BUNTS SANGHA MUMBAI

ANNA LEELA COLLEGE OF COMMERCE & ECONOMICS

SHOBHA JAYARAM SHETTY COLLEGE FOR BMS

(UNIVERSITY OF MUMBAI)

Shashi Manmohan Shetty Higher Education Complex,

Buntara Bhavan Marg, Kurla (E), Mumbai – 400 070.

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Exam. Seat No. _____

Date _____

CERTIFIED that the practical, and assignments duly signed, were performed by

Mr./Ms. _____

Roll No. _____ *of* _____*class in the Information Technology Laboratory of Anna Leela College of Commerce & Economics Shobha Jayaram Shetty College for BMS, Mumbai during the academic year 20* _____ *-20* _____

He/she has completed the course of laboratory assignments in Information Technology as contained in the course prescribed by the University of Mumbai.

Head of Dept.

Information Technology

Date _____

Sign. of the student

Date _____

Professor-in-charge

1) _____

Date _____

Sign. of Examiner's

1) _____

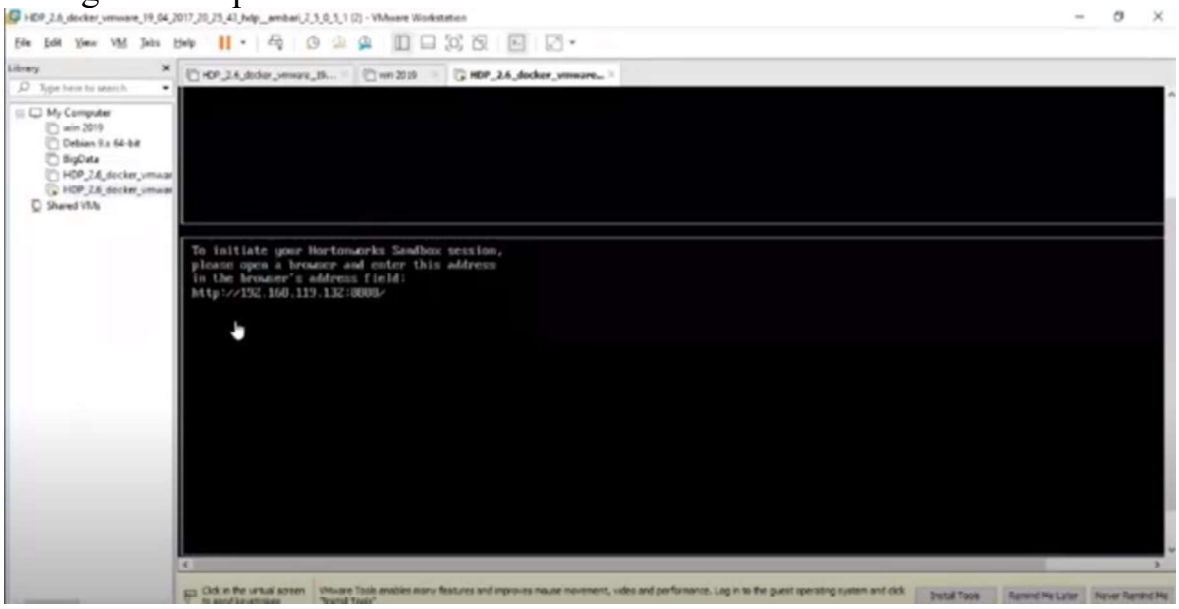
Date _____

Sr. No	PRACTICAL	DATE	SIGN
1	Install, configure, and run Hadoop and HDFS, and explore HDFS.		
2	Implement word count/frequency programs using MapReduce		
3	Implement a MapReduce program that processes a weather dataset.		
4	Implement the program using Pig.		
5	Implement the application in Hive.		
6	Implement an application that stores big data in HBase/ Python		
7	Implement Decision tree classification techniques		
8	Implement SVM classification techniques		
9	To implement the REGRESSION MODEL (Linear & Logistical Regression)		
10	To implement the Classification Model		
11	To implement the Clustering Model		

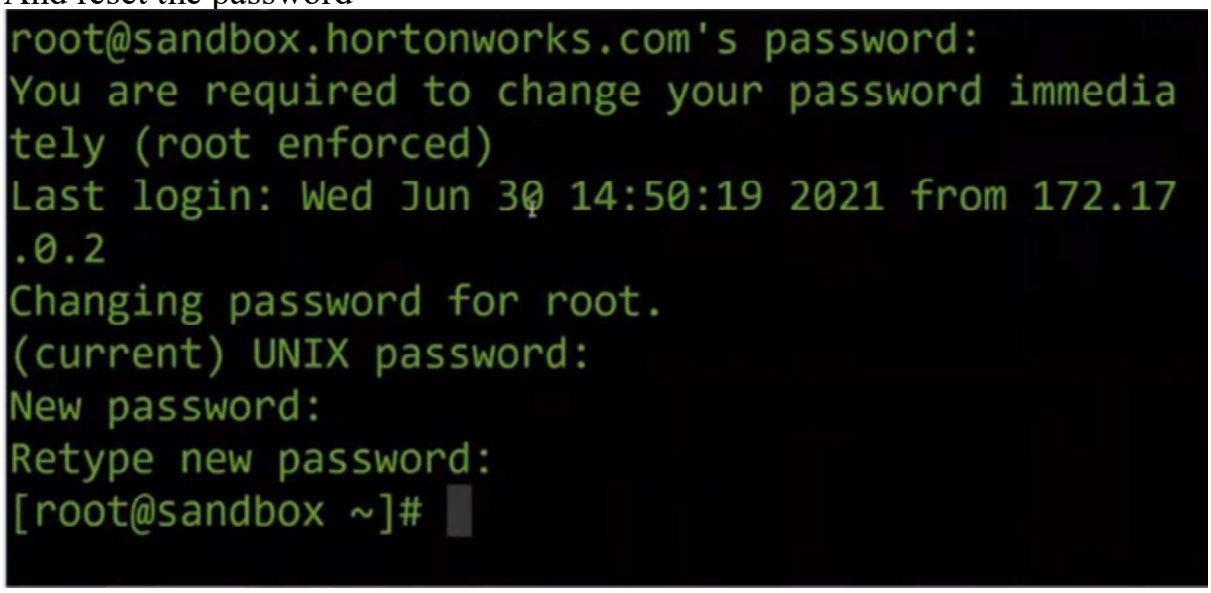
Practical 1

Aim: - Install, configure, and run Hadoop and HDFS, and explore HDFS.

Download Virtual machine setup, i.e., VMware setup (in which Hadoop is configured. Step 1: Load the server on VMware Workstation



Step 2: To enable admin login, open the shell and reset root login. Open Terminal 192.168.119.132:4200
In Sandbox, login with root
and the Password is Hadoop
And reset the password



Windows Linux system and Hadoop system are different
When we type **ls** command it is executed in local system

```
(current) UNIX password:
New password:
Retype new password:
[root@sandbox ~]# ls
anaconda-ks.cfg  install.log.syslog
blueprint.json   sandbox.info
build.out        start_ambari.sh
hdp              start_hbase.sh
install.log
[root@sandbox ~]#
```

When we type **hdfs dfs -ls** it will execute in Hadoop system directory

```
New password:
Retype new password:
[root@sandbox ~]# ls
anaconda-ks.cfg  install.log.syslog
blueprint.json   sandbox.info
build.out        start_ambari.sh
hdp              start_hbase.sh
install.log
[root@sandbox ~]# hdfs dfs -ls /
```

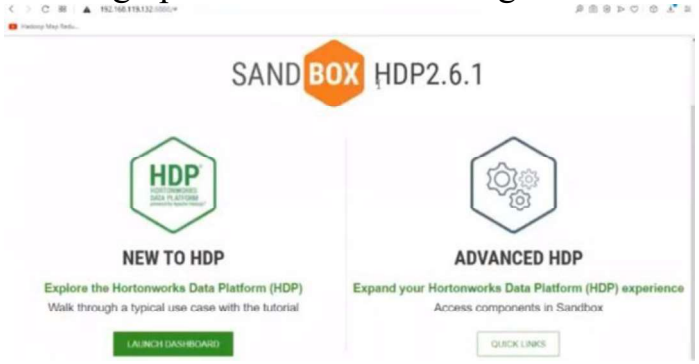
Step 3: Reset Admin account Password

```
[root@sandbox ~]# ambari-admin-password-reset
Please set the password for admin:
Please retype the password for admin:

The admin password has been set.
Restarting ambari-server to make the password change effective...

Using python  /usr/bin/python
Restarting ambari-server
Waiting for server stop...
Ambari Server stopped
Ambari Server running with administrator privileges.
Organizing resource files at /var/lib/ambari-server/resources...
Ambari database consistency check started...
Server PID at: /var/run/ambari-server/ambari-server.pid
Server out at: /var/log/ambari-server/ambari-server.out
Server log at: /var/log/ambari-server/ambari-server.log
Waiting for server start.....
```

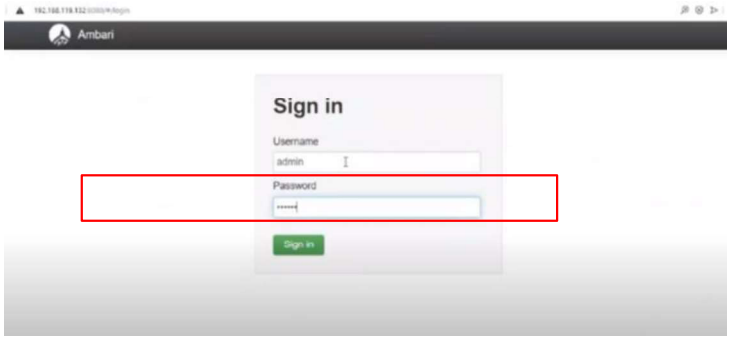
Server listening on 8080 and shell login is complete.

```
The admin password has been set.
Restarting ambari-server to make the password change effective...

Using python  /usr/bin/python
Restarting ambari-server
Waiting for server stop...
Ambari Server stopped
Ambari Server running with administrator privileges.
Organizing resource files at /var/lib/ambari-server/resources...
Ambari database consistency check started...
Server PID at: /var/run/ambari-server/ambari-server.pid
Server out at: /var/log/ambari-server/ambari-server.out
Server log at: /var/log/ambari-server/ambari-server.log
Waiting for server start.....................................................
Server started listening on 8080

DB configs consistency check: no errors and warnings were found.
[root@sandbox ~]#
```

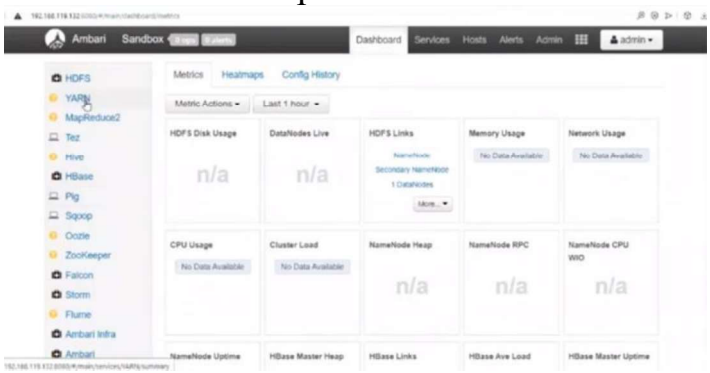
To use graphical user interface login to 192.168.119.132:4200



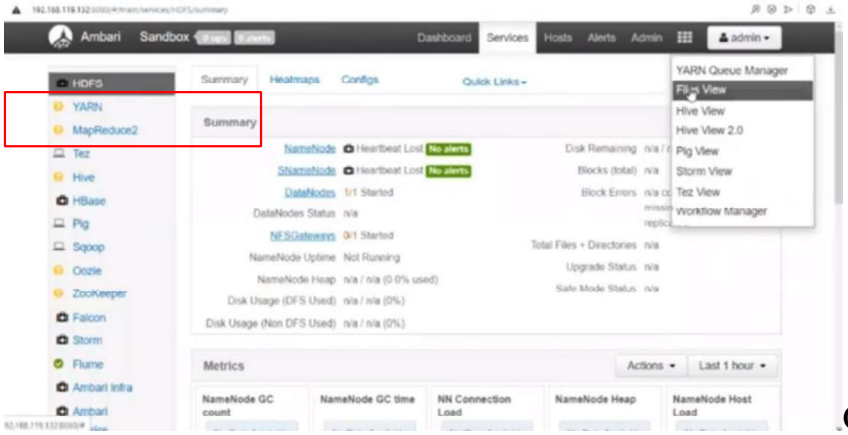
Click on Launch DashboardEnter the username and password for admin login.



Below is the Hadoop server.



To view file in HDFS click on HDFS and click on File view.

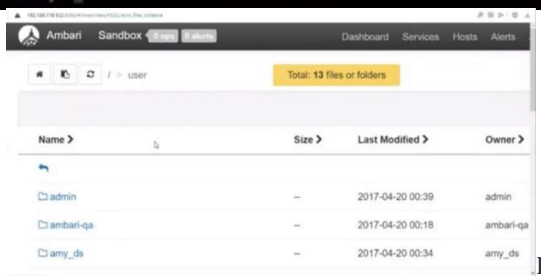


Commands:

1) To view root folder file from terminal use command hdfs dfs -ls /user and press enter.
   It will display all the files in the root user that we see in UI (Screenshot 2).

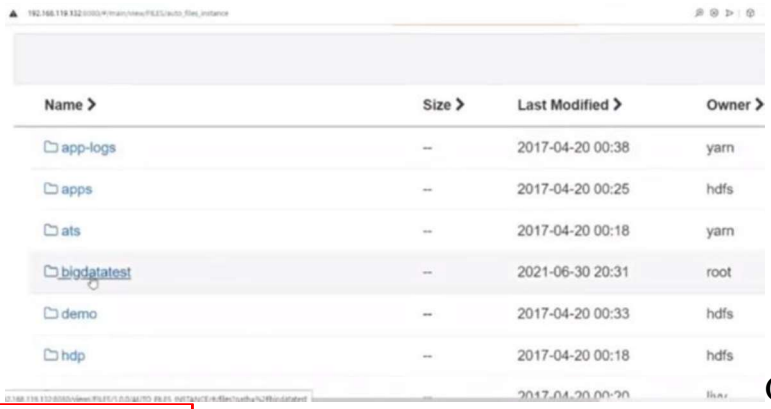
ls: This command is used to list all the files

```
DB configs consistency check: no errors and warnings were found.
[root@sandbox ~]# hdfs dfs -ls /user
Found 13 items
drwxr-xr-x   - admin      hdfs          0 2017-04-19 19:09 /user/admin
drwxrwx---   - ambari-qa  hdfs          0 2017-04-19 18:48 /user/ambari-qa
drwxr-xr-x   - amy_ds     hdfs          0 2017-04-19 19:04 /user/amy_ds
drwxr-xr-x   - hbase      hdfs          0 2017-04-19 18:48 /user/hbase
drwxr-xr-x   - hcat       hdfs          0 2017-04-19 18:51 /user/hcat
drwxr-xr-x   - hive       hdfs          0 2017-04-19 19:08 /user/hive
drwxr-xr-x   - holger_gov hdfs          0 2017-04-19 19:05 /user/holger_gov
drwxrwxr-x   - livy       hdfs          0 2017-04-19 18:49 /user/livy
drwxr-xr-x   - maria_dev  hdfs          0 2017-04-19 18:58 /user/maria_dev
drwxrwxr-x   - oozie      hdfs          0 2017-04-19 18:52 /user/oozie
drwxr-xr-x   - raj_ops    hdfs          0 2017-04-19 19:06 /user/raj_ops
drwxrwxr-x   - spark      hdfs          0 2017-04-19 18:49 /user/spark
drwxr-xr-x   - zeppelin   hdfs          0 2017-04-19 18:49 /user/zeppelin
[root@sandbox ~]#
```

2) 

mkdir: To create a directory.

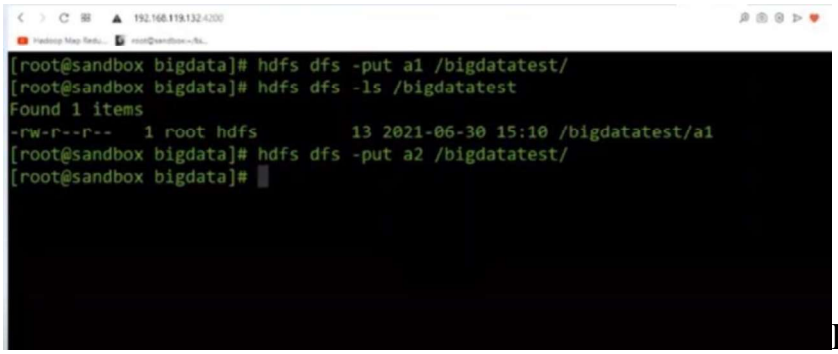
Create a folder in the Hadoop directory. Type the command hdfs dfs -mkdir /bigdata test and enter. After it executes the command, we will see whether it creates a folder in UI.



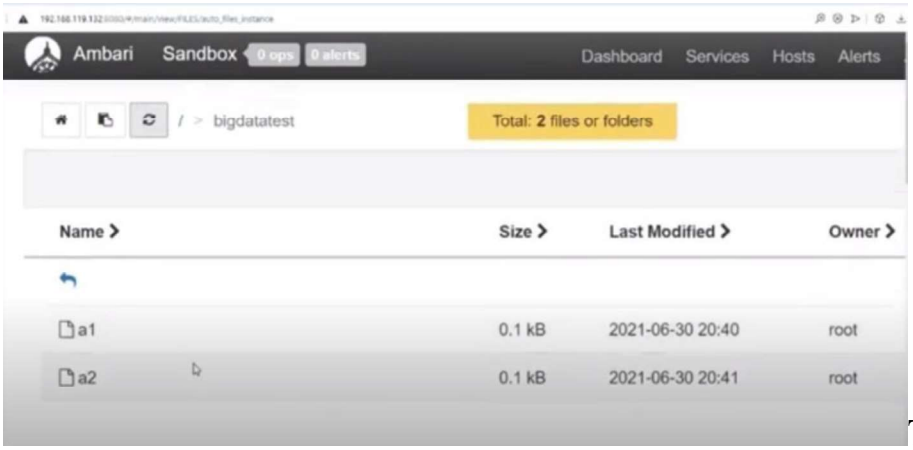
3) Create a file in local directory

Cat: Create a file.
Cat>>
To terminate, press Ctrl+d

4) To upload files/directory from local to HDFS
Put: to move a local file or directory into the distributed file systemCommand: hdfs dfs -put a1 /bigdata test/, and hdfs dfs -put a2
/bigdata test/ will upload both the files.

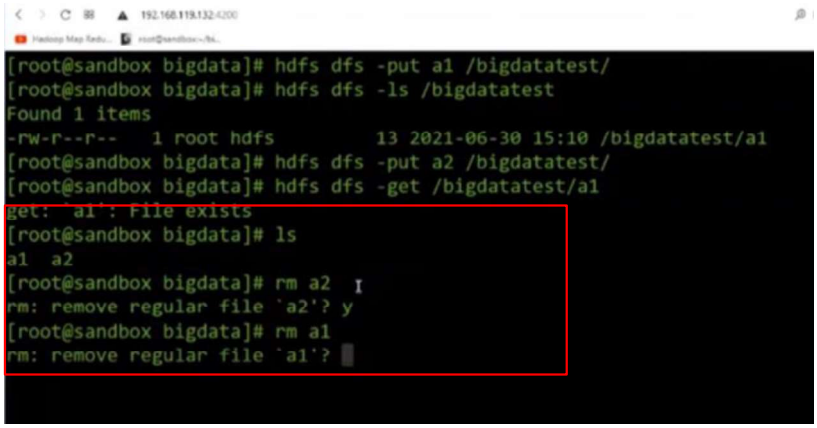


Refresh the user interface we can see both the files.



To download files/directories to from hdfs to local**Get:** To copy files/folders from hdfs store to local file system.Command: hdfs dfs -get /bigdatatest/a1 and hdfs dfs -get /bigdatatest/ a2 will upload both the files.

5) To remove file from local use rm command

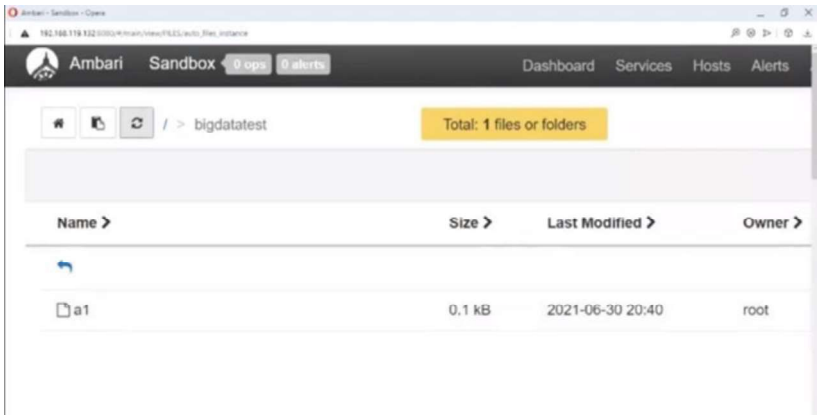


6) To remove file from Hadoop directory

Command: hdfs dfs -rm a2 /gibdatatest/a2

```
[root@sandbox bigdata]# hdfs dfs -rm /bigdattest/a2
rm: `/bigdattest/a2': No such file or directory
[root@sandbox bigdata]# hdfs dfs -rm /bigdatatest/a2
21/06/30 15:16:27 INFO fs.TrashPolicyDefault: Moved: 'hdfs://sandbox.hortonwor
om:8020/bigdatatest/a2' to trash at: hdfs://sandbox.hortonworks.com:8020/user/
/.Trash/Current/bigdatatest/a2
[root@sandbox bigdata]#
```

Now refresh the UI and the file will be deleted.



7) To download all the files from hdfs to local

Command: hdfs dfs -get /bigdatatest/*

8) Change user and directory and change user only

Command: su – hdfs (Change user and directory)
Su hdfs (change user only)

```
[root@sandbox bigdata]# hdfs dfs -get /bigdatatest/

[root@sandbox bigdata]# hdfs dfs -get /bigdatatest/*
[root@sandbox bigdata]# ls
a1  iot all.pdf  mscit  test
[root@sandbox bigdata]# hdfs dfs -put * /bigdatatest/x
put: unexpected URISyntaxException
put: `/bigdatatest/x': No such file or directory
[root@sandbox bigdata]# pwd
/root/bigdata
[root@sandbox bigdata]# su hdfs
[hdfs@sandbox bigdata]$ pwd
/root/bigdata
[hdfs@sandbox bigdata]$ exit
exit
[root@sandbox bigdata]# su -Ihdfs
[hdfs@sandbox ~]$ pwd
/home/hdfs
[hdfs@sandbox ~]$
```

Practical 2

Aim: Implement word count/frequency programs using MapReduce Map Reduce as a two-component Map and Reduce.

