configuration; import
org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.fs.Path; public class WordCount
{
public static class Map extends Mapper<LongWritable,Text,Text,IntWritable> { public
void map(LongWritable key, Text value,Context context) throws
IOException,InterruptedException{
String line = value.toString();
StringTokenizer tokenizer = new StringTokenizer(line);
while (tokenizer.hasMoreTokens()) {
value.set(tokenizer.nextToken()); context.write(value,
new IntWritable(1));
}
```

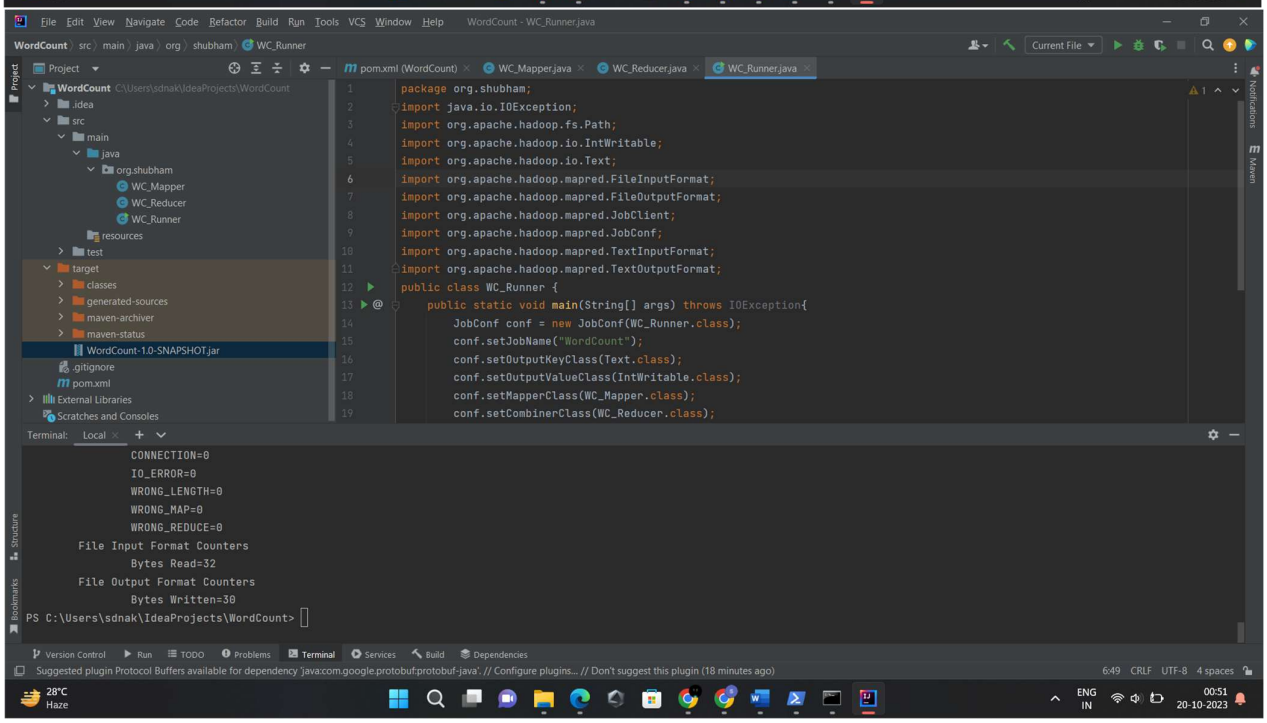
```
} }
public static class Reduce extends Reducer<Text,IntWritable,Text,IntWritable> {
public void reduce(Text key, Iterable<IntWritable> values,Context context)
throws IOException,InterruptedException { int sum=0; for(IntWritable x: values)
{ sum+=x.get();
}
context.write(key, new IntWritable(sum));
} }
public static void main(String[] args) throws Exception {
Configuration conf= new Configuration(); Job job = new
Job(conf,"My Word Count Program");
job.setJarByClass(WordCount.class);
job.setMapperClass(Map.class);
job.setReducerClass(Reduce.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
job.setInputFormatClass(TextInputFormat.class);
job.setOutputFormatClass(TextOutputFormat.class);
Path outputPath = new Path(args[1]);
//Configuring the input/output path from the filesystem into the job
FileInputFormat.addInputPath(job, new Path(args[0])); FileOutputFormat.setOutputPath(job,
new Path(args[1]));
//deleting the output path automatically from hdfs so that we don't have to delete
it explicitly
outputPath.getFileSystem(conf).delete(outputPath);  //exiting
the job only if the flag value becomes false
```

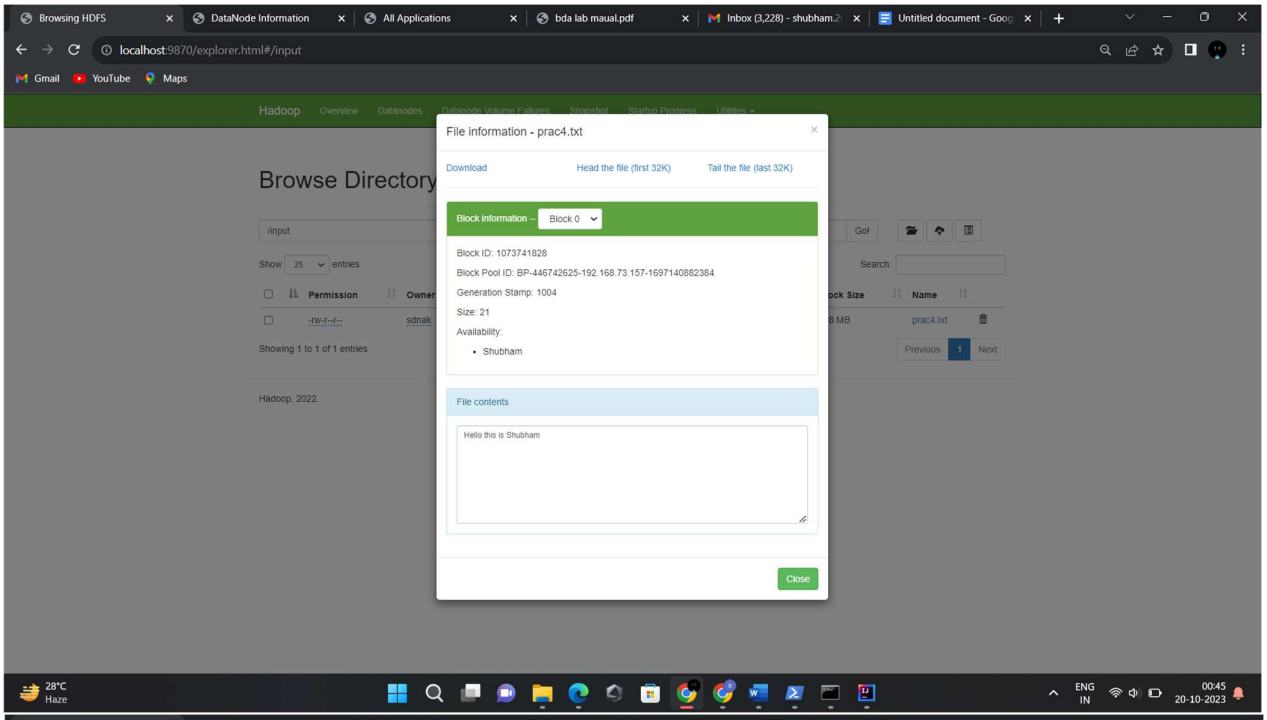
```
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