Java program:

Write a program, save as WordCount.java//////////////////////

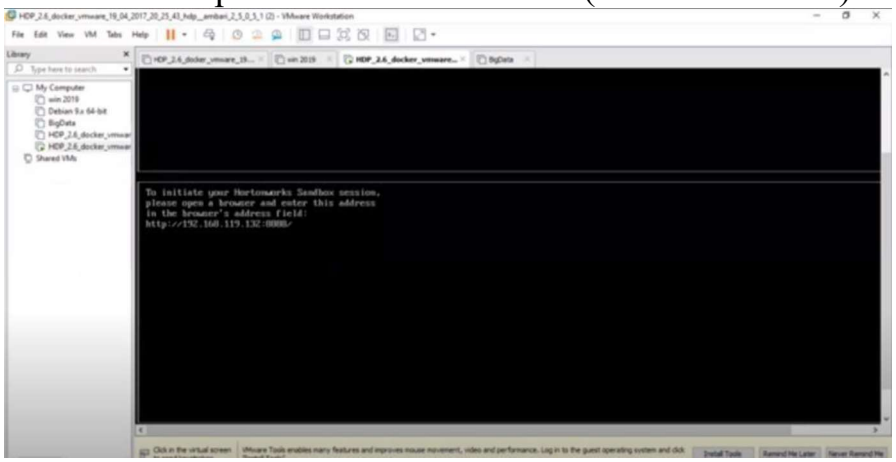
```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
public class WordCount { public static class TokenizerMapper extends
Mapper<Object, Text, Text, IntWritable>{
 private final static IntWritable one = new IntWritable(1);
 private Text word = new Text();
 public void map(Object key, Text value, Context
 ) throws IOException, InterruptedException {
 StringTokenizer itr = new StringTokenizer(value.toString());
 while (itr.hasMoreTokens()) {//"This is the output is the"
 word.set(itr.nextToken());
 context.write(word, one);
 } } }
 public static class IntSumReducer extends
Reducer<Text,IntWritable,Text,IntWritable> {
private IntWritable result = new IntWritable();
 public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException,
InterruptedException
 {//is,3
 int sum = 0;
 for (IntWritable val : values) {
 sum += val.get();
 }
 result.set(sum);
 context.write(key, result);
```

```
 }
 }public static void main(String[] args) throws Exception {
 Configuration conf = new Configuration();
 Job = Job.getInstance(conf, "word count");
 job.setJarByClass(WordCount.class);
 job.setMapperClass(TokenizerMapper.class);
 job.setCombinerClass(IntSumReducer.class);
 job.setReducerClass(IntSumReducer.class);
 job.setOutputKeyClass(Text.class);
 job.setOutputValueClass(IntWritable.class);
 FileInputFormat.addInputPath(job, new Path(args[0]));
 FileOutputFormat.setOutputPath(job, new Path(args[1]));
 System.exit(job.waitForCompletion(true)?0:1);
 }
}
```

//////////////////////**Text File:**

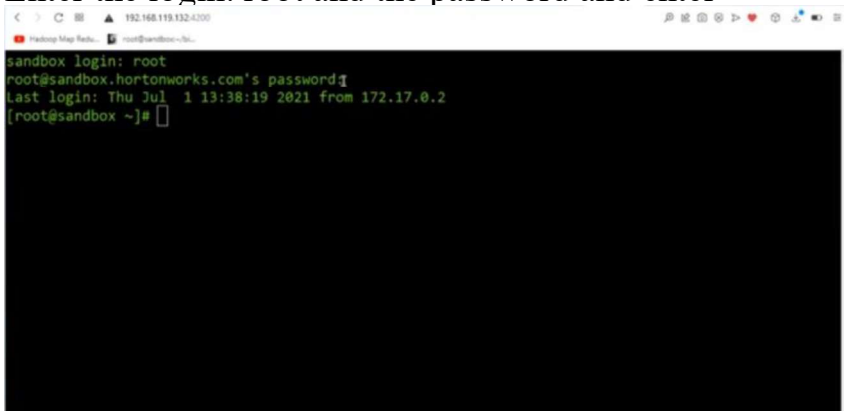
Hello World

This is the output is theStart the server (Horton Sandbox)



Open the terminal with 192.168.119.132/4200

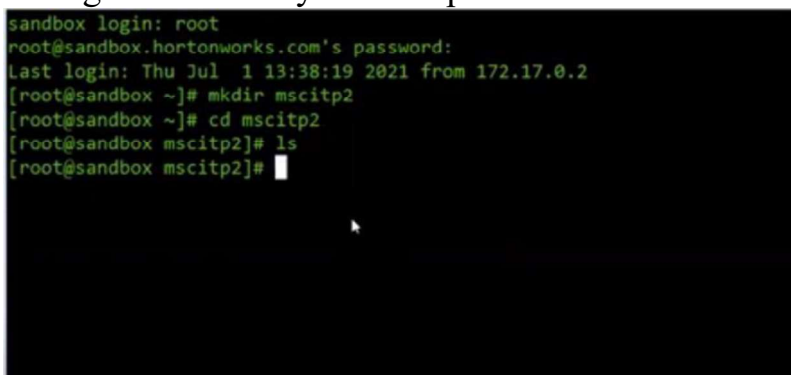
Enter the login: root and the password and enter



Create a folder in local directory.

Command: mkdir mscitp2

Change the directory cd mscitp2



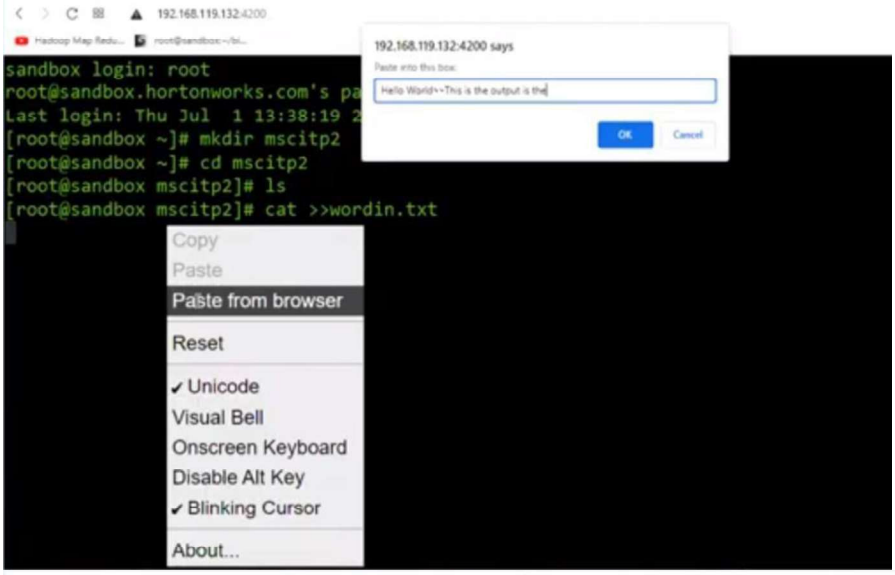
Now create input file

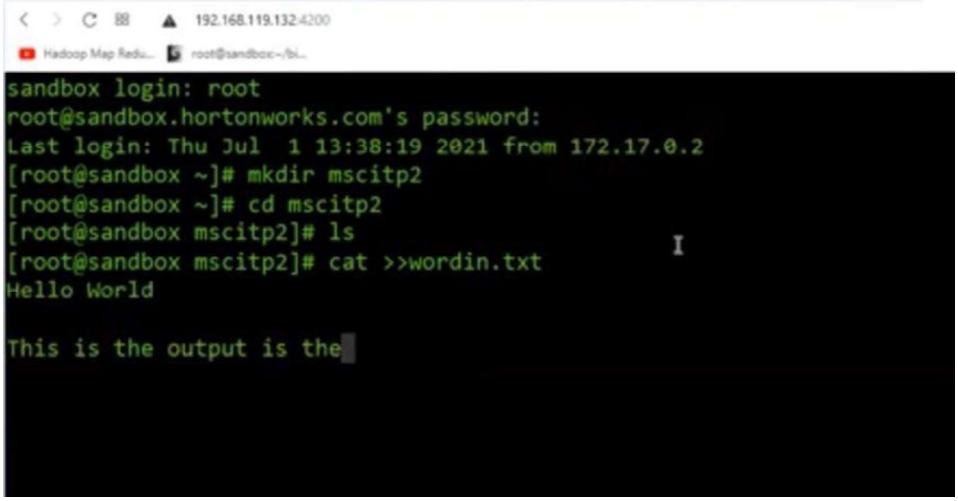
Command: cat >> wordin.txt

Paste the text by right clicking on terminal

Hello World

This is the output is the



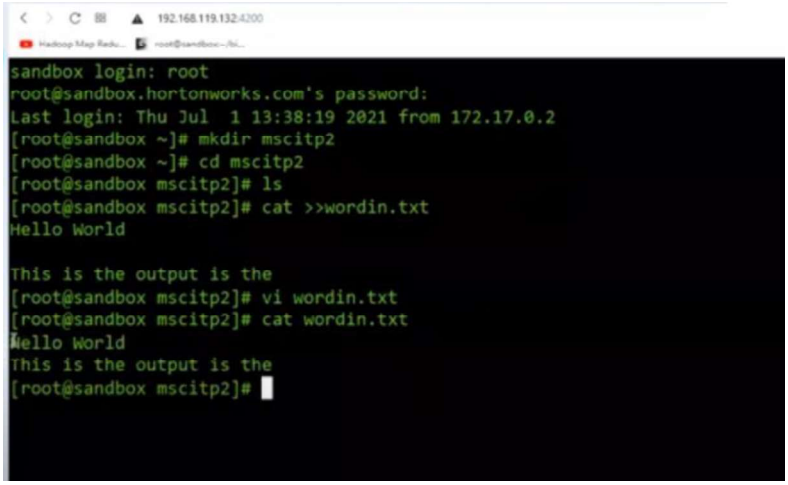


To remove the extra space type command

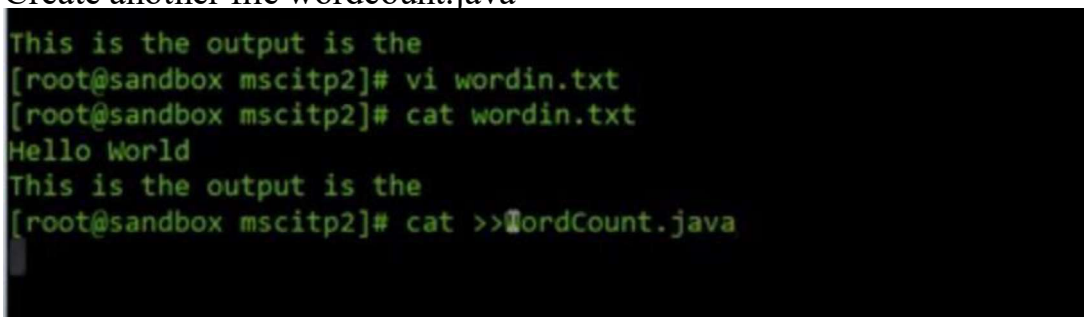
vi wordin.txt

After removing the extra space check the content of the file

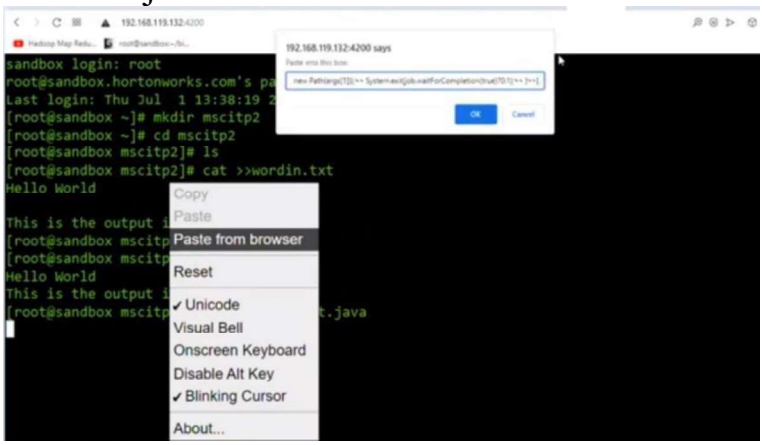
cat wordin.txt



Create another file wordcount.java



Paste the java code.



Press control d to save the file

Check both the files create with command ls

Now, to compile the java file
export HADOOP_CLASSPATH=$(hadoop classpath)
mkdir classes (To keep the compile files)
javac -classpath ${HADOOP_CLASSPATH} -d classes WordCount.java

```
[root@sandbox mscitp2]# ls
WordCount.java  wordin.txt
[root@sandbox mscitp2]# export HADOOP_CLASSPATH=$(hadoop classpath)
[root@sandbox mscitp2]# ls
WordCount.java  wordin.txt
[root@sandbox mscitp2]# mkdir classes
[root@sandbox mscitp2]# javac -classpath ${HADOOP_CLASSPATH} -d classes WordCount.java
[root@sandbox mscitp2]#
```

Check class files are created with command ls classes

```
WordCount.java  wordin.txt
[root@sandbox mscitp2]# mkdir classes
[root@sandbox mscitp2]# javac -classpath ${HADOOP_CLASSPATH} -d classes WordCount.java
[root@sandbox mscitp2]# ls classes
WordCount.class  WordCount$IntSumReducer.class  WordCount$TokenizerMapper.class
[root@sandbox mscitp2]#
```

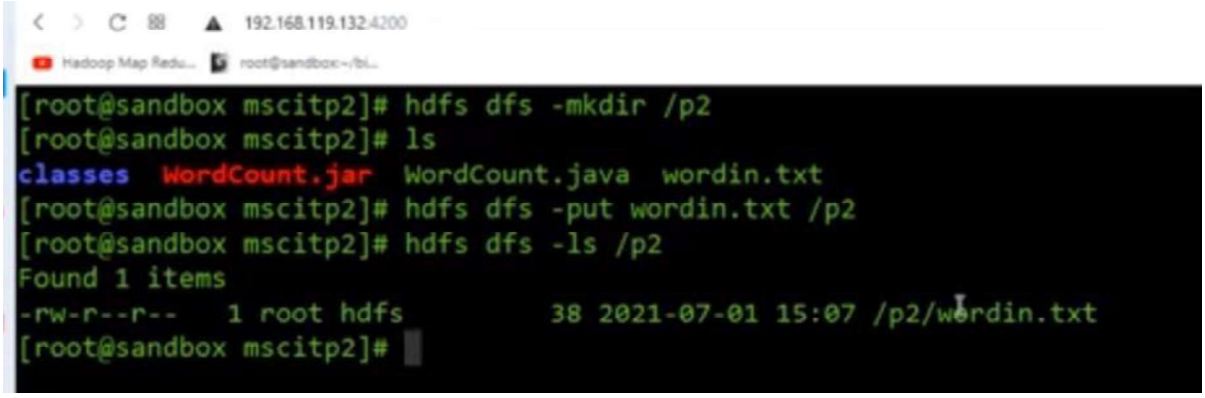
Now we have to bind all the class into single jar file with below command
jar -cvf WordCount.jar -C classes/ .

```
[root@sandbox mscitp2]# ls
WordCount.java  wordin.txt
[root@sandbox mscitp2]# export HADOOP_CLASSPATH=$(hadoop classpath)
[root@sandbox mscitp2]# ls
WordCount.java  wordin.txt
[root@sandbox mscitp2]# mkdir classes
[root@sandbox mscitp2]# javac -classpath ${HADOOP_CLASSPATH} -d classes WordCount.java
[root@sandbox mscitp2]# ls classes
WordCount.class  WordCount$IntSumReducer.class  WordCount$TokenizerMapper.class
[root@sandbox mscitp2]# ls
classes  WordCount.java  wordin.txt
[root@sandbox mscitp2]# jar -cvf WordCount.jar -C classes/ .
added manifest
adding: WordCount$IntSumReducer.class(in = 1739) (out= 742)(deflated 57%)
adding: WordCount$TokenizerMapper.class(in = 1736) (out= 756)(deflated 56%)
adding: WordCount.class(in = 1491) (out= 813)(deflated 45%)
[root@sandbox mscitp2]#
```

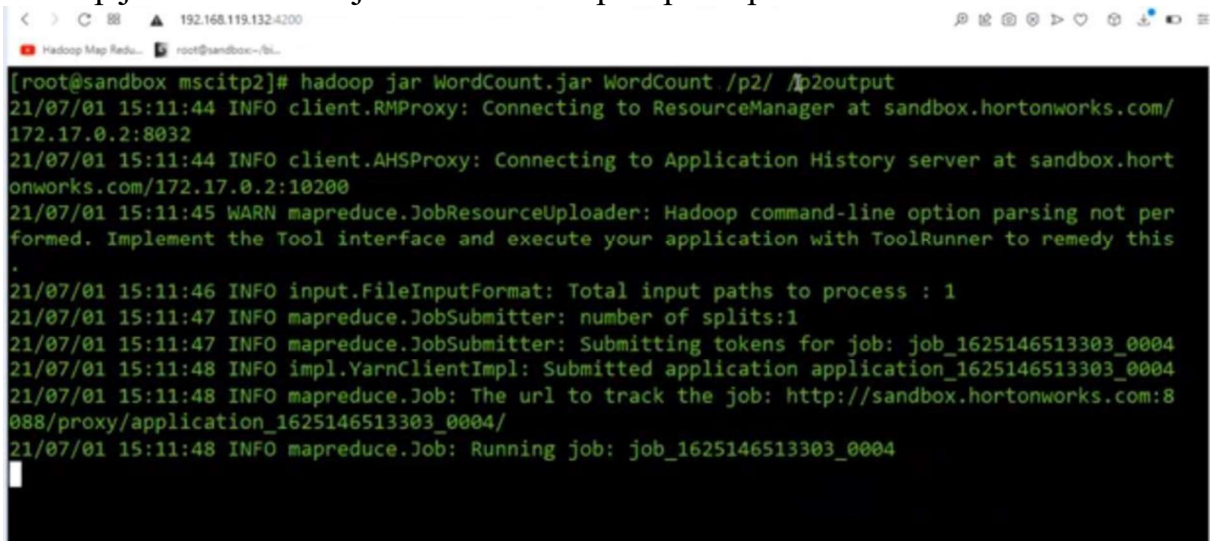
Run ls command we can see jar file is created.

```
[root@sandbox mscitp2]# ls
classes  WordCount.jar  WordCount.java  wordin.txt
[root@sandbox mscitp2]#
```

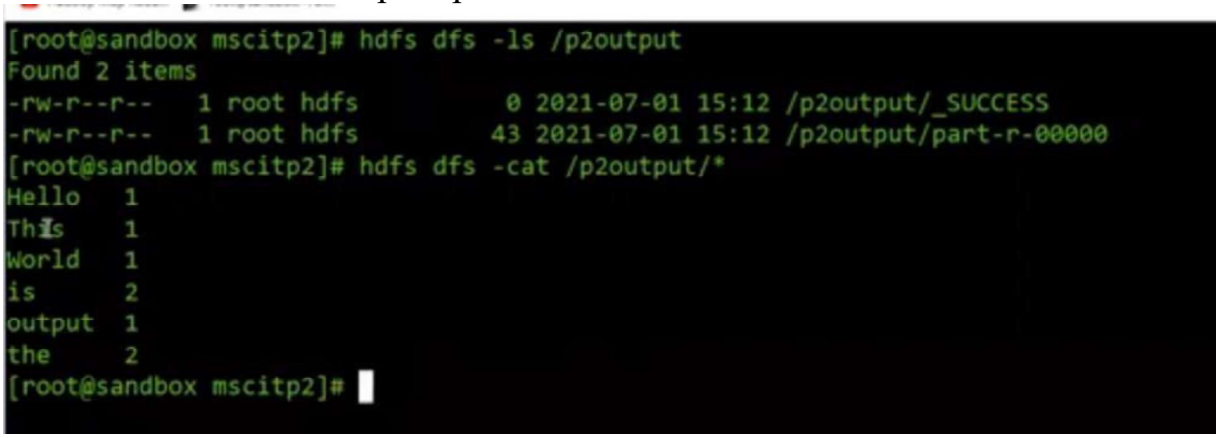
wordin.txt should be present in word directory of hdfs. So we need to upload wordin.txt file.



We need to put the final output p2output.
hadoop jar WordCount.jar WordCount /p2/ /p2output



Print the content of the output file
Command: hdfs dfs -cat /p2ouput/*



Ctrl + l to clear the screen.
vi filename.txt= this command will create/ open filename.txt
two modes of vi editor

1) Insert mode – i (press i key)
2) Command mode – esc key :wq is to save and exi

Practical 3

Aim: - Implement a MapReduce program that processes a weather dataset.

Java program:

MyMaxMin.java

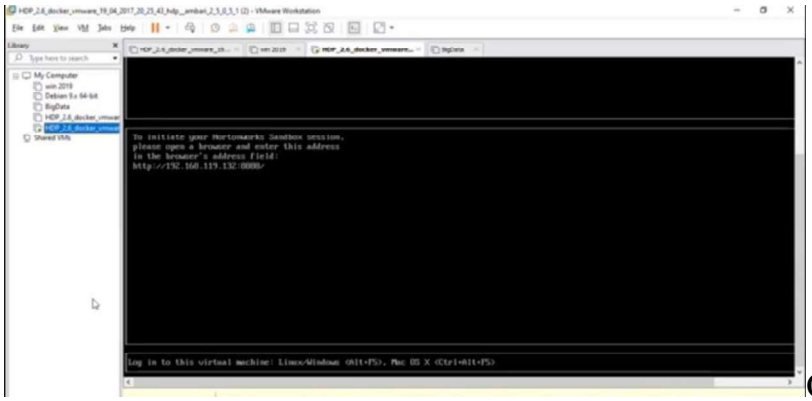
```
//////////////////////////
// importing Libraries
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.conf.Configuration;public class MyMaxMin {
        // Mapper

        /*MaxTemperatureMapper class is static
        * and extends Mapper abstract class
        * having four Hadoop generics type
        * LongWritable, Text, Text, Text.
        */


        public static class MaxTemperatureMapper extends
                        Mapper<LongWritable, Text, Text, Text> {
public static final int MISSING = 9999;

        @Override
                public void map(LongWritable arg0, Text Value, Context context)
                                throws IOException, InterruptedException {

                String line = Value.toString();

                        // Check for the empty line
                        if (!(line.length() == 0)) {
```

```
                        // from character 6 to 14 we have
                        // the date in our dataset
                        String date = line.substring(6, 14);
        // similarly we have taken the maximum
                        // temperature from 39 to 45 characters
                        float temp_Max = Float.parseFloat(line.substring(39,
45).trim());

                        // similarly we have taken the minimum
                        // temperature from 47 to 53 characters

                        float temp_Min = Float.parseFloat(line.substring(47,
53).trim());                    // if maximum temperature is
                        // greater than 30, it is a hot day
                        if (temp_Max > 30.0) {

                                // Hot day
                                context.write(new Text("The Day is Hot Day :" +
date),
                                                                new
Text(String.valueOf(temp_Max)));
                        }                       // if the minimum temperature
is
                        // less than 15, it is a cold day
                        if (temp_Min < 15) {

                                // Cold day
                                context.write(new Text("The Day is Cold Day :" +
date),
                                                        new
Text(String.valueOf(temp_Min)));
                        }
                }
        }       }// Reducer

        /*MaxTemperatureReducer class is static
        and extends Reducer abstract class
        having four Hadoop generics type
        Text, Text, Text, Text.
        */
        //The Day is Cold Day :20150101 ,-21.8
```

```
        public static class MaxTemperatureReducer extends
                        Reducer<Text, Text, Text, Text> {               /**
                * @method reduce
                * This method takes the input as key and
                * list of values pair from the mapper,
                * it does aggregation based on keys and
                * produces the final context.
                */

                public void reduce(Text Key, Iterator<Text> Values, Context
context)
                                throws IOException, InterruptedException {
                        // putting all the values in
                        // temperature variable of type String
                        String temperature = Values.next().toString();
                        context.write(Key, new Text(temperature));
                }       }       /**
        * @method main
        * This method is used for setting
        * all the configuration properties.
        * It acts as a driver for map-reduce
        * code.
        */

        public static void main(String[] args) throws Exception {          // reads the
default configuration of the
                // cluster from the configuration XML files
                Configuration conf = new Configuration();

                // Initializing the job with the
                // default configuration of the cluster
                Job = new Job(conf, "weather example");

                // Assigning the driver class name
                job.setJarByClass(MyMaxMin.class);          // Key type coming out
of mapper
                job.setMapOutputKeyClass(Text.class);

                // value type coming out of mapper
                job.setMapOutputValueClass(Text.class);             // Defining the
mapper class name
```

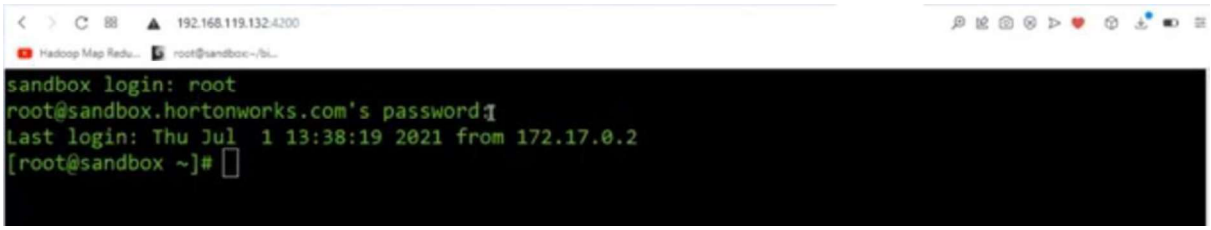
```
        job.setMapperClass(MaxTemperatureMapper.class);

        // Defining the reducer class name
        job.setReducerClass(MaxTemperatureReducer.class);        //
Defining input Format class which is
        // responsible to parse the dataset
        // into a key value pair
        job.setInputFormatClass(TextInputFormat.class);

        // Defining output Format class which is
        // responsible to parse the dataset
        // into a key value pair
        job.setOutputFormatClass(TextOutputFormat.class);        //
setting the second argument
        // as a path in a path variable
        Path OutputPath = new Path(args[1]);        // Configuring the input
path
        // from the filesystem into the job
        FileInputFormat.addInputPath(job, new Path(args[0]));        //
Configuring the output path from
        // the filesystem into the job
        FileOutputFormat.setOutputPath(job, new Path(args[1]));        //
deleting the context path automatically
        // from hdfs so that we don't have
        // to delete it explicitly
        OutputPath.getFileSystem(conf).delete(OutputPath);
        // flag value becomes false
        System.exit(job.waitForCompletion(true) ? 0 : 1);  }
}
```

//////////////////////Start the server

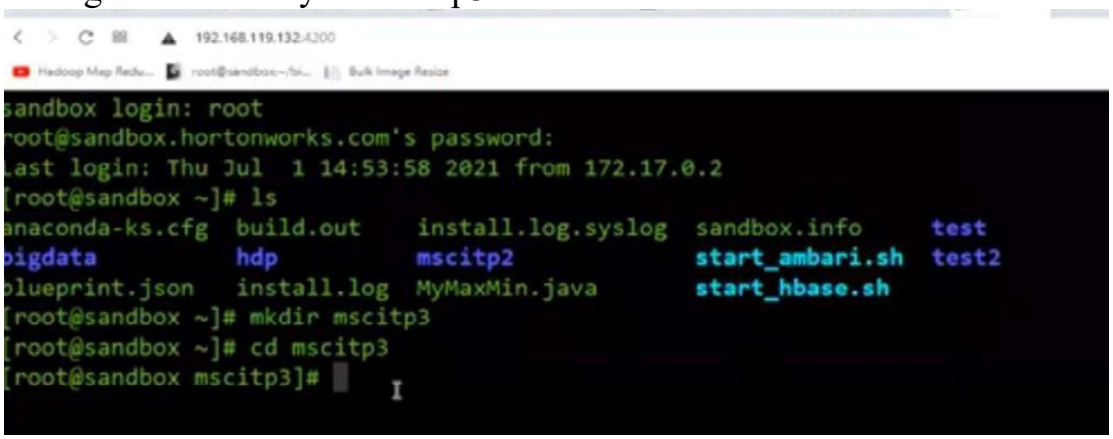


Open the terminal with 192.168.119.132/4200

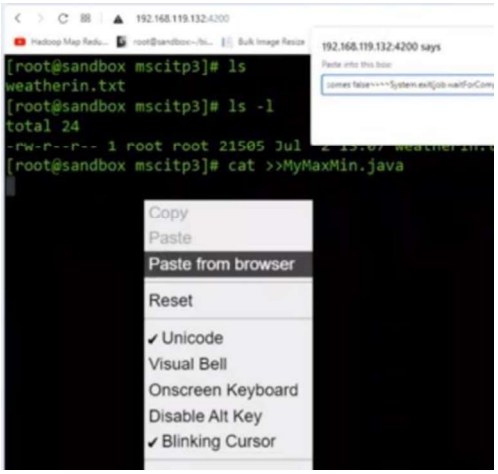
Enter the login: root and the password and enter



Create a folder in local directory.
Command: mkdir mscitp3
Change the directory cd mscitp3

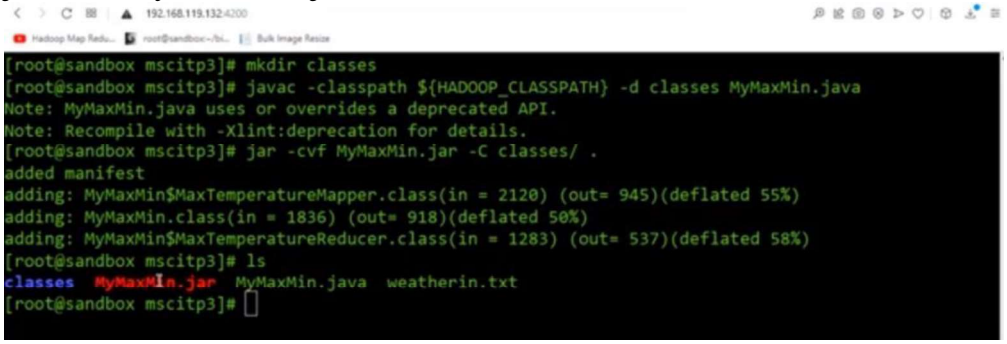


Now create input file
Command: cat >> weatherin2.txt
Paste the weather dataset by right clicking on terminal
Ctrl d will save the file
Run command ls to see the file.
Create java file
Command: cat >>MyMaxMin.java
Paste the java code and ctrl d to save the file



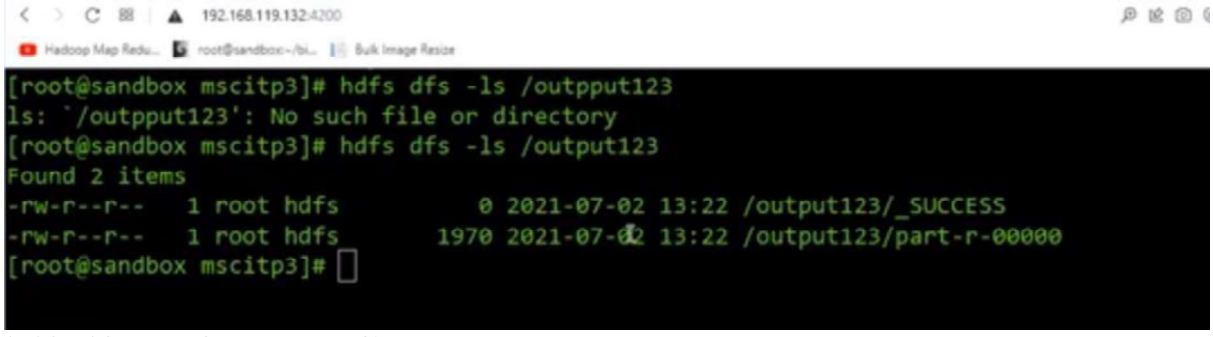
export HADOOP_CLASSPATH=$(hadoop classpath) ////compile and to create jar file
mkdir classes
javac -classpath ${HADOOP_CLASSPATH} -d classes MyMaxMin.java
After compile need to create a jar file

jar -cvf MyMaxMin.jar -C classes/ .

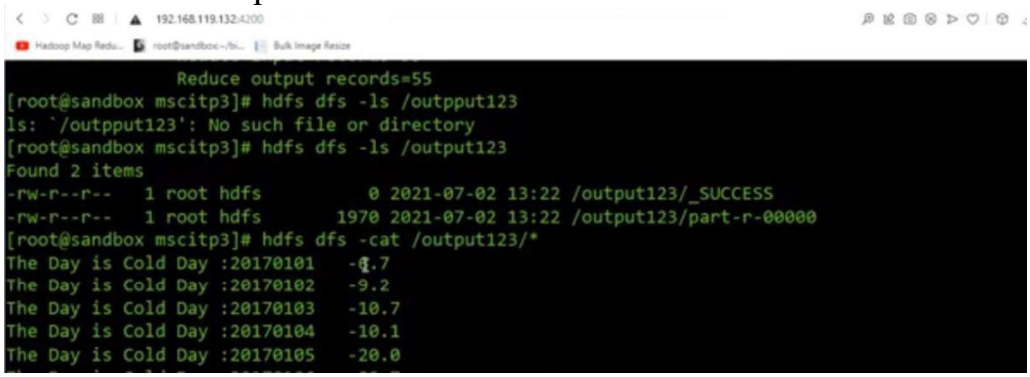


Now, put weatherin.txt in hdfs
Before that create a folder
Command: hdfs dfs -mkdir /p3input123
Then run command: hdfs dfs -put weatherin2.txt /p3input123
hadoop jar MyMaxMin.jar MyMaxMin /p3inputw /output123

Check outfile is created
Command: hdfs dfs -ls /output123



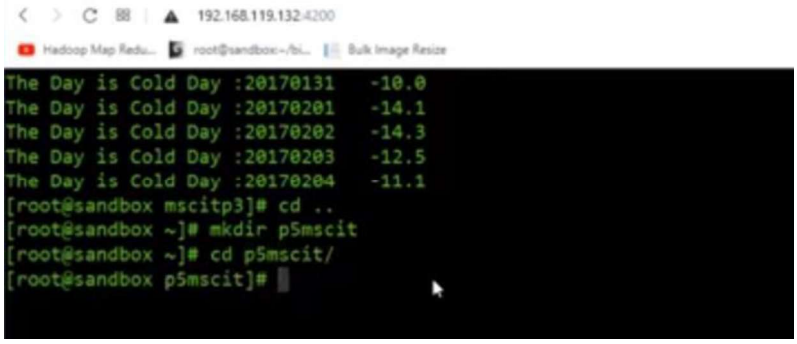
hdfs dfs -cat /output123/*



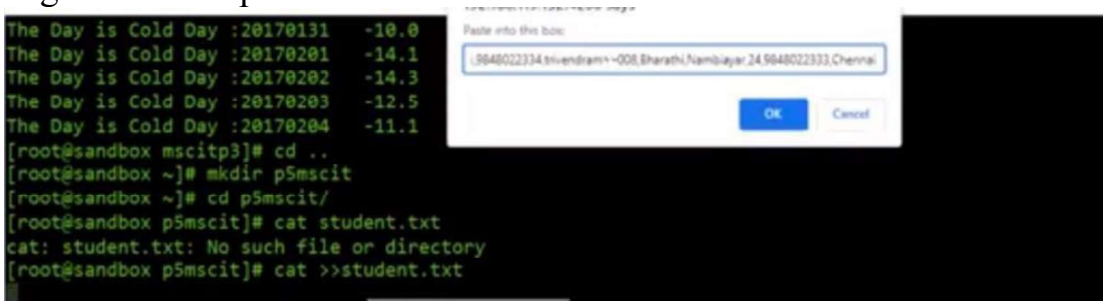
Practical 4 A

Aim: Implement the program using Pig.

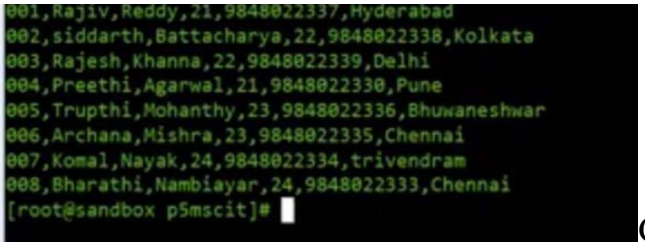
Dataset:
001,Rajiv,Reddy,21,9848022337,Hyderabad
002,siddarth,Battacharya,22,9848022338,Kolkata
003,Rajesh,Khanna,22,9848022339,Delhi
004,Preethi,Agarwal,21,9848022330,Pune
005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar
006,Archana,Mishra,23,9848022335,Chennai
007,Komal,Nayak,24,9848022334,trivendram
008,Bharathi,Nambiayar,24,9848022333,Chennai
#student.txtcreate a directory and get into that directory
Command: mkdir p5mscit



Create a file
Command: cat >>student.txt
Right click and paste the text



Remove the space with vi editor
Command: vi student.txt and press i for insert mode
After editing: wq and enter

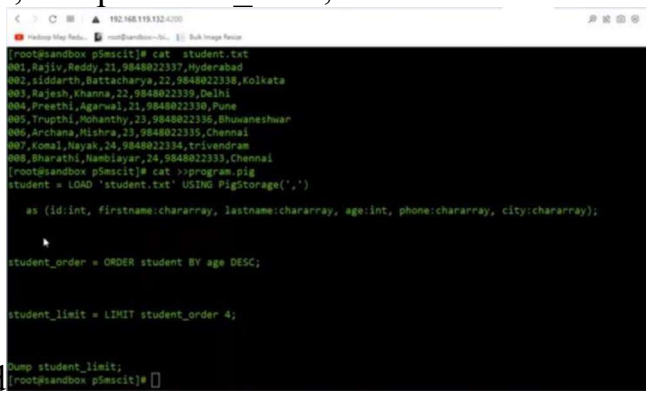
Print the content and see the text 

Create

a program file

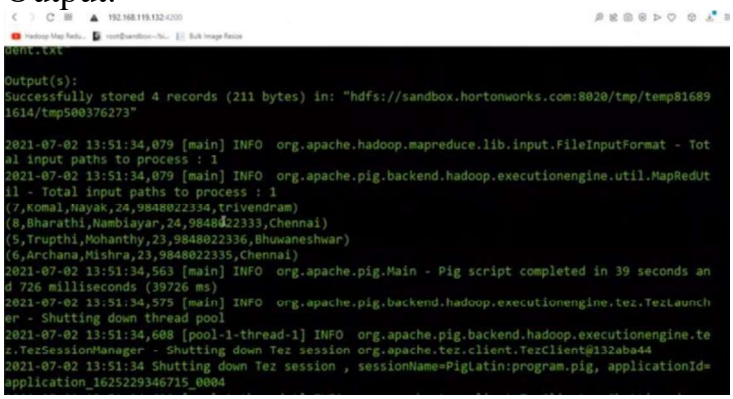
//////////////////////////script start
student = LOAD 'student.txt' USING PigStorage(',')
as (id:int, firstname:chararray, lastname:chararray, age:int, phone:chararray, city:chararray);

student_order = ORDER student BY age DESC;student_limit = LIMIT student_order 4;Dump student_limit;



////////script end
Upload student on hdfs
Command: hdfs dfs -put student.txt /user/root/
Run the pig program

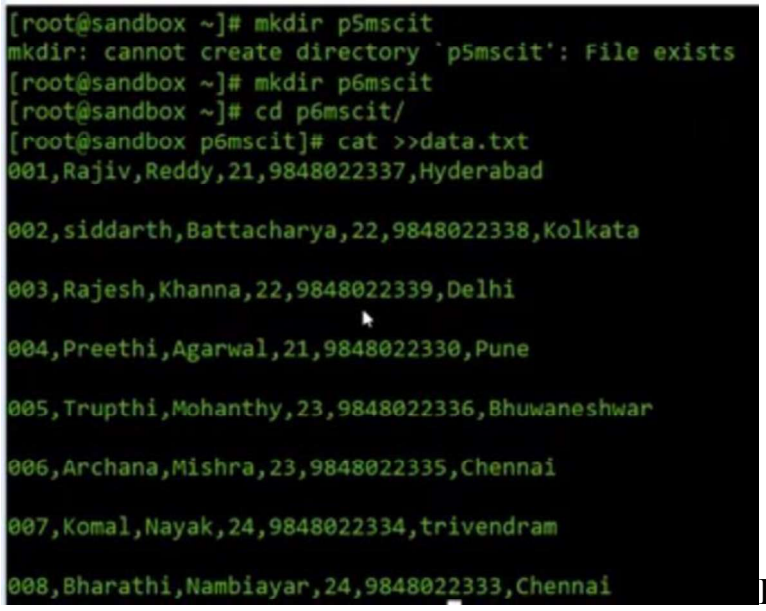
Output:



Practical 5

Aim: Implement the application in Hive.

Dataset:
001,Rajiv,Reddy,21,9848022337,Hyderabad
002,siddarth,Battacharya,22,9848022338,Kolkata
003,Rajesh,Khanna,22,9848022339,Delhi
004,Preethi,Agarwal,21,9848022330,Pune
005,Trupthi,Mohanthy,23,9848022336,Bhuwaneshwar
006,Archana,Mishra,23,9848022335,Chennai
007,Komal,Nayak,24,9848022334,trivendram
008,Bharathi,Nambiayar,24,9848022333,Chennai
#student.txtcreate a directory and get into that directory
Command: mkdir p6mscit
Create a file
Command: cat >>data.txt
Right click and paste the text



Remove the space with vi editor
Command: vi student.txt and press i for insert mode
After editing: wq and enter
Print the content and see the text
Now start the hive terminal
Command: hive

```
[root@sandbox p6mscit]# hive
log4j:WARN No such property [maxFileSize] in org.apache.log4j.DailyRollingFileAppender.

Logging initialized using configuration in file:/etc/hive/2.6.0.3-8/0/hive-log4j.properties
hive>
```

Copy paste below command on hive and entercreate table
CREATE TABLE IF NOT EXISTS employee (eid int, fname String,
lname String, age int, contact String, city String)
COMMENT 'Employee details'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;

```
[root@sandbox p6mscit]# vi data.txt
[root@sandbox p6mscit]# hive
log4j:WARN No such property [maxFileSize] in org.apache.log4j.DailyRollingFileAppender.

Logging initialized using configuration in file:/etc/hive/2.6.0.3-8/0/hive-log4j.properties
hive> CREATE TABLE IF NOT EXISTS employee ( eid int, fname String,
    >
    > lname String, age int, contact String, city String)
    >
    > COMMENT 'Employee details'
    >
    > ROW FORMAT DELIMITED
    >
    > FIELDS TERMINATED BY ','
    >
    > LINES TERMINATED BY '\n'
    >
    > STORED AS TEXTFILE;
```

Run command:
LOAD DATA LOCAL INPATH 'data.txt' OVERWRITE INTO TABLE employee;
Run the command like select * from employee;

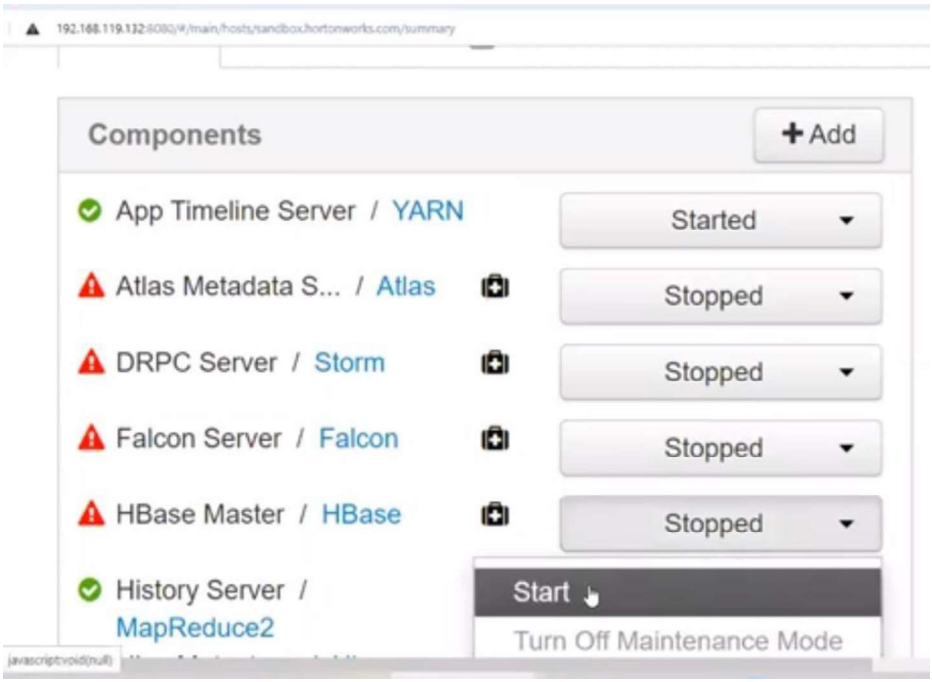
```
Loading data to table default.employee
Table default.employee stats: [numFiles=1, numRows=0, totalSize=339, rawDataSize=0]
OK
Time taken: 1.296 seconds
hive> select * from employee;
OK
1       Rajiv   Reddy   21      9848022337      Hyderabad
2       siddarth        Battacharya     22      9848022338      Kolkata
3       Rajesh  Khanna  22      9848022339      Delhi
4       Preethi Agarwal 21      9848022330      Pune
5       Trupthi Mohanthy        23      9848022336      Bhuwaneshwar
6       Archana Mishra  23      9848022335      Chennai
7       Komal   Nayak   24      9848022334      trivendram
8       Bharathi        Nambiayar       24      9848022333      Chennai
Time taken: 0.188 seconds, Fetched: 8 row(s)
hive> select * from employee where age > 23;
OK
7       Komal   Nayak   24      9848022334      trivendram
8       Bharathi        Nambiayar       24      9848022333      Chennai
Time taken: 0.551 seconds, Fetched: 2 row(s)
hive>
```

Practical 6

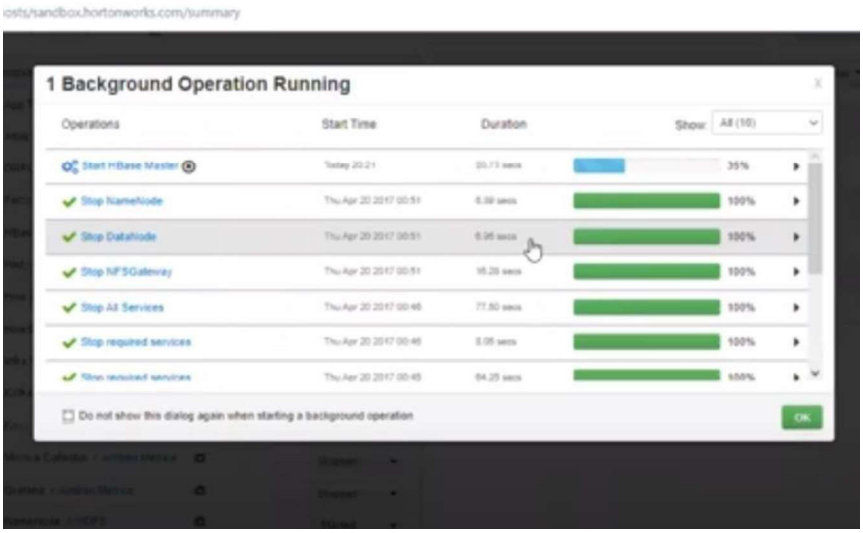
Aim: Implement an application that stores big data in HBase/ Python

What is HBase?

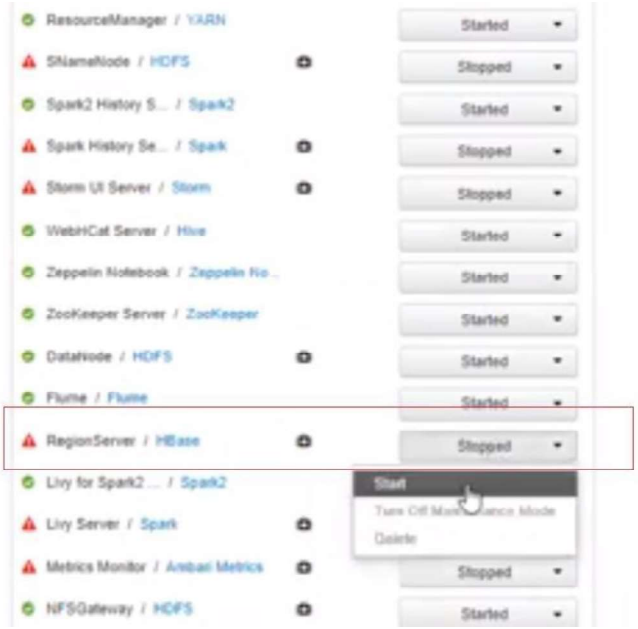
HBase is a distributed column-oriented database built on top of the Hadoop file system. It is an open-source project and is horizontally scalable. It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System. Go to GUI page and start the HBase service.