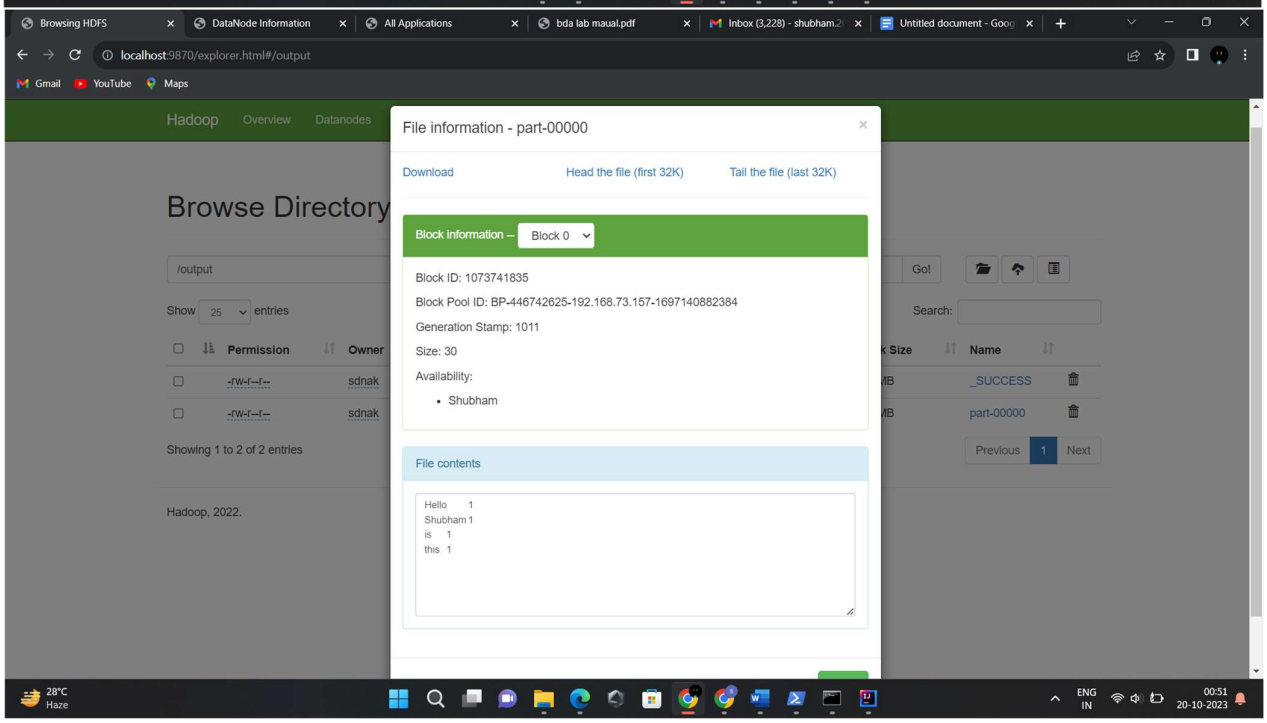
OUTPUT:




File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
WordCount - WC_Mapper.java
WordCount > src > main > java > org > shubham > WC_Mapper
Current File
Project
WordCount C:\Users\sdnak\IdeaProjects\WordCount
.idea
src
main
java
org.shubham
WC_Mapper
WC_Reducer
WC_Runner
resources
test
target
classes
generated-sources
maven-archiver
maven-status
WordCount-1.0-SNAPSHOT.jar
.gitignore
pom.xml
External Libraries
Scratches and Consoles
pom.xml (WordCount)
WC_Mapper.java
WC_Reducer.java
WC_Runner.java
package org.shubham;
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.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
1 usage
public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
1 usage
private final static IntWritable one = new IntWritable(value: 1);
2 usages
private Text word = new Text();
public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
Reporter reporter) throws IOException{
String line = value.toString();
Terminal: Local
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=32
File Output Format Counters
Bytes Written=30
PS C:\Users\sdnak\IdeaProjects\WordCount>
Version Control
Run
TODO
Problems
Terminal
Services
Build
Dependencies
Suggested plugin Protocol Buffers available for dependency 'java:com.google.protobuf:protobuf-java'. // Configure plugins... // Don't suggest this plugin (18 minutes ago)
40:1
CRLF
UTF-8
4 spaces
28°C
Haze
ENG
IN
00:51
20-10-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
WordCount - WC_Runner.java
WordCount > src > main > java > org > shubham > WC_Runner
Current File
Project
WordCount C:\Users\sdnak\IdeaProjects\WordCount
.idea
src
main
java
org.shubham
WC_Mapper
WC_Reducer