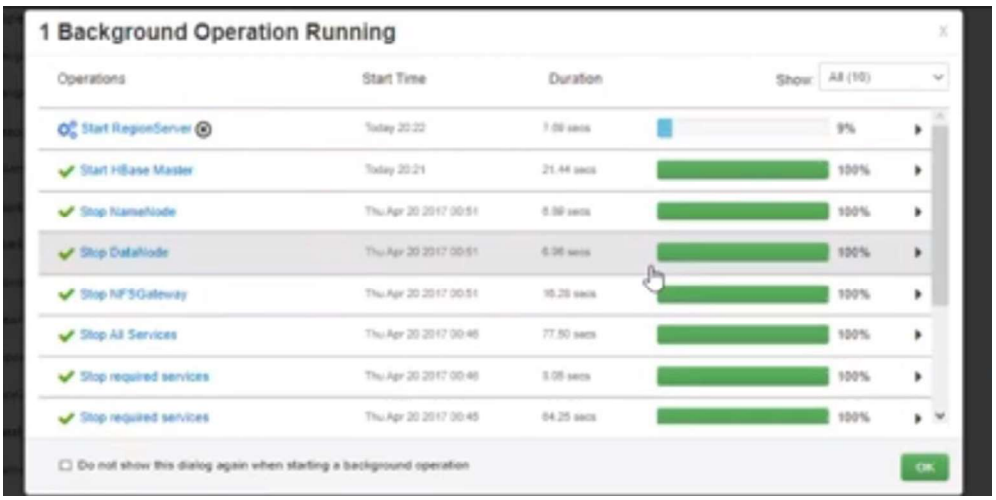


Click on OK to start the service.

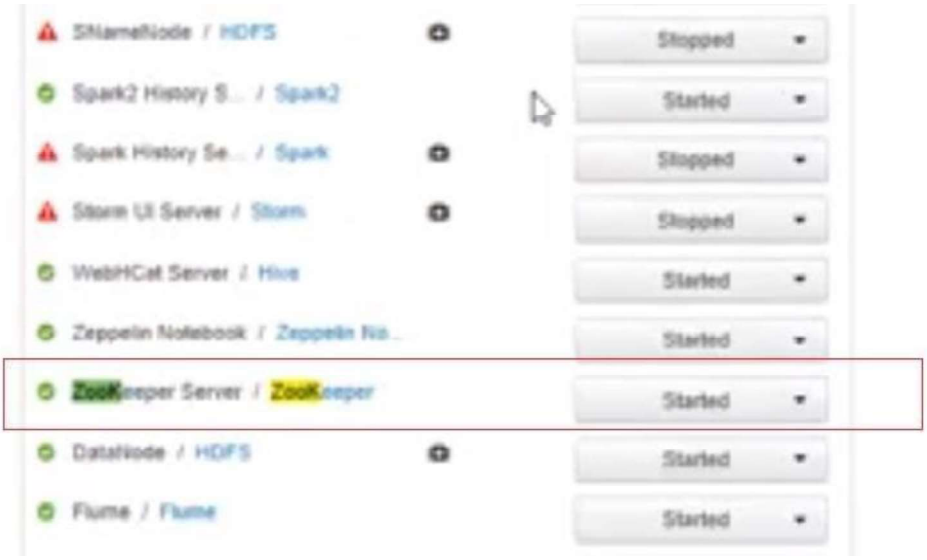


Now we must start region server.

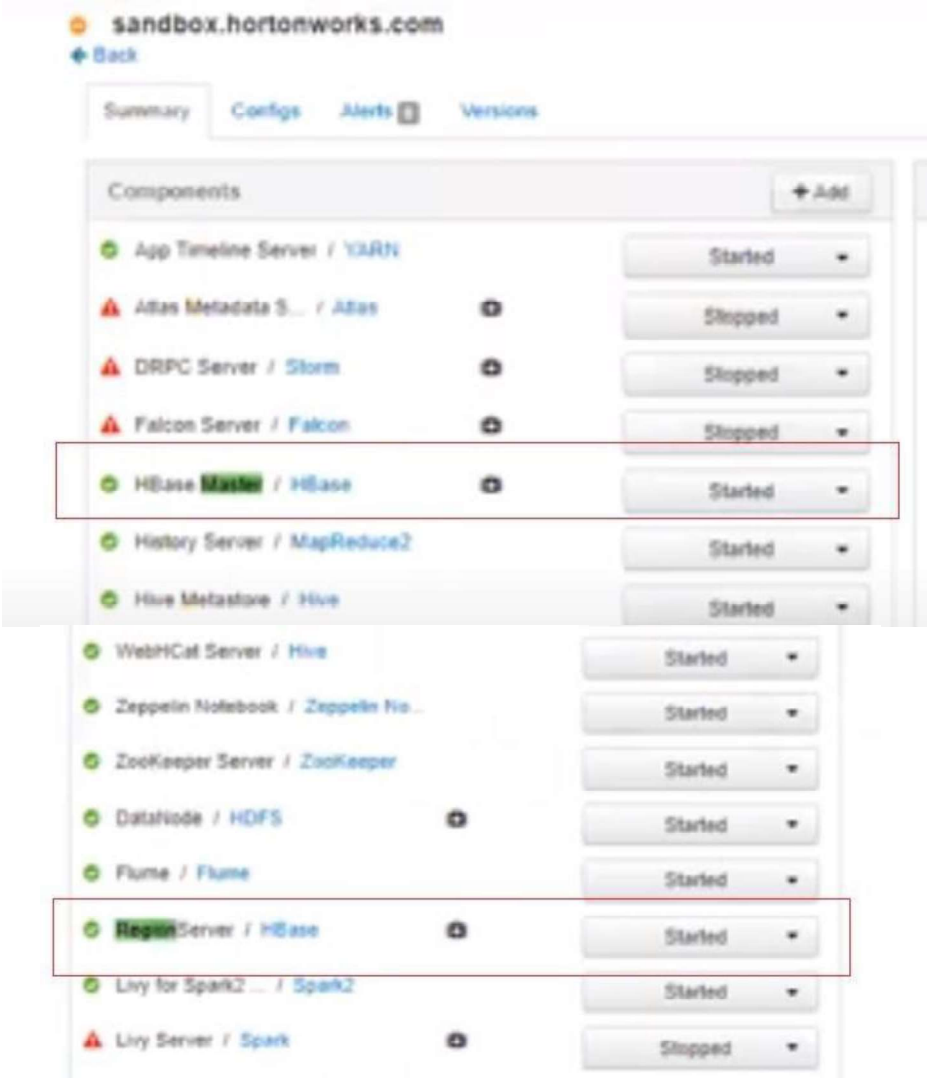




Check zooperkeeper server is started.

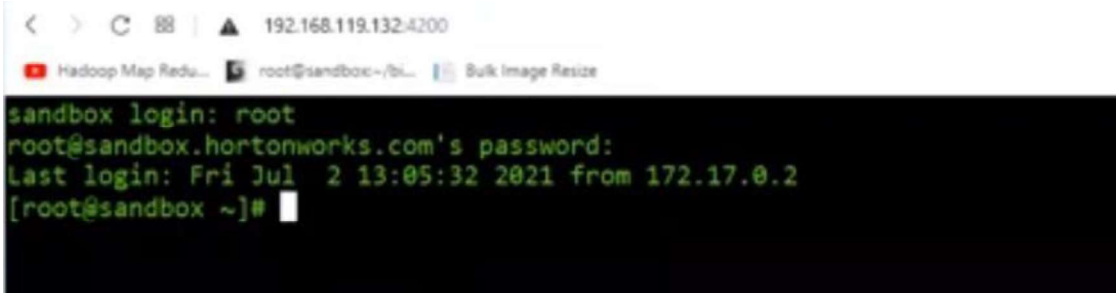


Check hbase and region server are started.

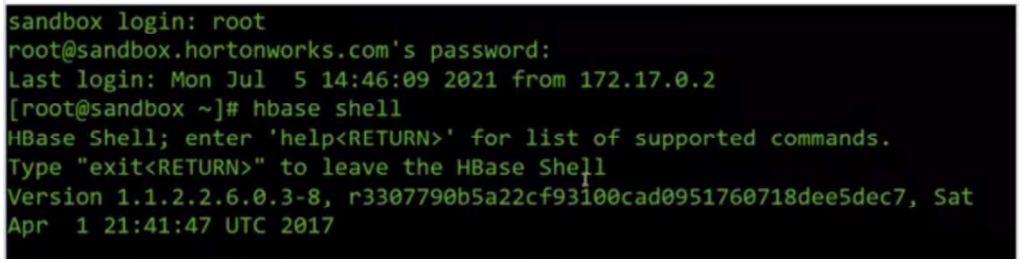


Command: **whichapplication-name** gives directory in which application-name is installed.

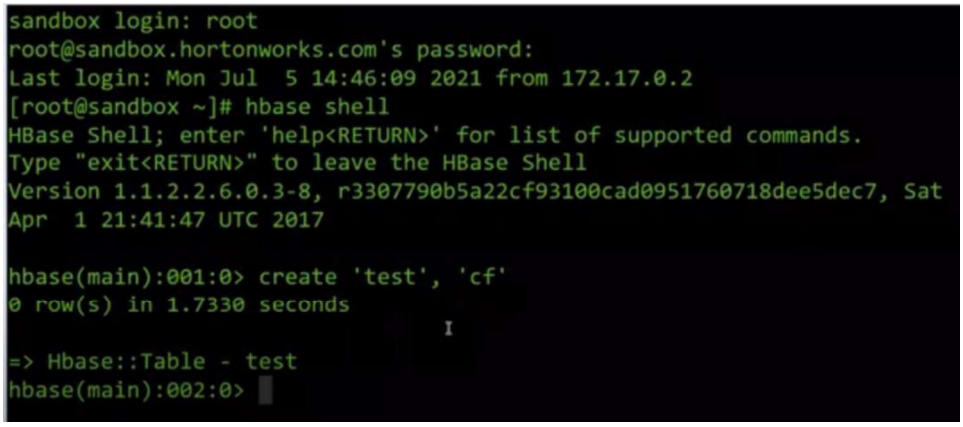
Open the shell
192.168.119.132:4200



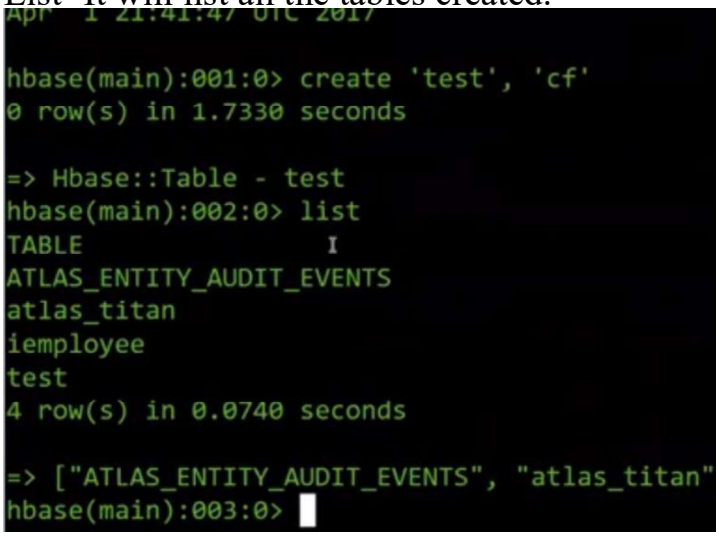
Command: hbase shell
It will start the server



Enter the command create ‘test’, ‘cf’ and it will create the table



Check the table is created with command
List- It will list all the tables created.



If we want to see column description of a table.

Command- describe tablename

```
hbase(main):003:0> describe 'test'
Table test is ENABLED
test
COLUMN FAMILIES DESCRIPTION
{NAME => 'cf', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'fals
e', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL =>
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE => 't
rue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.1950 seconds
```

Now, we have to put the values in table

Values:

put ‘test’, ‘row1’, ‘cf:a’, ‘value1’

put ‘test’, ‘row2’, ‘cf:b’, ‘value2’

put ‘test’, ‘row3’, ‘cf:c’, ‘value3’

copy paste the data in shell.

```
 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLO
rue', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
1 row(s) in 0.1950 seconds

hbase(main):004:0> put 'test', 'row1', 'cf:a', 'value1'
0 row(s) in 0.1930 seconds

hbase(main):005:0>
hbase(main):006:0* put 'test', 'row2', 'cf:b', 'value2'
0 row(s) in 0.0140 seconds

hbase(main):007:0>
hbase(main):008:0* put 'test', 'row3', 'cf:c', 'value3'
0 row(s) in 0.0340 seconds
```

We to display the records of table

Command: scan ‘test’

```
hbase(main):009:0> scan 'test'
ROW                   COLUMN+CELL
 row1                 column=cf:a, timestamp=1625496989589, value=value1
 row2                 column=cf:b, timestamp=1625496989697, value=value2
 row3                 column=cf:c, timestamp=1625496993087, value=value3
3 row(s) in 0.0620 seconds
```

Python: storage/retrieval

Start the service with command

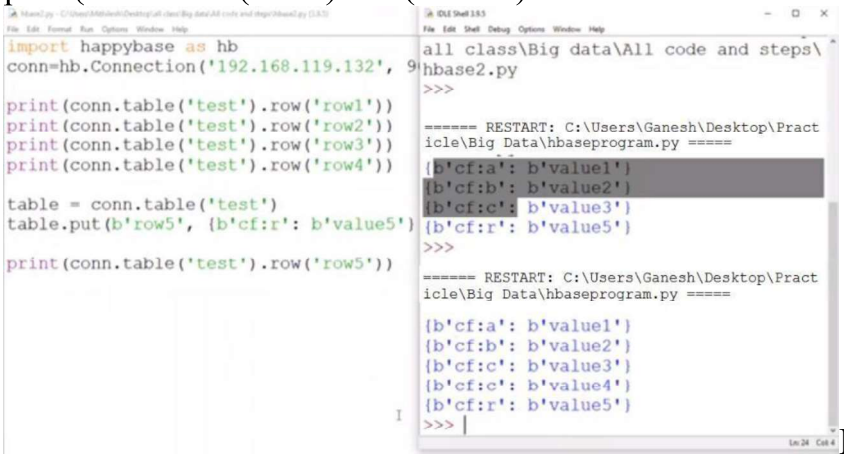
Hbase thrift start -p 9090 –inforport 9095

```
sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Tue Jul  6 13:22:05 2021 from 172.17.0.2
[root@sandbox ~]# hbase thrift start -p 9090 --infoport 9095
2021-07-06 14:52:38,870 INFO  [main] util.VersionInfo: HBase 1.1.2.2.6.0
.3-8
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: Source code repos
itory git://c66-slave-ff632c10-5/grid/0/jenkins/workspace/HDP-parallel-c
entos6/SOURCES/hbase revision=3307790b5a22cf93100cad0951760718dee5dec7
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: Compiled by jenki
ns on Sat Apr  1 21:41:47 UTC 2017
2021-07-06 14:52:38,873 INFO  [main] util.VersionInfo: From source with
checksum e816bb65a763f766331d511df40814e0
```

Create the table the way we did it in hbase and see the records using scan command

```
hbase(main):002:0> scan 'test'
ROW                   COLUMN+CELL
 row1                 column=cf:a, timestamp=1625496989589, value=value1
 row2                 column=cf:b, timestamp=1625496989697, value=value2
 row3                 column=cf:c, timestamp=1625496993087, value=value3
 row4                 column=cf:c, timestamp=1625545211781, value=value4
4 row(s) in 0.1960 seconds
```

Create a program file
Import happybase as hb
conn=hb.connection('192.168.119.132', 9090)
print(conn.table('test').row('row1')
print(conn.table('test').row('row2')
print(conn.table('test').row('row3')
print(conn.table('test').row('row4')
table = conn.table('test')
table.put(b'row5', {b'cf:r': b'value5'})
print(conn.table('test').row('row5')



Run a scan command on shell to display the values

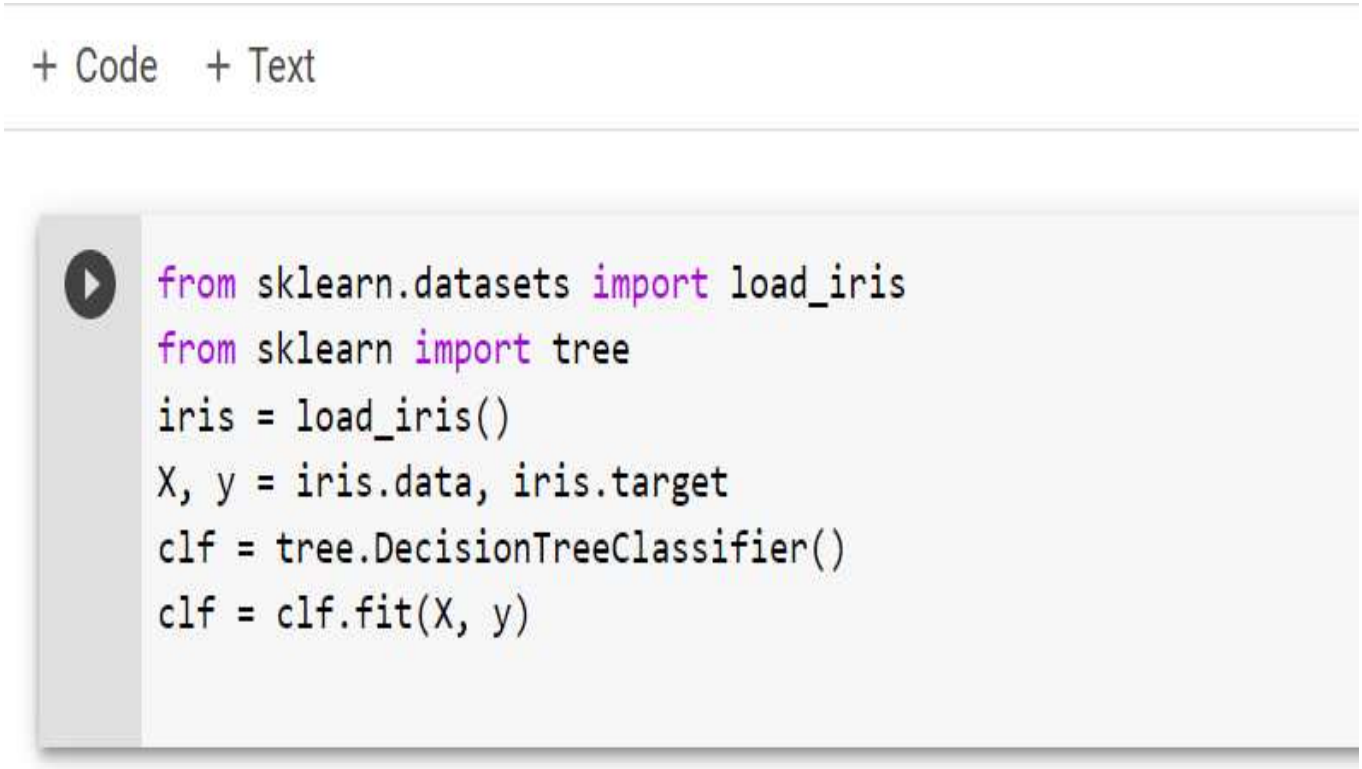
```
hbase(main):004:0> scan 'test'
ROW                   COLUMN+CELL
 row1                 column=cf:a, timestamp=1625496989589, value=value1
 row2                 column=cf:b, timestamp=1625496989697, value=value2
 row3                 column=cf:c, timestamp=1625496993087, value=value3
 row4                 column=cf:c, timestamp=1625545211781, value=value4
 row5                 column=cf:r, timestamp=1625583481042, value=value5
5 row(s) in 0.0320 seconds
```

Now, try with duplicate value at row 5 say value t

Practical 7

Aim: Implement Decision tree classification techniques.

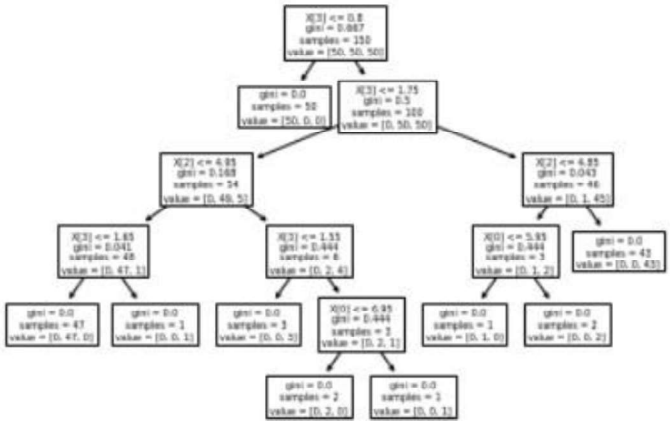
Implement Decision tree classification techniques **Decision Trees (DTs)** are a non- parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation. Using the Iris dataset, we can construct a tree as follows:



Once trained, we can plot the tree with the **plot tree** function:

```
[4]  tree.plot_tree(clf)
```

```
[Text(167.4, 199.32, 'X[3] <= 0.8\ngini = 0.667\nsamples = 150\nvalue = [50, 50,
 Text(141.64615384615385, 163.07999999999998, 'gini = 0.0\nsamples = 50\nvalue =
 Text(193.15384615384616, 163.07999999999998, 'X[3] <= 1.75\ngini = 0.5\nsamples
 Text(103.01538461538462, 126.83999999999999, 'X[2] <= 4.95\ngini = 0.168\nsampl
 Text(51.50769230769231, 90.6, 'X[3] <= 1.65\ngini = 0.041\nsamples = 48\nvalue
 Text(25.753846153846155, 54.359999999999985, 'gini = 0.0\nsamples = 47\nvalue =
 Text(77.26153846153846, 54.359999999999985, 'gini = 0.0\nsamples = 1\nvalue = [
 Text(154.52307692307693, 90.6, 'X[3] <= 1.55\ngini = 0.444\nsamples = 6\nvalue
 Text(128.76923076923077, 54.359999999999985, 'gini = 0.0\nsamples = 3\nvalue =
 Text(180.27692307692308, 54.359999999999985, 'X[0] <= 6.95\ngini = 0.444\nsampl
 Text(154.52307692307693, 18.119999999999976, 'gini = 0.0\nsamples = 2\nvalue =
 Text(206.03076923076924, 18.119999999999976, 'gini = 0.0\nsamples = 1\nvalue =
 Text(283.2923076923077, 126.83999999999999, 'X[2] <= 4.85\ngini = 0.043\nsample
 Text(257.53846153846155, 90.6, 'X[0] <= 5.95\ngini = 0.444\nsamples = 3\nvalue
 Text(231.7846153846154, 54.359999999999985, 'gini = 0.0\nsamples = 1\nvalue = [
 Text(283.2923076923077, 54.359999999999985, 'gini = 0.0\nsamples = 2\nvalue = [
 Text(309.04615384615386, 90.6, 'gini = 0.0\nsamples = 43\nvalue = [0, 0, 43]')]
```



Practical 8

Aim: Implement SVM classification techniques.

Code:

-

Support Vector Machines

Generally, Support Vector Machines is considered to be a classification approach, it but can be employed in both types of classification and regression problems. It can easily handle multiple continuous and categorical variables. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane (MMH) that best divides the dataset into classes. **Loading data:**

```
[1]  #Import scikit-learn dataset library
     from sklearn import datasets

     #Load dataset
     cancer = datasets.load_breast_cancer()
```

Exploring data:

```
# print the names of the 13 features
print("Features: ", cancer.feature_names)

# print the label type of cancer('malignant' 'benign')
print("Labels: ", cancer.target_names)
```

```
Features:  ['mean radius' 'mean texture' 'mean perimeter' 'mean area'
 'mean smoothness' 'mean compactness' 'mean concavity'
 'mean concave points' 'mean symmetry' 'mean fractal dimension'
 'radius error' 'texture error' 'perimeter error' 'area error'
 'smoothness error' 'compactness error' 'concavity error'
 'concave points error' 'symmetry error' 'fractal dimension error'
 'worst radius' 'worst texture' 'worst perimeter' 'worst area'
 'worst smoothness' 'worst compactness' 'worst concavity'
 'worst concave points' 'worst symmetry' 'worst fractal dimension']
Labels:  ['malignant' 'benign']
```

Check the shape of the dataset using shape.

```
# print data(feature)shape
cancer.data.shape
```

```
(569, 30)
```

Check top 5 records of the feature set.

```
[4]  # print the cancer data features (top 5 records)
     print(cancer.data[0:5])
```

```
[[1.799e+01 1.038e+01 1.228e+02 1.001e+03 1.184e-01 2.776e-01 3.001e-01
  1.471e-01 2.419e-01 7.871e-02 1.095e+00 9.053e-01 8.589e+00 1.534e+02
  6.399e-03 4.904e-02 5.373e-02 1.587e-02 3.003e-02 6.193e-03 2.538e+01
  1.733e+01 1.846e+02 2.019e+03 1.622e-01 6.656e-01 7.119e-01 2.654e-01
  4.601e-01 1.189e-01]
 [2.057e+01 1.777e+01 1.329e+02 1.326e+03 8.474e-02 7.864e-02 8.690e-02
  7.017e-02 1.812e-01 5.667e-02 5.435e-01 7.339e-01 3.398e+00 7.408e+01
  5.225e-03 1.308e-02 1.860e-02 1.340e-02 1.389e-02 3.532e-03 2.499e+01
  2.341e+01 1.588e+02 1.956e+03 1.238e-01 1.866e-01 2.416e-01 1.860e-01
  2.750e-01 8.902e-02]
 [1.969e+01 2.125e+01 1.300e+02 1.203e+03 1.096e-01 1.599e-01 1.974e-01
  1.279e-01 2.069e-01 5.999e-02 7.456e-01 7.869e-01 4.585e+00 9.403e+01
  6.150e-03 4.006e-02 3.832e-02 2.058e-02 2.250e-02 4.571e-03 2.357e+01
  2.553e+01 1.525e+02 1.709e+03 1.444e-01 4.245e-01 4.504e-01 2.430e-01
  3.613e-01 8.758e-02]
 [1.142e+01 2.038e+01 7.758e+01 3.861e+02 1.425e-01 2.839e-01 2.414e-01
  1.052e-01 2.597e-01 9.744e-02 4.956e-01 1.156e+00 3.445e+00 2.723e+01
  9.110e-03 7.458e-02 5.661e-02 1.867e-02 5.963e-02 9.208e-03 1.491e+01
  2.650e+01 9.887e+01 5.677e+02 2.098e-01 8.663e-01 6.869e-01 2.575e-01
  6.638e-01 1.730e-01]
 [2.029e+01 1.434e+01 1.351e+02 1.297e+03 1.003e-01 1.328e-01 1.980e-01
  1.043e-01 1.809e-01 5.883e-02 7.572e-01 7.813e-01 5.438e+00 9.444e+01
  1.149e-02 2.461e-02 5.688e-02 1.885e-02 1.756e-02 5.115e-03 2.254e+01
  1.667c+01 1.522c+02 1.575c+03 1.374c 01 2.050c 01 4.000c 01 1.625c 01
  2.364e-01 7.678e-02]]
```

Target set:

```
[5]  # print the cancer labels (0:malignant, 1:benign)
     print(cancer.target)
```

```
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0
 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1 1 0 1 1 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1 1
 1 1 1 1 1 1 1 0 0 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 1 1 0 1 1 0 1 1 1 1 0 1
 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 1 1 0 1 1 0 0 0 1 0
 1 0 1 1 1 0 1 1 0 0 1 0 0 0 0 1 0 0 0 1 0 1 0 1 1 0 1 0 0 0 0 1 1 0 0 1 1
 1 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 0 1 1 1 1 0 1 1 1 1 1 0 1 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 1 1 0 0 0 1 1
 1 1 0 1 0 1 0 1 1 1 0 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0
 0 1 0 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 0 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1
 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1 1
 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 0 1
 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0
 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 0 0 0 0 0 0 1]
```

Splitting Data:

To understand model performance, dividing the dataset into a training set and a test set is a good strategy.

Split the dataset by using the function train_test_split (). you need to pass 3 parameters features, target, and test set size. Additionally, you can use random state to select records randomly.

```
[7] # Import train_test_split function
    from sklearn.model_selection import train_test_split

    # Split dataset into training set and test set
    X_train, X_test, y_train, y_test = train_test_split(cancer.data, cancer.target, test_size=0.3,random_state=109) # 70% training and 30% test
```

Generate Model:

Let's build support vector machine model. First, import the SVM module and create support vector classifier object by passing argument kernel as the linear kernel in SVC () function.

Then, fit your model on train set using fit () and perform prediction on the test set using predict().

```
[8] #Import svm model
    from sklearn import svm

    #Create a svm Classifier
    clf = svm.SVC(kernel='linear') # Linear Kernel

    #Train the model using the training sets
    clf.fit(X_train, y_train)

    #Predict the response for test dataset
    y_pred = clf.predict(X_test)
```

Evaluating the Model:

Let's estimate how accurately the classifier or model can predict the breast cancer of patients. Accuracy can be computed by comparing actual test set values and predicted values.

```
#Import scikit-learn metrics module for accuracy calc
from sklearn import metrics

# Model Accuracy: how often is the classifier correct
print("Accuracy:",metrics.accuracy_score(y_test, y_pr
```

```
Accuracy: 0.9649122807017544
```

Practical No 9

Aim: To implement REGRESSION MODEL(Linear & Logistical Regression)

Software: Python editor

Packages used: numpy, pandas, sklearn

Description:

Linear Regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a straight line to the data. It assumes a linear relationship and is commonly used for predicting continuous values. The model learns by minimizing the difference between predicted and actual values using metrics such as Root Mean Squared Error (RMSE) and R-squared score (R^2), which measure the accuracy and variance explained by the model.

Logistic Regression is a classification algorithm used to predict categorical outcomes, typically binary (e.g., yes/no, 0/1). Instead of fitting a straight line, it uses a logistic (sigmoid) function to estimate probabilities, mapping values between 0 and 1. The model predicts class labels based on a threshold, commonly 0.5. It is evaluated using metrics such as accuracy, precision, recall, and F1-score, which assess the model's ability to distinguish between classes effectively.

Code:

Linear Regression

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
# Generate Sample Data
np.random.seed(42)
X = 2 * np.random.rand(100, 1)
y = 4 + 3 * X + np.random.randn(100, 1) # Linear relation with some noise
# Convert to DataFrame
df = pd.DataFrame(np.hstack((X, y)), columns=['Feature', 'Target'])
# Split Data into Training and Testing Sets
```

```
X_train, X_test, y_train, y_test = train_test_split(df[['Feature']], df['Target'],
test_size=0.2, random_state=42)
# Initialize and Train Linear Regression Model
model = LinearRegression()
model.fit(X_train, y_train)
# Make Predictions
y_pred = model.predict(X_test)
# Evaluate the Model
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)
# Print Results
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
print("Root Mean Squared Error (RMSE):", rmse)
print("R-squared Score (R2):", r2)
```

Output:

```
Coefficients: [2.79932366]
Intercept: 4.142913319458566
Root Mean Squared Error (RMSE): 0.8085168605026132
R-squared Score (R2): 0.8072059636181392
```

Logistical Regression Code

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

```
# Generate Sample Data
np.random.seed(42)
X = 2 * np.random.rand(100, 1)
y = (X > 1).astype(int).ravel() # Binary classification based on threshold
# Convert to DataFrame
df = pd.DataFrame(np.hstack((X, y.reshape(-1, 1))), columns=['Feature', 'Target'])
# Split Data into Training and Testing Sets
X_train, X_test, y_train, y_test = train_test_split(df[['Feature']], df['Target'],
test_size=0.2, random_state=42)
# Initialize and Train Logistic Regression Model
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
# Make Predictions
y_pred = model.predict(X_test)
# Evaluate the Model
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
# Print Results
print("Coefficients:", model.coef_)
```

```
print("Intercept:", model.intercept_)
print("Accuracy:", accuracy)
print("Classification Report:\n", report)
```

Output

```
Coefficients: [[3.85477457]]
Intercept: [-3.75556844]
Accuracy: 1.0
Classification Report:
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00        11
         1.0       1.00      1.00      1.00         9

    accuracy                           1.00        20
   macro avg       1.00      1.00      1.00        20
weighted avg       1.00      1.00      1.00        20
```

Practical No. 10

Aim: To implement Classification Model

Software: Python editor

Packages used: numpy, pandas, sklearn

Description:

A classification model using the Random Forest Classifier on the Iris dataset is used. It first loads the dataset and converts it into a Pandas DataFrame, where the features represent different measurements of iris flowers, and the target variable indicates the species. The data is split into training and testing sets, with 80% used for training and 20% for evaluation. A Random Forest Classifier with 100 decision trees is trained on the dataset, and predictions are made on the test set. The model's performance is assessed using accuracy, confusion matrix, and classification report, which provide insights into the model's correctness and error distribution.

A classification model is a supervised machine learning model that categorizes input data into predefined classes or labels. It learns patterns from training data and uses them to classify new observations. Classification models are widely used in real-world applications such as spam detection, medical diagnosis, and image recognition.

Code:

Classification Model

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.datasets import load_iris

# Load Dataset
```

```
data = load_iris()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['Target'] = data.target


# Split Data into Training and Testing Sets
X = df.drop(columns=['Target'])
y = df['Target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)


# Initialize and Train Classifier
classifier = RandomForestClassifier(n_estimators=100, random_state=42)
classifier.fit(X_train, y_train)


# Make Predictions
y_pred = classifier.predict(X_test)


# Evaluate the Model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
report = classification_report(y_test, y_pred)


# Print Results
print("Accuracy:", accuracy)
print("Confusion Matrix:\n", conf_matrix)
print("Classification Report:\n", report)
```

Output:

Accuracy: 1.0

Confusion Matrix:

[[10 0 0]

[0 9 0]

[0 0 11]]

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

Practical No. 11

Aim: To implement Clustering Model

Software: Python editor

Packages used: numpy, pandas, sklearn

Description:

K-Means clustering is applied with **Iris dataset**, grouping the data into three clusters based on feature similarities. It first loads the dataset and uses **K-Means**, a widely used unsupervised learning algorithm, to categorize data points into three clusters. To visualize the clusters, **Principal Component Analysis (PCA)** is used to reduce the dataset to two dimensions. The results are plotted using **Seaborn** and **Matplotlib**, where each cluster is represented with a different color. This helps in understanding how the algorithm groups similar data points together based on patterns in the dataset.

A **clustering model** is an **unsupervised learning** technique that groups data points into clusters based on their similarities without predefined labels. It is used in applications such as customer segmentation, anomaly detection, and image segmentation. K-Means is a popular clustering algorithm that partitions data into K clusters by minimizing the distance between points and their respective cluster centroids.

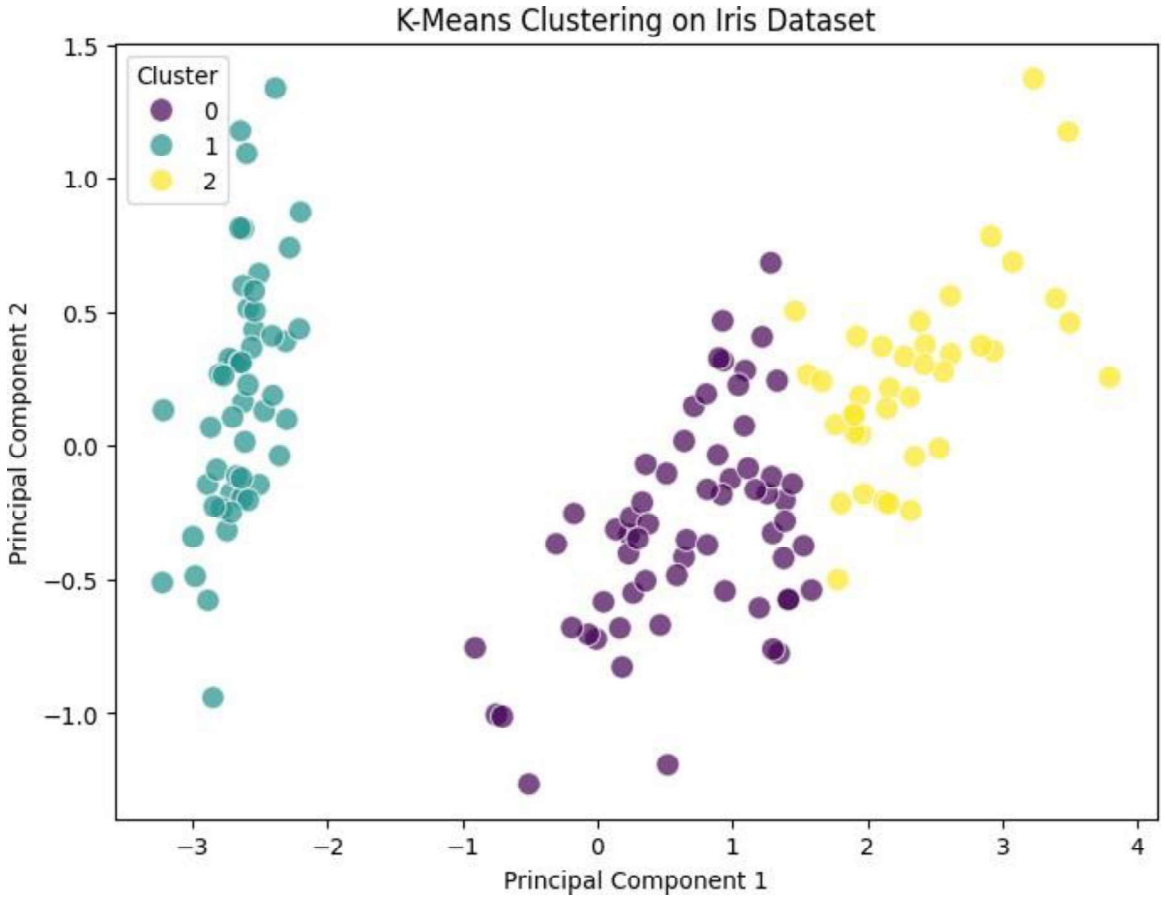
Code:

Clustering Model

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
```

```
# Load Dataset
data = load_iris()
df = pd.DataFrame(data.data, columns=data.feature_names)

# Apply K-Means Clustering
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
kmeans.fit(df)
df['Cluster'] = kmeans.labels_

# Reduce Dimensions for Visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(df[data.feature_names])
df['PCA1'] = X_pca[:, 0]
df['PCA2'] = X_pca[:, 1]

# Plot Clusters
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df['PCA1'], y=df['PCA2'], hue=df['Cluster'], palette='viridis', s=100,
alpha=0.7)
plt.title("K-Means Clustering on Iris Dataset")
plt.xlabel("Principal Component 1")
plt.ylabel("Principal Component 2")
plt.legend(title="Cluster")
plt.show()
```

Output:


K-Means Clustering on Iris Dataset
Cluster
0
1
2
Principal Component 2
Principal Component 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−3
−2
−1
0
1
2
3
4

BUNTS SANGHA MUMBAI

ANNA LEELA COLLEGE OF COMMERCE & ECONOMICS

SHOBHA JAYARAM SHETTY COLLEGE FOR BMS

(UNIVERSITY OF MUMBAI)

Shashi Manmohan Shetty Higher Education Complex,

Buntara Bhavan Marg, Kurla (E), Mumbai – 400 070.

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Exam. Seat No. _____

Date _____

CERTIFIED that the practical, and assignments duly signed, were performed by

Mr./Ms. _____

Roll No. _____ *of* _____*class in the Information Technology Laboratory of Anna Leela College of Commerce & Economics Shobha Jayaram Shetty College for BMS, Mumbai during the academic year 20* _____ *-20* _____

He/she has completed the course of laboratory assignments in Information Technology as contained in the course prescribed by the University of Mumbai.

Head of Dept.

Information Technology

Date _____

Sign. of the student

Date _____

Professor-in-charge

1) _____

Date _____

Sign. of Examiner's

1) _____

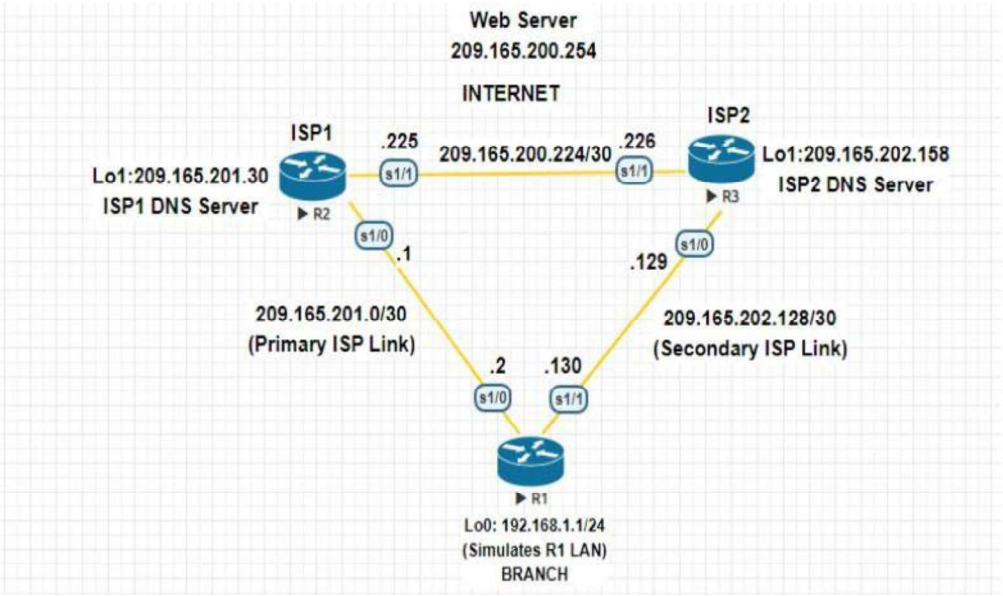
Date _____

Sr. No	PRACTICAL	DATE	SIGN
1	Configure IP SLA Tracking and Path Control Topology.		
2	Using the AS_PATH Attribute.		
3	Configuring IBGP and EBGP Sessions, Local Preference, and MED.		
4	Secure the Management Plane.		
5	Configure and Verify Path Control Using PBR.		
6	IP Service Level Agreements and Remote SPAN in a Campus Environment.		
7	Inter-VLAN Routing.		
8	Simulating an MPLS environment and Simulating VRF.		
9	Simulating SDN with • Open Daylight SDN Controller with the Mininet Network Emulator. • OF Net SDN network emulator.		
10	Simulating OpenFlow Using MININET.		

Practical 1

Aim: - Configure IP SLA Tracking and Path Control Topology.

NETWORK TOPOLOGY



R1

```
Router>enable
Router# conf t R
outer(config)#hostname R1
 R1(config)#interface Loopback 0
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#exit
R1(config)#interface s1/0
R1(config-if)#ip address 209.165.201.2 255.255.255.252
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface s1/1
R1(config-if)#ip address 209.165.202.130 255.255.255.252
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 209.165.201.1
R1(config)#ip sla 12
R1(config-ip-sla)#icmp-echo 209.165.201.30
R1(config-ip-sla-echo)#frequency 11
R1(config-ip-sla-echo)#exit
R1(config)#ip sla schedule 12 life forever start-time now R

->1#sh ip sla configuration 12
 IP SLAs Infrastructure Engine-III
Entry number: 12
Owner:
```

Tag:
Operation timeout (milliseconds): 5000
Type of operation to perform: icmp-echo
Target address/Source address: 209.165.201.30/0.0.0.0
Type Of Service parameter: 0x0
Request size (ARR data portion): 28
Verify data: No
Vrf Name:
Schedule:
Operation frequency (seconds): 11 (not considered if randomly scheduled)
Next Scheduled Start Tim
e: Start Time already passed
Group Scheduled : FALSE Randomly Scheduled : FALSE
Life (seconds): Forever
Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE
Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 5000
Distribution Statistics:
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1
Statistic distribution interval (milliseconds): 20
Enhanced History:
History Statistics:
Number of history Lives kept: 0
Number of history Buckets kept: 15
History Filter Type: None R1#sh ip sla statistics
IPSLAs Latest Operation Statistics
IPSLA operation id: 12
Latest RTT: 11 milliseconds
Latest operation start time: 18:21:25 EET Thu Apr 9 2020
Latest operation return code: OK
Number of successes: 22
Number of failures: 0
Operation time to live: Forever
R1(config)#ip sla 24
R1(config-ip-sla)#icmp-echo 209.165.202.158
R1(config-ip-sla-echo)#frequency 10
R1(config-ip-sla-echo)#exit
R1(config)#ip sla schedule 24 life forever start-time now
R1#sh ip sla configuration 24
IP SLAs Infrastructure Engine-III
Entry number: 24
Owner:
Tag:
Operation timeout (milliseconds): 5000
Type of operation to perform: icmp-echo
Target address/Source address: 209.165.202.158/0.0.0.0
Type Of Service parameter: 0x0

Request size (A
RR data portion): 28
Verify data: No Vrf Name:
Schedule:
Operation frequency (seconds): 10 (not considered if randomly scheduled)
Next Scheduled Start Time: Start Time already passed
Group Scheduled : FALSE
Randomly Scheduled : FALSE
Life (seconds): Forever
Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE
Status of entry (SNMP RowStatus): Active
Threshold (milliseconds): 5000
Distribution Statistics:
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1
Statistic distribution interval (milliseconds): 20
Enhanced History:
History Statistics:
Number of history Lives kept: 0
Number of history Buckets kept: 15
History Filter Type: None
R1#sh ip sla statistics 24
IPSLAs Latest Operation Statistics
IPSLA operation id: 24 Latest RTT: 20 milliseconds
Latest operation start time: 18:33:25 EET Thu Apr 9 2020
Latest operation return code: OK
Number of successes: 16
Number of failures: 0
Operation time to live: Forever
R1(config)#no ip route 0.0.0.0 0.0.0.0 209.165.201.1
R1(config)#ip route 0.0.0.0 0.0.0.0 209.165.201.1 5
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 209.165.201.1 to network 0.0.0.0
S* 0.0.0.0/0 [5/0] via 209.165.201.1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback0
L 192.168.1.1/32 is directly connected, Loopback0 209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.201.0/30 is directly connected, Serial1/0
L 209.165.201.2/32 is directly connected, Serial1/0 209.165.202.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.202.128/30 is directly connected, Serial1/1
L 209.165.202.130/32 is directly connected, Serial1/1

R1(config)#track 1 ip sla 12 reachability
R1(config-track)#delay down 10 up 1
R1(config-track)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 209.165.201.1 2 track 1
R1(config)#track 2 ip sla 12 reachability
R1(config-track)#delay down 10 up 1
R1(config-track)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 209.165.201.1 3 track 2
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 209.165.201.1 to network 0.0.0.0
S* 0.0.0.0/0 [3/0] via 209.165.201.1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback0
L 192.168.1.1/32 is directly connected, Loopback0 209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.201.0/30 is directly connected, Serial1/0
L 209.165.201.2/32 is directly connected, Serial1/0 209.165.202.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.202.128/30 is directly connected, Serial1/1
L 209.165.202.130/32 is directly connected, Serial1/1
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 209.165.201.1 to network 0.0.0.0
S* 0.0.0.0/0 [5/0] via 209.165.201.1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback0
L 192.168.1.1/32 is directly connected, Loopback0 209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.201.0/30 is directly connected, Serial1/0
L 209.165.201.2/32 is directly connected, Serial1/0 209.165.202.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.202.128/30 is directly connected, Serial1/1
L 209.165.202.130/32 is directly connected, Serial1/1
R1#sh ip sla statistics
IPSLAs Latest Operation Statistics
IPSLA operation id: 12
Latest RTT: NoConnection/Busy/Timeout
Latest operation start time: 19:02:29 EET Thu Apr 9 2020
Latest operation return code: Timeout
Number of successes: 227

Number of failures: 19
Operation time to live: Forever
IPSLA operation id: 24
Latest RTT: 20 milliseconds
Latest operation start time: 19:02:35 EET Thu Apr 9 2020
Latest operation return code: OK
Number of successes: 190
Number of failures: 1
Operation time to live: Forever
R1#trace 209.165.200.254 source 192.168.1.1
Type escape sequence to abort.
Tracing the route to 209.165.200.254
VRF info: (vrf in name/id, vrf out name/id)
1 209.165.201.1 10 msec 14 msec *
R1#sh ip sla statistics
IPSLAs Latest Operation Statistics
IPSLA operation id: 12
Latest RTT: 10 milliseconds
Latest operation start time: 19:07:04 EET Thu Apr 9 2020
Latest operation return code: OK
Number of successes: 236
Number of failures: 35
Operation time to live: Forever
IPSLA operation id: 24 Latest RTT: 21 milliseconds
Latest operation start time: 19:07:05 EET Thu Apr 9 2020
Latest operation return code: OK
Number of successes: 217
Number of failures: 1
Operation time to live: Forever
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 209.165.201.1 to network 0.0.0.0
S* 0.0.0.0/0 [3/0] via 209.165.201.1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback0
L 192.168.1.1/32 is directly connected, Loopback0 209.165.201.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.201.0/30 is directly connected, Serial1/0
L 209.165.201.2/32 is directly connected, Serial1/0 209.165.202.0/24 is variably subnetted, 2 subnets, 2 masks
C 209.165.202.128/30 is directly connected, Serial1/1
L 209.165.202.130/32 is directly connected, Serial1/1

ISP1 (R2)
Router>enable