WC_Runner
resources
test
target
classes
generated-sources
maven-archiver
maven-status
WordCount-1.0-SNAPSHOT.jar
.gitignore
pom.xml
External Libraries
Scratches and Consoles
pom.xml (WordCount)
WC_Mapper.java
WC_Reducer.java
WC_Runner.java
package org.shubham;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
import org.apache.hadoop.mapred.TextInputFormat;
import org.apache.hadoop.mapred.TextOutputFormat;
public class WC_Runner {
public static void main(String[] args) throws IOException{
JobConf conf = new JobConf(WC_Runner.class);
conf.setJobName("WordCount");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(WC_Mapper.class);
conf.setCombinerClass(WC_Reducer.class);
Terminal: Local
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
Bytes Read=32
File Output Format Counters
Bytes Written=30
PS C:\Users\sdnak\IdeaProjects\WordCount>
Version Control
Run
TODO
Problems
Terminal
Services
Build
Dependencies
Suggested plugin Protocol Buffers available for dependency 'java:com.google.protobuf:protobuf-java'. // Configure plugins... // Don't suggest this plugin (18 minutes ago)
6:49
CRLF
UTF-8
4 spaces
28°C
Haze
ENG
IN
00:51
20-10-2023


Browsing HDFS
DataNode Information
All Applications
bda lab maual.pdf
Inbox (3,228) - shubham.2
Untitled document - Goog
localhost:9870/explorer.html#/input
Gmail
YouTube
Maps
Hadoop
Overview
Datanodes
Browse Directory
/input
Go!
Search:
Show 25 entries
Permission
Owner
Name
-rw-r--r--
sdnak
prac4.txt
Showing 1 to 1 of 1 entries
Previous
1
Next
Hadoop, 2022.
File information - prac4.txt
Download
Head the file (first 32K)
Tail the file (last 32K)
Block information --
Block 0
Block ID: 1073741828
Block Pool ID: BP-446742625-192.168.73.157-1697140882384
Generation Stamp: 1004