```
Router#conf t
Router(config)#hostname ISP1
ISP1(config)#interface Loopback0
 ISP1(config-if)#description Simulated Internet Web Server
ISP1(config-if)#ip address 209.165.200.254 255.255.255.255
ISP1(config-if)#exit
ISP1(config)#interface Loopback1
ISP1(config-if)#ip address 209.165.201.30 255.255.255.255
 ISP1(config-if)#exit
 ISP1(config)#interface s1/0
 ISP1(config-if)#ip address 209.165.201.1 255.255.255.252
ISP1(config-if)#no shutdown
ISP1(config-if)#exit
 ISP1(config)#interface s1/1
 ISP1(config-if)#ip address 209.165.200.225 255.255.255.252
 ISP1(config-if)#no shutdown
 ISP1(config-if)#exit
ISP1(config)#router eigrp 200
 ISP1(config-router)#network 209.165.200.224
ISP1(config-router)#network 209.165.201.0
ISP1(config-router)#no auto-summary
ISP1(config-router)#exit
ISP1(config)#ip route 192.168.1.0 255.255.255.0 209.165.201.2
 ISP1(config)#interface loopback 1
 ISP1(config-if)#shut
 ISP1(config)#interface loopback 1
 ISP1(config-if)#no shutdown
```

ISP2 (R3)

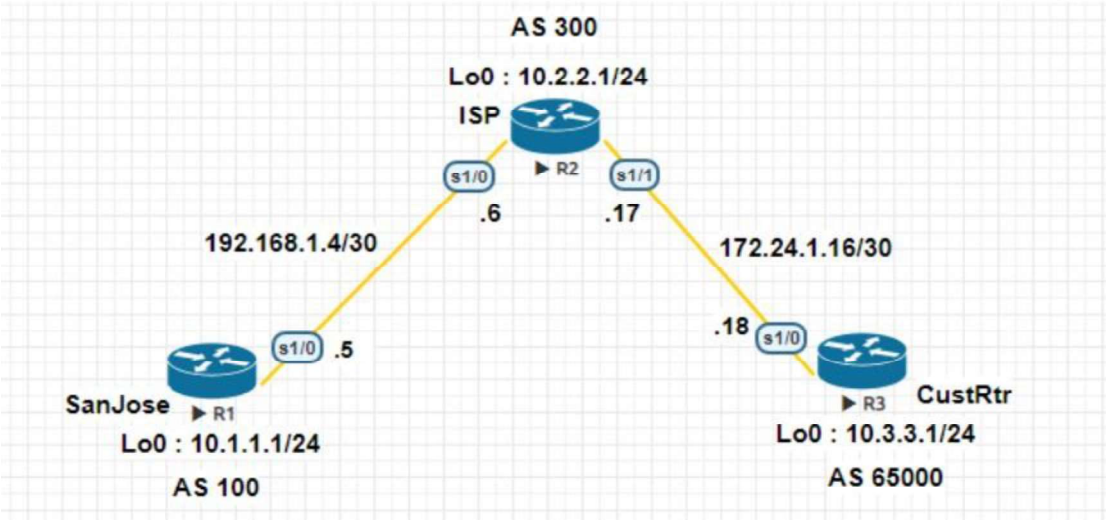
```
 Router>enable
Router#conf t
Router(config)#hostname ISP2
ISP2(config)#interface Loopback0
ISP2(config-if)#description Simulated Internet Web Server
ISP2(config-if)#ip address 209.165.200.254 255.255.255.255
ISP2(config-if)#exit
ISP2(config)#interface Loopback1
ISP2(config-if)#ip address 209.165.202.158 255.255.255.255
ISP2(config-if)#exit
 ISP2(config)#interface s1/1
ISP2(config-if)#ip address 209.165.200.226 255.255.255.252
ISP2(config-if)#no shutdown
 ISP2(config-if)#exit ISP2(config)#interface s1/0
 ISP2(config-if)#ip address 20
9.165.202.129 255.255.255.252
ISP2(config-if)#no shutdown
```

Practical 2

Aim: - Using the AS_PATH Attribute.

Code: -

NETWORK TOPOLOGY



SanJose

```
Router>enable
Router#conf t
Router(config)#hostname SanJose
SanJose(config)#interface Loopback0
SanJose(config-if)#ip address 10.1.1.1 255.255.255.0
SanJose(config-if)#exit
SanJose(config)#interface Serial1/0
SanJose(config-if)#ip address 192.168.1.5 255.255.255.252
SanJose(config-if)#no shutdown
SanJose(config-if)#end
SanJose(config)#router bgp 100
SanJose(config-router)#network 10.1.1.0 mask 255.255.255.0
SanJose(config-router)#neighbor 192.168.1.6 remote-as 300
SanJose(config-router)#exit
SanJose#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C 10.1.1.0/24 is directly connected, Loopback0
```

L 10.1.1.1/32 is directly connected, Loopback0
B 10.2.2.0/24 [20/0] via 192.168.1.6, 00:05:47
B 10.3.3.0/24 [20/0] via 192.168.1.6, 00:02:13 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.4/30 is directly connected, Serial1/0
L 192.168.1.5/32 is directly connected, Serial1/0
SanJose#sh ip bgp
BGP table version is 4, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete RPKI validation codes: V valid, I invalid, N Not found

Network	Next Hop	Metric	LocPrf Weight Path
*> 10.1.1.0/24	0.0.0.0	0	32768 i
*> 10.2.2.0/24	192.168.1.6	0	0 300 i
*> 10.3.3.0/24	192.168.1.6		0 300 65000 i

SanJose#sh ip bgp
BGP table version is 5, local router ID is 10.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete RPKI validation codes: V valid, I invalid, N Not found

Network	Next Hop	Metric	LocPrf Weight Path
*> 10.1.1.0/24	0.0.0.0	0	32768 i
*> 10.2.2.0/24	192.168.1.6	0	0 300 i
*> 10.3.3.0/24	192.168.1.6		0 300 i

ISP Router>enable

Router#conf t
Router(config)#hostname ISP
ISP(config)#interface Loopback0
ISP(config-if)#ip address 10.2.2.1 255.255.255.0
ISP(config-if)#exit ISP(config)#interface Serial1/0
ISP(config-if)#ip address 192.168.1.6 255.255.255.252 I
SP(config-if)#no shutdown
ISP(config-if)#exit
ISP(config)#interface Serial1/1
ISP(config-if)#ip address 172.24.1.17 255.255.255.252
ISP(config-if)#no shutdown
ISP(config-if)#end
ISP(config)#router bgp 300
ISP(config-router)#network 10.2.2.0 mask 255.255.255.0
ISP(config-router)#neighbor 192.168.1.5 remote-as 100
ISP(config-router)#neighbor 172.24.1.18 remote-as 65000
ISP(config)#router bgp 300
ISP(config-router)#neighbor 192.168.1.5 remove-private-as
ISP(config-router)#end

ISP#clear ip bgp * soft
ISP(config)#ip as-path access-list 1 deny ^100$ ISP(config)#ip as-path access-list 1 permit .*
ISP(config)#router bgp 300
ISP(config-router)#neighbor 172.24.1.18 filter-list 1 out
ISP(config-router)#end
ISP#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
B 10.1.1.0/24 [20/0] via 192.168.1.5, 00:46:41
C 10.2.2.0/24 is directly connected, Loopback0
L 10.2.2.1/32 is directly connected, Loopback0
B 10.3.3.0/24 [20/0] via 172.24.1.18, 00:43:07 172.24.0.0/16 is variably subnetted, 2 subnets, 2 masks
C 172.24.1.16/30 is directly connected, Serial1/1
L 172.24.1.17/32 is directly connected, Serial1/1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.4/30 is directly connected, Serial1/0
L 192.168.1.6/32 is directly connected, Serial1/0
ISP#show ip bgp regexp ^100$
BGP table version is 4, local router ID is 10.2.2.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

Network	Next Hop	Metric	LocPrf Weight Pat
h *> 10.1.1.0/24	192.168.1.5	0	0 100 i

CustRtr

Router>enable
Router#conf t
Router(config)#hostname CustRtr
CustRtr(config)#interface Loopback0
CustRtr(config-if)#ip address 10.3.3.1 255.255.255.0
CustRtr(config-if)#exit
CustRtr(config)#interface Serial1/0
CustRtr(config-if)#ip address 172.24.1.18 255.255.255.252
CustRtr(config-if)#no shutdown
CustRtr(config-if)#end
CustRtr(config)#router bgp 65000
CustRtr(config-router)#network 10.3.3.0 mask 255.255.255.0
CustRtr(config-router)#neighbor 172.24.1.17 remote-as 30

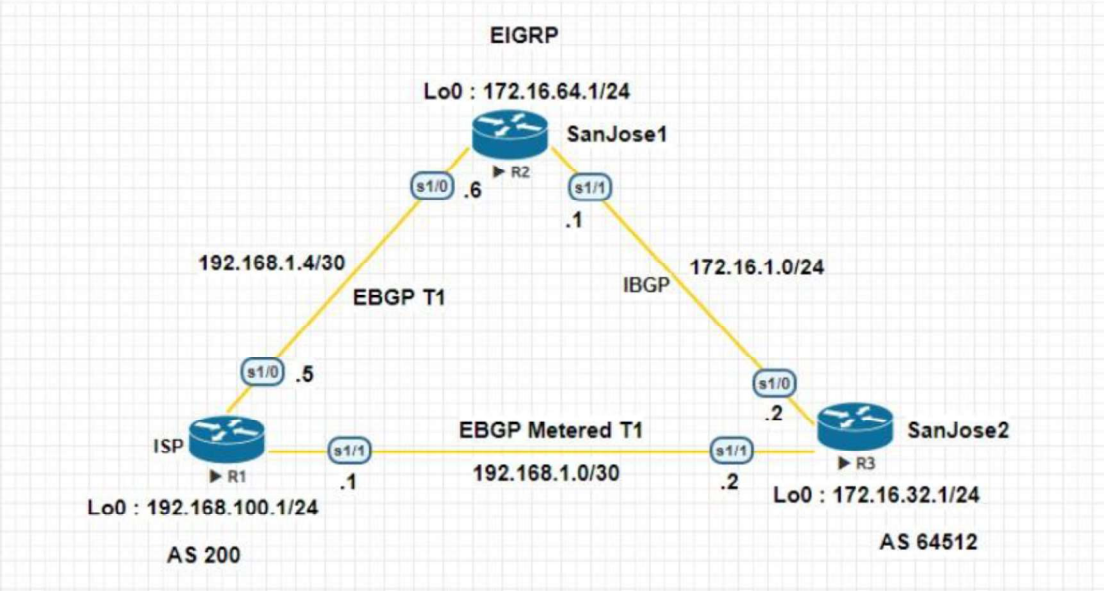
0 CustRtr(config-router)#end
CustRtr#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks
B 10.2.2.0/24 [20/0] via 172.24.1.17, 00:45:59
C 10.3.3.0/24 is directly connected, Loopback0
L 10.3.3.1/32 is directly connected, Loopback0 172.24.0.0/16 is variably subnetted, 2 subnets, 2 mask
s C 172.24.1.16/30 is directly connected, Serial1/0
L 172.24.1.18/32 is directly connected, Serial1/0

Practical 3

Aim: - Configuring IBGP and EBGP Sessions, Local Preference, and MED.

Code: -

NETWORK TOPOLOGY



R1(ISP)

```
Router>enable
Router#conf t
Router(config)#hostname ISP
ISP(config)#interface Loopback0
ISP(config-if)#ip address 192.168.100.1 255.255.255.0
 ISP(config-if)#exit
 ISP(config)#interface Serial1/0
ISP(config-if)#ip address 192.168.1.5 255.255.255.252
 ISP(config-if)#no shutdown
ISP(config-if)#exit
ISP(config)#interface Serial1/1
ISP(config-if)#ip address 192.168.1.1 255.255.255.252
ISP(config-if)#no shutdown
 ISP(config-if)#exit
ISP(config)#router bgp 200
 ISP(config-router)#network 192.168.100.0
ISP(config-router)#neighbor 192.168.1.6 remote-as 64512
ISP(config-router)#neighbor 192.168.1.2 remote-as 64512
 ISP(config-router)#exit
ISP#sh ip bgp
 BGP table version is 3, local router ID is 192.168.100.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S
Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-
compressed,
Origin codes: i - IGP, e - EGP, ? – incomplete
```

```
RPKI validation codes: V valid, I invalid, N Not found
Network          Next Hop        Metric     LocPrf
Weight Path
* 172.16.0.0     192.168.1.2     0          0 64512
i
*> 192.168.1.6   0               0          64512
i
*> 192.168.100.0 0.0.0.0         0          32768 i
ISP#ping 172.16.1.1 source 192.168.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.1.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.100.1 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 10/10/11 ms
ISP#ping 172.16.32.1 source 192.168.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.32.1, timeout is 2 seconds:
Packet sent with a source address of 192.168.100.1 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 15/15/16 ms
ISP#ping 172.16.1.2 source 192.
168.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.1.2, timeout is 2 seconds:
Packet sent with a source address of 192.168.100.1 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 15/17/25 ms
ISP(config)#router bgp 200
ISP(config-router)#network 192.168.1.0 mask 255.255.255.252
ISP(config-router)#network 192.168.1.4 mask 255.255.255.252
ISP(config-router)#exit
ISP#sh ip bgp
BGP table version is 5, local router ID is 192.168.100.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S
Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-
compressed,
Origin codes: i - IGP, e - EGP, ? – incomplete
RPKI validation codes: V valid, I invalid, N Not found
Network          Next Hop        Metric     LocPrf
Weight Path * 172.16.0.0     192.168.1.6     0
0 64512 i
*> 192.168.1.2   0               0
64512
i *> 192.168.1.0/30 0.0.0.0      0
32768 i
*> 192.168.1.4/30 0.0.0.0        0
32768 i
*> 192.168.100.0 0.0.0.0         0
32768 i
ISP#sh ip bgp
BGP table version is 6, local router ID is 192.168.100.1
```

Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed,
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

Network	Next Hop	Metric	LocPrf	Weight	Path
*> 172.16.0.0	192.168.1.6	50		0	64512 i
* 192.168.1.2	75	0			64512 i
*> 192.168.1.0/30	0.0.0.0	0			32768 i
*> 192.168.1.4/30	0.0.0.0	0			32768 i
*> 192.168.100.0	0.0.0.0	0			32768 i

ISP#ping 172.16.1.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.1.1, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 9/10/11 ms
ISP#ping 172.16.1.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.16.1.2, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 20/21/25 ms
ISP#traceroute 172.16.1.1
Type escape sequence to abort.
Tracing the route to 172.16.1.1
VRF info: (vrf in name/id, vrf out name/id)
1 192.168.1.6 10 msec 10 msec *
ISP#traceroute 172.16.1.2
Type escape sequence to abort.
Tracing the route to 172.16.1.2
VRF info: (vrf in name/id, vrf out name/id)
1 192.168.1.6 10 msec 10 msec 13 msec
2 172.16.1.2 [AS 64512] 20 msec 19 msec

* R2 (SanJose1)

Router>enable Router#conf t
Router(config)#hostname SanJose1
SanJose1(config)#interface Loopback0
SanJose1(config-if)#ip address 172.16.64.1 255.255.255.0
SanJose1(config-if)#ip address 172.16.64.1 255.255.255.0
SanJose1(config-if)#exit
SanJose1(config)#interface Serial1/0
SanJose1(config-if)#ip address 192.168.1.6 255.255.255.252
SanJose1(config-if)#no shutdown
SanJose1(config-if)#exit
SanJose1(config)#interface Serial1/1
SanJose1(config-if)#ip address 172.16.1.1 255.255.255.0

```
SanJose1(config-if)#no shutdown
SanJose1(config-if)#exit
SanJose1(config)#router eigrp 64512
SanJose1(config-router)#network 172.16.0.0
SanJose1(config-router)#no auto-summary
SanJose1(config-router)#exit
SanJose1(config)#router bgp 64512
 SanJose1(config-router)#neighbor 172.16.32.1 remote-as 64512
 SanJose1(config-router)#neighbor 172.16.32.1 update-source loopback0
 SanJose1(config-router)#exit
SanJose1(config)#ip route 172.16.0.0 255.255.0.0 null 0
SanJose1(config)#router bgp 64512
SanJose1(config-router)#network 172.16.0.0
 SanJose1(config-router)#neighbor 192.168.1.5 remote-as 200 S
anJose1(config-router)#exit
SanJose1(config)#router bgp 64512
 SanJose1(config-router)#neig
hbor 172.16.32.1 next-hop-self
SanJose1(config-router)#exit
SanJose1#sh ip bgp
 BGP table version is 5, local router ID is 172.16.64.1
 Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S
Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-
compressed, Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found
Network                Next Hop            Metric                LocPrf
Weight Path * i 172.16.0.0          172.16.32.1              0
100 0 i
 *> 0.0.0.0                 0                      32768 i
* i 192.168.1.0/30 172.16.32.1         0                    100                      0 200 i
 *> 192.168.1.5 0 0 200 i r i 192.168.1.4/30 172.16.32.1 0 100  0                200 i
   r> 192.168.1.5 0 0 200 i
 * i 192.168.100.0 172.16.32.1         0                    100                      0 200
i
*> 192.168.1.5                0                      0                              200
i
```

```
SanJose1(config)#route-map PRIMARY_T1_IN permit 10
SanJose1(config-route-map)#set local-preference 160
SanJose1(config-route-map)#exit
SanJose1(config)#router bgp 64512
SanJose1(config-router)#neighbor 192.168.1.5 route-map PRIMARY_T1_IN in
 SanJose1(config-router)#exit
 SanJose1#clear ip bgp * soft
 SanJose1#sh ip bgp
```

```
BGP table version is 8, local router ID is 172.16.64.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S
Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-
compressed, Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found
 Network                Next Hop            Metric     LocPrf
Weight Path * i 172.16.0.0          172.16.32.1              0
100 0 i
*> 0.0.0.0                0
32768
i *> 192.168.1.0/30    192.168.1.5              0             160 0
200 i
r> 192.168.1.4/30 192.168.1.5      0                  160                0
200 i
 *> 192.168.100.0           192.168.1.5               0
160 0 200 i
```

```
SanJose1(config)#route-map PRIMARY_T1_MED_OUT permit 10
SanJose1(config-route-map)#set Metric 50
SanJose1(config-route-map)#exit
SanJose1(config)#router bgp 64512
SanJose1(config-router)#neighbor 192.168.1.5 route-map PRIMARY_T1_MED_OUT out
SanJose1(config-router)#exit
SanJose1(config)#exit
SanJose1#clear ip bgp * soft
SanJose1#sh ip bgp
BGP table version is 8, local router ID is 172.16.64.1
 Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S
Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-
compressed, Origin codes: i - IGP, e - EGP, ? – incomplete
 RPKI validation codes: V valid, I invalid, N Not found
 Network                Next Hop            Metric     LocPrf
Weight Path * i 172.16.0.0          172.16.32.1              0
100 0 i
 *> 0.0.0.0                0
32768 i
 *> 192.168.1.0/30          192.168.1.5              0
160 0 200 i
r> 192.168.1.4/30           192.168.1.5          0
160 0 200 i
 *> 192.168.100.0           192.168.1.5          0
160 0 200 i
```

SanJose1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
172.16.0.0/16 is variably subnetted, 6 subnets, 3 masks
S 172.16.0.0/16 is directly connected, Null0
C 172.16.1.0/24 is directly connected, Serial1/1
L 172.16.1.1/32 is directly connected, Serial1/1
D 172.16.32.0/24 [90/2297856] via 172.16.1.2, 01:28:25, Serial1/1
C 172.16.64.0/24 is directly connected, Loopback0
L 172.16.64.1/32 is directly connected, Loopback0 192.168.1.0/24 is variably subnetted, 3 subnets, 2 masks
B 192.168.1.0/30 [20/0] via 192.168.1.5, 00:45:28
C 192.168.1.4/30 is directly connected, Serial1/0
L 192.168.1.6/32 is directly connected, Serial1/0
B 192.168.100.0/24 [20/0] via 192.168.1.5, 00:45:28
After issuing ip default-network
SanJose1(config)#ip default-network 192.168.100.0
SanJose1(config)#end SanJose1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 192.168.1.5 to network 192.168.100.0
S* 0.0.0.0/0 [20/0] via 192.168.1.5 172.16.0.0/16 is variably subnetted, 6 subnets, 3 masks
S 172.16.0.0/16 is directly connected, Null0
C 172.16.1.0/24 is directly connected, Serial1/1
L 172.16.1.1/32 is directly connected, Serial1/1
D 172.16.32.0/24 [90/2297856] via 172.16.1.2, 01:33:38, Serial1/1
C 172.16.64.0/24 is directly connected, Loopback0
L 172.16.64.1/32 is directly connected, Loopback0 192.168.1.0/24 is variably subnetted, 3 subnets, 2 masks
B 192.168.1.0/30 [20/0] via 192.168.1.5, 00:50:41
C 192.168.1.4/30 is directly connected, Serial1/0
L 192.168.1.6/32 is directly connected, Serial1/0
B* 192.168.100.0/24 [20/0] via 192.168.1.5, 00:50:41
SanJose1#ping 192.168.1.2

```
Type escape sequence to abort.
 Sending 5, 100-byte ICMP Echos to 192.168.1.2, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 14/15/16 ms
SanJose1#traceroute 192.168.1.2
Type escape sequence to abort.
Tracing the route to 192.168.1.2
VRF info: (vrf in name/id, vrf out name/id) 1 192.168.1.5 [AS 200] 10 msec 10 msec 10 msec 2
192.168.1.2 [AS 200] 15 msec 15 msec *
 SanJose1#ping 192.168.1.1
Type escape sequence to abort.
 Sending 5, 100-byte ICMP Echos to 192.168.1.1, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 9/9/11 ms
SanJose1#traceroute 192.168.1.1
Type escape sequence to abort.
 Tracing the route to 192.168.1.1
 VRF info: (vrf in name/id, vrf out name/id
 1 192.168.1.5 [AS 200] 10 msec 11 msec *
R3 (SanJose2)
Router>en
Router#conf t
Router(config)#hostname SanJose2
SanJose2(config)#interface Loopback0
 SanJose2(config-if)#ip address 172.16.32.1 255.255.255.0
 SanJose2(config-if)#exit
SanJose2(config)#interface Serial1/1
 SanJose2(config-if)#ip address 192.168.1.2 255.255.255.252
SanJose2(config-if)#no shutdown
 SanJose2(config-if)#exit
SanJose2(config)#interface Serial1/0
SanJose2(config-if)#ip address 172.16.1.2 255.255.255.0
SanJose2(config-if)#no shutdown
SanJose2(config-if)#exit
SanJose2(config)#router eigrp 64512
 SanJose2(config-router)#network 172.16.0.0
SanJose2(config-router)#no auto-summary
SanJose2(config-router)#exit
SanJose2(config)#router bgp 64512
 SanJose2(config-router)#neighbor 172.16.64.1 remote-as 64512
SanJose2(config-router)#neighbor 172.16.64.1 update-source loopback0
SanJose2(config-router)#exit
 SanJose2(config)#ip route 172.16.0.0 255.255.0.0 null 0
 SanJose2(config)#router bgp 64512
SanJose2(config-router)#network 172.16.0.0
 SanJose2(config-router)#neighbor 192.168.1.1 remote-as 200
SanJose2(config-router)#exit
SanJose2#sh ip bgp summary
BGP router identifier 172.16.32.1, local AS number 64512
BGP table version is 4, main routing table version 4
2 network entries using 280 bytes of memory
```

4 path entries using 320 bytes of memory 4/2 BGP path/bestpath attribute entries using 576 bytes of memory
1 BGP AS-PATH entries using 24 bytes of memory
0 BGP route-map cache entries using 0 bytes of memory 0
BGP filter-list cache entries using 0 bytes of memory
BGP using 1200 total bytes of memory
BGP activity 2/0 prefixes, 4/0 paths, scan interval 60 secs Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd 172.16.64.1 4 64512 31 32 4 0 0 00:24:41 2
192.168.1.1 4 200 8 6 4 0 0 00:01:22 1
SanJose2#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
172.16.0.0/16 is variably subnetted, 6 subnets, 3 masks
S 172.16.0.0/16 is directly connected, Null0
C 172.16.1.0/24 is directly connected, Serial1/0 L 172.16.1.2/32 is directly connected, Serial1/0
C 172.16.32.0/24 is directly connected, Loopback0
L 172.16.32.1/32 is directly connected, Loopback0
D 172.16.64.0/24 [90/2297856] via 172.16.1.1, 00:08:46, Serial1/0 192.168.1.0/24 is variably subnetted, 3 subnets, 2 mask
s C 192.168.1.0/30 is directly connected, Serial1/1
L 192.168.1.2/32 is directly connected, Serial1/1
B 192.168.1.4/30 [20/0] via 192.168.1.1, 00:02:19
B 192.168.100.0/24 [20/0] via 192.168.1.1, 00:07:40
SanJose2#sh ip bgp
BGP table version is 5, local router ID is 172.16.32.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

Network	Next Hop	Metric	LocPrf	Weight	Path
* i 172.16.0.0	172.16.64.1	0	100	0	i
*> 0.0.0.0	0			32768	i
r i 192.168.1.0/30	192.168.1.5	0	100	0	200 i
r> 192.168.1.1	0			0	200 i
* i 192.168.1.4/30	192.168.1.5	0	100	0	200 i

*> 192.168.1.1 0 0
200 i
* i 192.168.100.0 192.168.1.5 0 100
0 200 i
*> 192.168.1.1 0 0
200 i SanJose2(config)#router bgp 64512

SanJose2(config-router)#neighbor 172.16.64.1 next-hop-self
SanJose2(config-router)#exi
t SanJose2#sh ip bgp
BGP table version is 5, local router ID is 172.16.32.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found Network Next Hop Metric LocPrf Weight Path * i 172.16.0.0 172.16.64.1 0 100 0 i *> 0.0.0.0 0 32768 i r i 192.168.1.0/30 172.16.64.1 0 100 0 200 i r> 192.168.1.1 0 0 200 i * i 192.168.1.4/30 172.16.64.1 0 100 0 200 i *> 192.168.1.1 0 0 200 i * i 192.168.100.0 172.16.64.1 0 100 0 200 i *> 192.168.1.1 0 0 200 i
SanJose2(config)#route-map SECONDARY_T1_IN permit 10
SanJose2(config-route-map)#set local-preference 125
SanJose2(config-route-map)#exi
t SanJose2(config)#router bgp 64512
SanJose2(config-router)#neighbor 192.168.1.1 route-map SECONDARY_T1_IN in
SanJose2(config-router)#exit
SanJose2#clear ip bgp * soft
SanJose2#sh ip bgp
BGP table version is 8, local router ID is 172.16.32.1 Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? – incomplete

RPKI validation codes: V valid, I invalid, N Not found Network Next Hop Metric LocPrf Weight Path * i 172.16.0.0 172.16.64.1 0 100 0 i *> 0.0.0.0 0 32768 i r>i 192.168.1.0/30 172.16.64.1 0 160 0 200 i r 192.168.1.1 0 125 0 200 i *>i 192.168.1.4/30 172.16.64.1 0 160 0 200 i * 192.168.1.1 0 125 0 200 i *>i 192.168.100.0 172.16.64.1 0 160 0 200 i * 192.168.1.1 0 125 0 200 i
SanJose2(config)#route-map SECONDARY_T1_MED_OUT permit 10
SanJose2(config-route-map)#set Metric 75
SanJose2(config-route-map)#exit
SanJose2(config)#router bgp 64512
SanJose2(config-router)#$2.168.1.1 route-map SECONDARY_T1_MED_OUT out