Size: 21
Availability:
Shubham
File contents
Hello this is Shubham
Close
28°C
Haze
ENG
IN
00:45
20-10-2023


Browsing HDFS
DataNode Information
All Applications
bda lab maual.pdf
Inbox (3,228) - shubham.2
Untitled document - Goog
localhost:9870/explorer.html#/output
Gmail
YouTube
Maps
Hadoop
Overview
Datanodes
Browse Directory
/output
Go!
Search:
Show 25 entries
Permission
Owner
Name
-rw-r--r--
sdnak
_SUCCESS
part-00000
Showing 1 to 2 of 2 entries
Previous
1
Next
Hadoop, 2022.
File information - part-00000
Download
Head the file (first 32K)
Tail the file (last 32K)
Block information --
Block 0
Block ID: 1073741835
Block Pool ID: BP-446742625-192.168.73.157-1697140882384
Generation Stamp: 1011
Size: 30
Availability:
Shubham
File contents
Hello 1
Shubham 1
is 1
this 1
28°C
Haze
ENG
IN
00:51
20-10-2023

CONCLUSION:

The experiment aimed at implementing a word count program using MapReduce yielded successful results. MapReduce demonstrated its ability to scale and efficiently process extensive datasets by distributing tasks across multiple nodes for parallel execution. It also showcased its fault tolerance, ensuring the integrity of data processing in distributed systems. MapReduce's simplicity, characterized by its straightforward mapper and reducer functions, makes it accessible to a broad spectrum of developers. The practical relevance of this experiment extends to more intricate data processing tasks, such as log analysis and machine learning. By incorporating performance optimizations like combiners and partitioners, the program's efficiency can be further improved. In essence, this experiment established a robust foundation in distributed computing, a highly valuable skill in today's data-driven landscape.