```
SanJose2(config-router)#end
SanJose2#clear ip bgp * soft
SanJose2#sh ip bgp
BGP table version is 8, local router ID is 172.16.32.1 Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? – incomplete
RPKI validation codes: V valid, I invalid, N Not found
Network Next Hop Metric LocPrf Weight Path * i 172.16.0.0 172.16.64.1 0 100 0 i
*> 0.0.0.0 0 32768 i
r>i 192.168.1.0/30 172.16.64.1 0 160 0 200 i
r 192.168.1.1 0 125 0 200 i
*>i 192.168.1.4/30 172.16.64.1 0 160 0 200 i
* 192.168.1.1 0 125 0 200 i
*>i 192.168.100.0 172.16.64.1 0 160 0 200 i
```

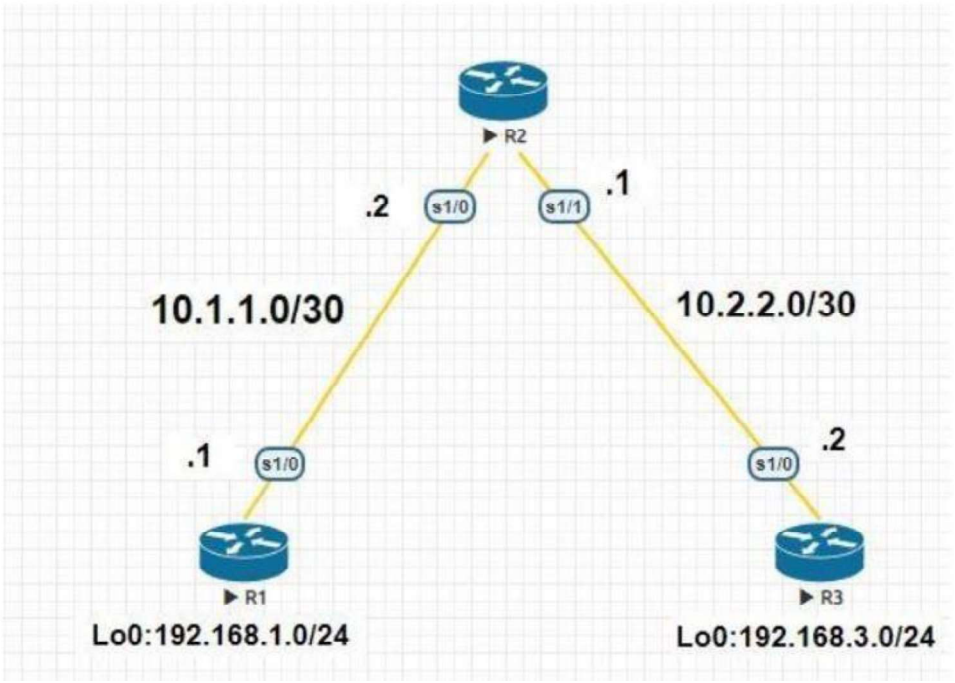
Practical 4

Aim: Secure the Management Plane.

Code:

-

NETWORK TOPOLOGY



R1 Router>en

Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname R1
R1(config)#interface Loopback 0
*Dec 19 07:53:42.473: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback0, changed state to up
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#exit
R1(config)#interface s1/0
R1(config-if)#ip address 10.1.1.1 255.255.255.252
R1(config-if)#no shutdown
*Dec 19 07:57:21.998: %LINK-3-UPDOWN: Interface Serial1/0, changed state to up
*Dec 19 07:57:22.999: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed state to up
R1(config-if)#exit

R1(config)#exit Configure static routes a.
On R1, configure a default static route to ISP.
R1(config)# ip route 0.0.0.0 0.0.0.0 10.1.1.2
R1#show ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B – BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF

NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default,U - per-user static route o - ODR, P - periodic downloaded static route,H - NHRP,l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 10.1.1.2 to network 0.0.0.0
S* 0.0.0.0/0 [1/0] via 10.1.1.2 10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C 10.1.1.0/30 is directly connected, Serial1/0
L 10.1.1.1/32 is directly connected, Serial1/0 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback0
L 192.168.1.1/32 is directly connected, Loopback0 Secure management access
R1(config)#security passwords min-length 10
R1(config)#enable secret class12345
R1(config)#line console 0
R1(config-line)#password ciscoconpass
R1(config-line)#exec-timeout 5 0
R1(config-line)#login
R1(config-line)#logging synchronous
R1(config-line)#exit
R1(config)#line vty 0 4
R1(config-line)#password ciscovtypass
R1(config-line)#exec-timeout 5 0
R1(config-line)#login
R1(config-line)#exit
R1(config)#line aux 0
R1(config-line)#no exec

R1(config-line)#end
R1(config)#service password-encryption
R1(config)#banner motd $Unauthorized access strictly prohibited!$
R1(config)#exit Configure enhanced username password security
R1(config)#username JR-ADMIN secret class12345
R1(config)#username ADMIN secret class54321
R1(config)#line console 0
R1(config-line)#login local
R1(config-line)#end
R1(config)#line vty 0 4

R1(config-line)#login local
R1(config-line)#end Enabling AAA RADIUS Authentication with Local User for Backup
R1(config)# aaa new-model
R1(config)# radius server RADIUS-1
R1(config-radius-server)# address ipv4 192.168.1.101
R1(config-radius-server)# key RADIUS-1-pa55w0rd
R1(config-radius-server)# exit
R1(config)# radius server RADIUS-2
R1(config-radius-server)# address ipv4 192.168.1.102
R1(config-radius-server)# key RADIUS-2-pa55w0rd
R1(config-radius-server)# exit

R1(config)# aaa group server radius RADIUS-GROUP
R1(config-sg-radius)# server name RADIUS-1
R1(config-sg-radius)# server name RADIUS-2
R1(config-sg-radius)# exit
R1(config)# aaa authentication login default group RADIUS-GROUP local
R1(config)# aaa authentication login TELNET-LOGIN group RADIUS-GROUP localcase
R1(config)# line vty 0 4
R1(config-line)# login authentication TELNET-LOGIN
R1(config-line)# exit
R2 Router>enable
Router#conf t Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname R2
R2(config)#interface s1/0
R2(config-if)#ip address 10.1.1.2 255.255.255.252
R2(config-if)#no shutdown
*Dec 19 08:01:10.279: %LINK-3-UPDOWN: Interface Serial1/0, changed state to up *Dec 19 08:01:11.279: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed state to up R2(config-if)#exit
R2(config)#interface s1/1
R2(config-if)#ip address 10.2.2.1 255.255.255.252
R2(config-if)#no shutdown
*Dec 19 08:02:33.002: %LINK-3-UPDOWN: Interface Serial1/1, changed state to up
*Dec 19 08:02:34.009: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/1, changed state to up
R2(config-if)#exit

R2(config)#exit Configure static routes a. On R2, configure two static routes.
R2(config)# ip route 192.168.1.0 255.255.255.0 10.1.1.1
R2(config)# ip route 192.168.3.0 255.255.255.0 10.2.2.2
R2#show ip route Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B – BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set 10.0.0.0/8 is variably subnetted, 4 subnets, 2 mask
s C 10.1.1.0/30 is directly connected, Serial1/0 L 10.1.1.2/32 is directly connected, Serial1/0
C 10.2.2.0/30 is directly connected, Serial1/1
L 10.2.2.1/32 is directly connected, Serial1/1
S 192.168.1.0/24 [1/0] via 10.1.1.1
S 192.168.3.0/24 [1/0] via 10.2.2.2 Secure management access
R2(config)#security passwords min-length 10
R2(config)#enable secret class12345
R2(config)#line console 0 R2(config-line)#password ciscoconpass
R2(config-line)#exec-timeout 5 0

R2(config-line)#login
R2(config-line)#logging synchronous
R2(config-line)#exit
R2(config)#line vty 0 4
R2(config-line)#password ciscovtypass
R2(config-line)#exec-timeout 5 0
R2(config-line)#login
R2(config-line)#exit
R2(config)#line aux 0
R2(config-line)#no exec
R2(config-line)#end
R2(config)#service password-encryption
R2(config)#banner motd $Unauthorized access strictly prohibited!$
R2(config)#exit Configure enhanced username password security
R2(config)#username JR-ADMIN secret class12345
R2(config)#username ADMIN secret class54321
R2(config)#line console 0
R2(config-line)#login local
R2(config-line)#end

R2(config)#line vty 0 4
R2(config-line)#login local
R2(config-line)#end Enabling AAA RADIUS Authentication with Local User for Backup
R2(config)# aaa new-model
R2(config)# radius server RADIUS-1
R2(config-radius-server)# address ipv4 192.168.1.101
R2(config-radius-server)# key RADIUS-1-pa55w0rd
R2(config-radius-server)# exit
R2(config)# radius server RADIUS-2
R2(config-radius-server)# address ipv4 192.168.1.102
R2(config-radius-server)# key RADIUS-2-pa55w0rd
R2(config-radius-server)# exit R2(config)# aaa group server radius RADIUS-GROUP
R2(config-sg-radius)# server name RADIUS-1
R2(config-sg-radius)# server name RADIUS-2
R2(config-sg-radius)# exit
R2(config)# aaa authentication login default group RADIUS-GROUP local
R2(config)# aaa authentication login TELNET-LOGIN group RADIUS-GROUP localcase
R2(config)# line vty 0 4
R2(config-line)# login authentication TELNET-LOGIN
R2(config-line)# exit
R3 Router>enable
Router#conf t Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname R3
R3(config)#interface loopback 0
*Dec 19 08:07:50.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback0, changed state to up
R3(config-if)#ip address 192.168.3.1 255.255.255.0

R3(config-if)#exit
R3(config)#interface s1/0
R3(config-if)#ip address 10.2.2.2 255.255.255.252
R3(config-if)#no shutdown
R3(config-if)#exit
*Dec 19 08:09:26.986: %LINK-3-UPDOWN: Interface Serial1/0, changed state to up
*Dec 19 08:09:27.996: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1/0, changed state to up

R3(config)#end Configure static routes a. On R3, configure a default static route to ISP.
R3(config)# ip route 0.0.0.0 0.0.0.0 10.2.2.1
R3#show ip route Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B – BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is 10.2.2.1 to network 0.0.0.0
S* 0.0.0.0/0 [1/0] via 10.2.2.1 10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C 10.2.2.0/30 is directly connected, Serial1/0
L 10.2.2.2/32 is directly connected, Serial1/0 192.168.3.0/24 is variably subnetted, 2 subnets, 2 masks

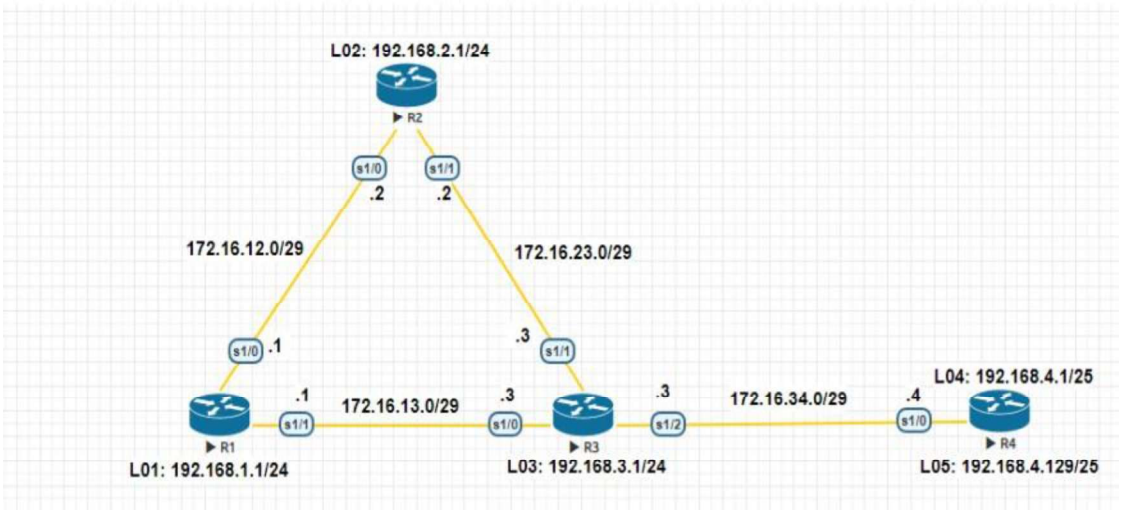
C 192.168.3.0/24 is directly connected, Loopback0
L 192.168.3.1/32 is directly connected, Loopback0 Secure management access
R3(config)#security passwords min-length 10
R3(config)#enable secret class12345
R3(config)#line console 0
R3(config-line)#password ciscoconpass
R3(config-line)#exec-timeout 5 0
R3(config-line)#login
R3(config-line)#logging synchronous
R3(config-line)#exit
R3(config)#line vty 0 4
R3(config-line)#password ciscovtypass
R3(config-line)#exec-timeout 5 0
R3(config-line)#login R3(config-line)#exit
R3(config)#line aux 0
R3(config-line)#no exec
R3(config-line)#end
R3(config)#service password-encryption
R3(config)#banner motd $Unauthorized access strictly prohibited!$ Configure enhanced username password security
R3(config)#username JR-ADMIN secret class12345
R3(config)#username ADMIN secret class54321
R3(config)#line console 0

R3(config-line)#login local
R3(config-line)#exit
R3(config)#line vty 0 4
R3(config-line)#login local
R3(config-line)#exit
Enabling AAA RADIUS Authentication with Local User for Backup
R3(config)# aaa new-model
R3(config)# radius server RADIUS-1
R3(config-radius-server)# address ipv4 192.168.1.101
R3(config-radius-server)# key RADIUS-1-pa55w0rd
R3(config-radius-server)# exit
R3(config)# radius server RADIUS-2
R3(config-radius-server)# address ipv4 192.168.1.102
R3(config-radius-server)# key RADIUS-2-pa55w0rd
R3(config-radius-server)# exit
R3(config)# aaa group server radius RADIUS-GROUP
R3(config-sg-radius)# server name RADIUS-1
R3(config-sg-radius)# server name RADIUS-2 R3(config-sg-radius)# exit
R3(config)# aaa authentication login default group RADIUS-GROUP loca
l R3(config)# aaa authentication login TELNET-LOGIN group RADIUS-GROUP localcase
R3(config)# line vty 0 4
R3(config-line)# login authentication TELNET-LOGIN
R3(config-line)# exit

Practical 5

Aim: Configure and Verify Path Control Using PBR.

NETWORK TOPOLOGY



R1 Router>enable

Router#conf t Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname R1
R1(config)#interface Lo1
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#exit
R1(config)#interface s1/0
R1(config-if)#ip address 172.16.12.1 255.255.255.248
R1(config-if)#no shutdown
R1(config-if)#exit R1(config)#interface s1/1
R1(config-if)#ip address 172.16.13.1 255.255.255.248 R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#router eigrp 100
R1(config-router)#network 192.168.1.0
R1(config-router)#network 172.16.12.0
R1(config-router)#network 172.16.13.0
R1(config-router)#no auto-summary
R1(config-router)#exit
R1#sh ip eigrp neighbors
EIGRP-IPv4 Neighbors for AS(100) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 1 172.16.13.3 Se1/1 14 00:04:43 11 100 0 10 0 172.16.12.2 Se1/0 12 00:07:05 19 114 0 8
R1#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF

NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C 172.16.12.0/29 is directly connected, Serial1/0
L 172.16.12.1/32 is directly connected, Serial1/0
C 172.16.13.0/29 is directly connected, Serial1/1
L 172.16.13.1/32 is directly connected, Serial1/1
D 172.16.23.0/29 [90/2681856] via 172.16.13.3, 00:08:31, Serial1/1 [90/2681856] via 172.16.12.2, 00:08:31, Serial1/0
D 172.16.34.0/29 [90/2681856] via 172.16.13.3, 00:08:31, Serial1/1 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback1
L 192.168.1.1/32 is directly connected, Loopback1
D 192.168.2.0/24 [90/2297856] via 172.16.12.2, 00:08:31, Serial1/0
D 192.168.3.0/24 [90/2297856] via 172.16.13.3, 00:08:31, Serial1/1 192.168.4.0/25 is subnetted, 2 subnets
D 192.168.4.0 [90/2809856] via 172.16.13.3, 00:05:15, Serial1/1
D 192.168.4.128 [90/2809856] via 172.16.13.3, 00:05:15, Serial1/1

R2 Router>enable
Router#conf t
Router(config)#hostname R2
R2(config)#interface Lo2
R2(config-if)#ip address 192.168.2.1 255.255.255.0
R2(config-if)#exit
R2(config)#interface s1/0
R2(config-if)#ip address 172.16.12.2 255.255.255.248
R2(config-if)#no shutdown
R2(config-if)#exit
R2(config)#interface s1/1
R2(config-if)#ip address 172.16.23.2 255.255.255.248
R2(config-if)#no shutdown R2(config-if)#exit
R2(config)#router eigrp 100
R2(config-router)#network 192.168.2.0
R2(config-router)#network 172.16.12.0
R2(config-router)#network 172.16.23.0
R2(config-router)#no auto-summary
R2#sh ip eigrp neighbors EIGRP-IPv4
Neighbors for AS(100) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 1 172.16.23.3 Se1/1 12 00:05:23 12 100 0 11 0 172.16.12.1 Se1/0 12 00:07:45 22 132 0 8
R3 **Router>enable**
Router#conf t
Router(config)#hostname R3
R3(config)#interface Lo3
R3(config-if)#ip address 192.168.3.1 255.255.255.0
R3(config-if)#exit
R3(config)#interface s1/0

R3(config-if)#ip address 172.16.13.3 255.255.255.248
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#interface s1/1
R3(config-if)#ip address 172.16.23.3 255.255.255.248
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#interface s1/2
R3(config-if)#ip address 172.16.34.3 255.255.255.248
R3(config-if)#no shutdown
R3(config-if)#exit
R3(config)#router eigrp 100
R3(config-router)#network 192.168.3.0
R3(config-router)#network 172.16.13.0
R3(config-router)#network 172.16.23.0
R3(config-router)#network 172.16.34.0
R3(config-router)#no auto-summary
R3#sh ip eigrp neighbors EIGRP-IPv4 Neighbors for AS(100) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 2 172.16.34.4 Se1/2 14 00:03:09 15 100 0 3 1 172.16.13.1 Se1/0 14 00:06:25 21 126 0 9 0 172.16.23.2 Se1/1 13 00:06:25 20 120 0 9
R3#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
172.16.0.0/16 is variably subnetted, 7 subnets, 2 masks
D 172.16.12.0/29 [90/2681856] via 172.16.23.2, 00:16:48, Serial1/1 [90/2681856] via 172.16.13.1, 00:16:48, Serial1/0
C 172.16.13.0/29 is directly connected, Serial1/0
L 172.16.13.3/32 is directly connected, Serial1/0
C 172.16.23.0/29 is directly connected, Serial1/1
L 172.16.23.3/32 is directly connected, Serial1/1
C 172.16.34.0/29 is directly connected, Serial1/2
L 172.16.34.3/32 is directly connected, Serial1/2
D 192.168.1.0/24 [90/2297856] via 172.16.13.1, 00:16:48, Serial1/0
D 192.168.2.0/24 [90/2297856] via 172.16.23.2, 00:16:48, Serial1/1 192.168.3.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.3.0/24 is directly connected, Loopback3
L 192.168.3.1/32 is directly connected, Loopback3 192.168.4.0/25 is subnetted, 2 subnets D 192.168.4.0 [90/2297856] via 172.16.34.4, 00:13:32, Serial1/2
D 192.168.4.128 [90/2297856] via 172.16.34.4, 00:13:32, Serial1/2
R3(config)#ip access-list standard PBR-ACL
R3(config-std-nacl)#remark ACL matches
R4 LAN B traffic
R3(config-std-nacl)#permit 192.168.4.128 0.0.0.127
R3(config-std-nacl)#exit

R3(config)#route-map R3-to-R1 permit
R3(config-route-map)#match ip address PBR-ACL
R3(config-route-map)#set ip next-hop 172.16.13.1
R3(config-route-map)#end
R3(config)#int s1/2
R3(config-if)#ip policy route-map R3-to-R1
R3(config-if)#exit
R3#sh route-map route-map R3-to-R1, permit, sequence 10 Match clauses: ip address (access-lists): PBR-ACL Set clauses: ip next-hop 172.16.13.1 Policy routing matches: 0 packets, 0 bytes
R3(config)#access-list 1 permit 192.168.4.0 0.0.0.255

R4

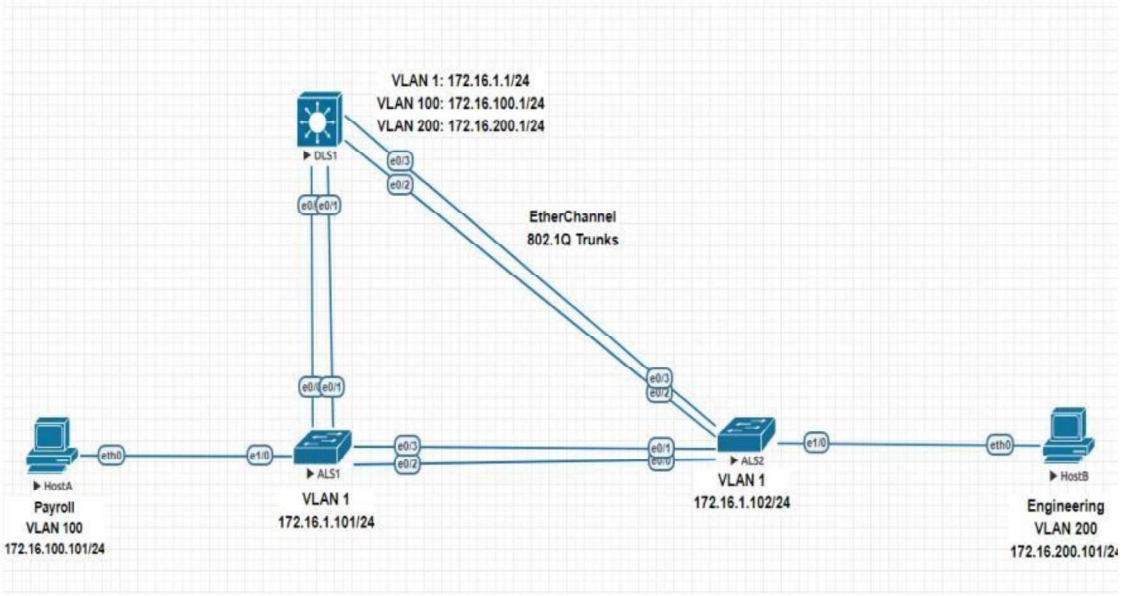
Router>enable
Router#conf t
Router(config)#hostname R4
R4(config)#interface lo4
R4(config-if)#ip address 192.168.4.1 255.255.255.128
R4(config-if)#exit
R4(config)#interface lo5
R4(config-if)#ip address 192.168.4.129 255.255.255.128
R4(config-if)#exit
R4(config)#interface s1/0
R4(config-if)#ip address 172.16.34.4 255.255.255.248
R4(config-if)#no shutdown
R4(config-if)#exit
R4(config)#router eigrp 100
R4(config-router)#network 192.168.4.0
R4(config-router)#network 172.16.34.0
R4(config-router)#no auto-summary
R4#sh ip eigrp neighbors EIGRP-IPv4 Neighbors for AS(100) H Address Interface Hold Uptime SRTT RTO Q Seq (sec) (ms) Cnt Num 0 172.16.34.3 Se1/0 14 00:04:07 25 150 0 9
Before Route Maps R4#traceroute 192.168.1.1 source 192.168.4.1
Type escape sequence to abort.
Tracing the route to 192.168.1.1
VRF info: (vrf in name/id, vrf out name/id) 1 172.16.34.3 13 msec 11 msec 10 msec 2 172.16.13.1 20 msec 17 msec *
R4#traceroute 192.168.1.1 source 192.168.4.129
Type escape sequence to abort.
Tracing the route to 192.168.1.1
VRF info: (vrf in name/id, vrf out name/id) 1 172.16.34.3 15 msec 10 msec 10 msec 2 172.16.13.1 19 msec 24 msec *
After Route Maps R4#traceroute 192.168.1.1 source 192.168.4.1
Type escape sequence to abort. Tracing the route to 192.168.1.1
VRF info: (vrf in name/id, vrf out name/id) 1 172.16.34.3 11 msec 10 msec 10 msec 2 172.16.13.1 21 msec 22 msec *
R4#traceroute 192.168.1.1 source 192.168.4.129
Type escape sequence to abort.
Tracing the route to 192.168.1.1

Practical 6

Aim: IP Service Level Agreements and Remote SPAN in a Campus Environment.

Code: -

NETWORK TOPOLOGY



DLS1 Switch>en

Switch#conf t
Switch(config)#hostname DLS1
DLS1(config)#interface vlan 1
DLS1(config-if)#ip address 172.16.1.1 255.255.255.0
DLS1(config-if)#no shutdown
DLS1(config-if)#exit Configure the trunks and EtherChannel from DLS1 to ALS1.
DLS1(config)#interface range e0/0-1
DLS1(config-if-range)#switchport trunk encapsulation dot1q
DLS1(config-if-range)#switchport mode trunk
DLS1(config-if-range)#channel-group 1 mode desirable Creating a port-channel interface Port-

channel 1
DLS1(config-if-range)#exit Configure the trunks and EtherChannel from DLS1 to ALS2.
DLS1(config)#interface range e0/2-3
DLS1(config-if-range)#switchport trunk encapsulation dot1q
DLS1(config-if-range)#switchport mode trunk
DLS1(config-if-range)#channel-group 2 mode desirable Creating a port-channel interface Port-

channel 2
DLS1(config-if-range)#exit Configure VTP on DLS1 and create VLANs 100 and 200 for the domain DLS1(config)#vtp domain SWPOD Changing VTP domain name from NULL to SWPOD DLS1(config)#vtp version 2
DLS1(config)#vlan 100
DLS1(config-vlan)#name Payroll
DLS1(config-vlan)#exit
DLS1(config)#vlan 200
DLS1(config-vlan)#name Engineering
DLS1(config-vlan)#exit On DLS1, create the SVIs for VLANs 100 and 200.
Note that the corresponding Layer 2 VLANs must be configured for the Layer 3 SVIs to activate
DLS1(config)#interface vlan 100
DLS1(config-if)#ip address 172.16.100.1 255.255.255.0
DLS1(config-if)#no shutdown
DLS1(config-if)#exit
DLS1(config)#interface vlan 200
DLS1(config-if)#ip address 172.16.200.1 255.255.255.0
DLS1(config-if)#no shutdown
DLS1(config-if)#exit The ip routing command is also needed to allow the
DLS1 switch to act as a Layer 3 device to route between these VLANs. Because the VLANs are all considered directly connected, a routing protocol is not needed at this time. The default configuration on 3560 switches is no ip routing.
DLS1(config)#ip routing
DLS1#sh ip route Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
172.16.0.0/16 is variably subnetted, 6 subnets, 2 masks
C 172.16.1.0/24 is directly connected, Vlan1
L 172.16.1.1/32 is directly connected, Vlan1
C 172.16.100.0/24 is directly connected, Vlan100
L 172.16.100.1/32 is directly connected, Vlan100
C 172.16.200.0/24 is directly connected, Vlan200
L 172.16.200.1/32 is directly connected, Vlan200
Configure the Cisco IOS IP SLA source to measure network performance
DLS1(config)#ip sla 1
DLS1(config-ip-sla)#icmp-echo 172.16.100.101
DLS1(config-ip-sla-echo)#exit
DLS1(config)#ip sla 2
DLS1(config-ip-sla)#icmp-echo 172.16.200.101
DLS1(config-ip-sla-echo)#exit
DLS1(config)#ip sla 3
DLS1(config-ip-sla)#udp-jitter 172.16.1.101 5000
DLS1(config-ip-sla-jitter)#exi
t DLS1(config)#ip sla 4
DLS1(config-ip-sla)#udp-jitter 172.16.1.102 5000

DLS1(config-ip-sla-jitter)#exit
DLS1(config)#ip sla schedule 1 life forever start-time now
DLS1(config)#ip sla schedule 2 life forever start-time now
DLS1(config)#ip sla schedule 3 life forever start-time now
DLS1(config)#ip sla schedule 4 life forever start-time now Monitor IP SLAs operations
DLS1#show ip sla configuration 1
IP SLAs Infrastructure Engine-III Entry number: 1 Owner: Tag: Operation timeout (milliseconds): 5000 Type of operation to perform: icmp-echo Target address/Source address: 172.16.100.101/0.0.0.0 Type Of Service parameter: 0x0 Request size (ARR data portion): 28 Data pattern: 0xABCDABCD Verify data: No Vrf Name: Schedule: Operation frequency (seconds): 60 (not considered if randomly scheduled)
Next Scheduled Start Time: Start Time already passed Group Scheduled : FALSE Randomly Scheduled : FALSE Life (seconds): Forever Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE Status of entry (SNMP RowStatus): Active Threshold (milliseconds): 5000

Distribution Statistics:
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1
Statistic distribution interval (milliseconds): 20 E
nhanced History: History Statistics:
Number of history Lives kept: 0
Number of history Buckets kept: 15
History Filter Type: None
DLS1#show ip sla configuration 3 IP SLAs Infrastructure Engine-III
Entry number: 3
Owner: Tag: Operation timeout (milliseconds): 5000
Type of operation to perform: udp-jitter
Target address/Source address: 172.16.1.101/0.0.0.0 Target port/Source port: 5000/0
Type Of Service parameter: 0x0
Request size (ARR data portion): 32

Packet Interval (milliseconds)/Number of packets: 20/10 Verify data:
No Vrf Name: Control Packets: enabled Schedule:
Operation frequency (seconds): 60 (not considered if randomly scheduled)
Next Scheduled Start Time: Start Time already passed Group Scheduled : FALSE Randomly Scheduled : FALSE Life (seconds): Forever Entry Ageout (seconds): never
Recurring (Starting Everyday): FALSE Status of entry (SNMP RowStatus): Active Threshold (milliseconds): 5000 Distribution Statistics:
Number of statistic hours kept: 2
Number of statistic distribution buckets kept: 1 Statistic distribution interval (milliseconds): 20 Enhanced History: Percentile:
DLS1#show ip sla application IP Service Level Agreements Version: Round Trip Time MIB 2.2.0, Infrastructure Engine-III
Supported Operation Types: icmpEcho, path-echo, path-jitter, udpEcho, tcpConnect, http dns, udpJitter, dhcp, ftp, lsp Group, lspPing, lspTrace pseudowirePing, udpApp, wspApp, mcast,

generic Supported Features: IPSLAs Event Publisher IP SLAs low memory water mark: 225778552 Estimated system max number of entries: 165365
Estimated number of configurable operations: 165241 Number of Entries configured : 4
Number of active Entries : 4
Number of pending Entries : 0
Number of inactive Entries : 0 Time of last change in whole IP SLAs: *14:08:46.139 EET Sat Apr 11 2020 DLS1#show ip sla statistics 1 IPSLAs
Latest Operation Statistics IPSLA operation id: 1 Latest RTT: 1 milliseconds Latest operation start time: 14:34:23 EET Sat Apr 11 2020
Latest operation return code: OK
Number of successes: 26
Number of failures: 1 Operation time to live: Forever
DLS1#show ip sla statistics 3 IPSLAs Latest Operation Statistics IPSLA operation id: 3 Type of operation: udp-jitter Latest RTT: 1 milliseconds Latest operation start time: 14:34:36 EET Sat Apr 11 2020 Latest operation return code: OK RTT Values: Number Of RTT: 10 RTT Min/Avg/Max: 1/1/2 milliseconds Latency one-way time
: Number of Latency one-way Samples: 6
Source to Destination Latency one way Min/Avg/Max: 0/0/1 milliseconds Destination to Source Latency one way Min/Avg/Max: 0/0/1 milliseconds Jitter Time:
Number of SD Jitter Samples: 9
Number of DS Jitter Samples: 9

Source to Destination Jitter Min/Avg/Max: 0/1/1 milliseconds
Destination to Source Jitter Min/Avg/Max: 0/1/1 milliseconds Over Threshold: Number Of RTT Over Threshold: 0 (0%) Packet Loss Values: Loss Source to Destination: 0
Source to Destination Loss Periods Number: 0 Source to Destination Loss Period Length Min/Max: 0/0 Source to Destination Inter Loss Period Length Min/Max: 0/0 Loss Destination to Source: 0 Destination to Source Loss Periods Number: 0
Destination to Source Loss Period Length Min/Max: 0/0
Destination to Source Inter Loss Period Length Min/Max: 0/0 Out Of Sequence: 0 Tail Drop: 0 Packet Late Arrival: 0 Packet Skipped: 0 Voice Score Values: Calculated Planning Impairment Factor (ICPIF): 0 Mean Opinion Score (MOS): 0 Number of successes: 27 Number of failures: 0 Operation time to live: Forever Configure Remote Span
DLS1(config)#vlan 100 DLS1(config-vlan)#remote-span
DLS1(config-vlan)#exi
t DLS1(config)#monitor session 1 source interface e0/0 both
DLS1(config)# monitor session 1 destination remote vlan 100 ALS1
Switch>en Switch#conf t
Switch(config)#hostname ALS1
ALS1(config)#interface vlan 1
ALS1(config-if)#ip address 172.16.1.101 255.255.255.0
ALS1(config-if)#no shutdown
ALS1(config-if)#exit
ALS1(config)#ip default-gateway 172.16.1.1
Configure the trunks and EtherChannel between ALS1 and DLS1

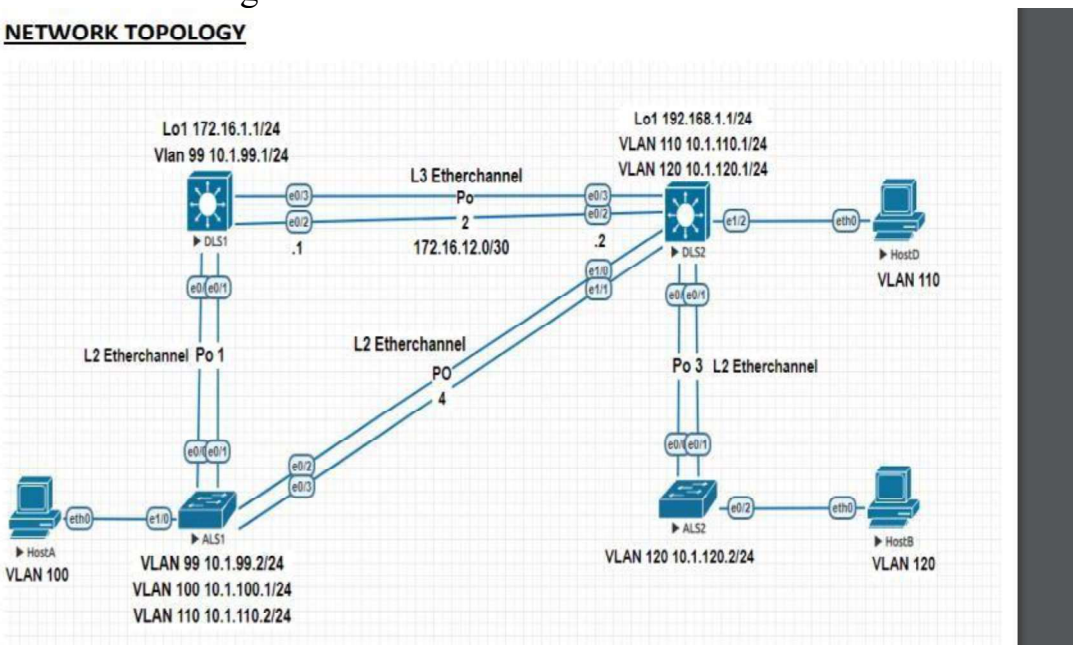
ALS1(config)#interface range e0/0-1
ALS1(config-if-range)# switchport trunk encapsulation dot1q
ALS1(config-if-range)#switchport mode trunk
ALS1(config-if-range)#channel-group 1 mode desirable Creating a port-channel interface Port-channel 1

ALS1(config-if-range)#exit
Configure the trunks and EtherChannel between ALS1 and ALS2
ALS1(config)#interface range e0/2-3
ALS1(config-if-range)#switchport trunk encapsulation dot1q
ALS1(config-if-range)#switchport mode trunk
ALS1(config-if-range)#channel-group 2 mode desirable Creating a port-channel interface Port-channel 2 Configure VTP on ALS1
ALS1(config)#vtp mode client Setting device to VTP Client mode for VLANS.
ALS1(config)#int e1/0
ALS1(config-if)#switchport mode access
ALS1(config-if)#switchport access vlan 100
ALS1(config-if)#exit Configure Cisco IOS IP SLA responders.
ALS1(config)#ip sla responder
ALS1(config)#ip sla responder udp-echo ipaddress 172.16.1.1 port 5000
ALS1#show ip sla responder General IP SLA Responder on Control port 1967
General IP SLA Responder on Control V2 port 1167 General IP SLA Responder is: Enabled
Number of control message received: 16
Number of errors: 0 Recent sources: 172.16.1.1 [14:23:36.259 EET Sat Apr 11 2020] 172.16.1.1 [14:22:36.257 EET Sat Apr 11 2020] 172.16.1.1 [14:21:36.255 EET Sat Apr 11 2020] 172.16.1.1 [14:20:36.256 EET Sat Apr 11 2020] 172.16.1.1 [14:19:36.258 EET Sat Apr 11 2020] Recent error sources:
Number of control v2 message received: 0
Number of errors: 0
Recent sources: Recent error sources:
Permanent Port IP SLA Responder Permanent Port IP SLA Responder is: Enabled udpEcho Responder: IP Address Port 172.16.1.1 5000
ALS2 Switch>en Switch#conf t Enter configuration commands, one per line. End with CNTL/Z. Switch(config)#hostname ALS2
ALS2(config)#interface vlan 1
ALS2(config-if)#ip address 172.16.1.102 255.255.255.0
ALS2(config-if)#no shutdown
ALS2(config-if)#exit
ALS2(config)#ip default-gateway 172.16.1.1 Configure the trunks and EtherChannel between ALS2 and ALS1
ALS2(config)#interface range e0/0-1
ALS2(config-if-range)#switchport trunk encapsulation dot1q
ALS2(config-if-range)#switchport mode trunk
ALS2(config-if-range)#channel-group 2 mode desirable Creating a port-channel interface Port-channel 2

ALS2(config-if-range)#exit Configure the trunks and EtherChannel between ALS2 and DLS1
ALS2(config)#interface range e0/2-3
ALS2(config-if-range)#switchport trunk encapsulation dot1q
ALS2(config-if-range)#switchport mode trunk
ALS2(config-if-range)#channel-group 1 mode desirable Creating a port-channel interface Port-channel 1
ALS2(config-if-range)#exit Configure VTP on ALS2
ALS2(config)#vtp mode
client Setting device to VTP Client mode for VLANS
ALS2(config)#int e1/0 ALS2(config-if)#switchport mode access
ALS2(config-if)#switchport access vlan 200
ALS2(config-if)#exit Configure Cisco IOS IP SLA responders.
ALS2(config)#ip sla responder
ALS2(config)#ip sla responder udp-echo ipaddress 172.16.1.1 port 5000

Practical 7

Aim: Inter-VLAN Routing.



DLS1 Switch>enable
Switch#conf t
Switch(config)#hostname DLS1
DLS1(config)#interface loopback 1
DLS1(config-if)#ip address 172.16.1.1 255.255.255.0
DLS1(config-if)#exit
DLS1(config)#interface vlan 99
DLS1(config-if)#ip address 10.1.99.1 255.255.255.0
DLS1(config-if)#no shutdown
Implement a Layer 3 EtherChannel
DLS1(config)#int range e0/2-3
DLS1(config-if-range)#no switchport
DLS1(config-if-range)#no ip address
DLS1(config-if-range)#channel-group 2 mode on Creating a port-channel interface Port-channel 2 DLS1(config-if-range)#exit
DLS1(config)#interface port-channel 2
DLS1(config-if)#ip address 172.16.12.1 255.255.255.252
DLS1(config-if)#end
DLS1(config)#int range e0/0-1
DLS1(config-if-range)#switchport trunk encapsulation dot1q
DLS1(config-if-range)#switchport mode trunk
DLS1(config-if-range)#channel-group 1 mode desirable Creating a port-channel interface Port-channel 1
DLS1(config-if-range)#end
DLS1#sh interfaces trunk Port Mode Encapsulation Status Native vlan Po1 on 802.1q trunking 1 Port Vlans allowed on trunk Po1 1-4094 Port Vlans allowed and active in management

domain Po1 1,99 Port Vlans in spanning tree forwarding state and not pruned Po1 1,99
Implement Static Routing DLS1(config)#ip routing
DLS1(config)#ip route 192.168.1.0 255.255.255.252 172.16.12.2
DLS1(config)# ip route 192.168.1.0 255.255.255.0 10.1.120.1
DLS1(config)# ip route 192.168.1.0 255.255.255.0 10.1.110.1
DLS1#sh ip route Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set 10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C 10.1.99.0/24 is directly connected, Vlan99
L 10.1.99.1/32 is directly connected, Vlan99 172.16.0.0/16 is variably subnetted, 4 subnets, 3 masks C 172.16.1.0/24 is directly connected, Loopback1
L 172.16.1.1/32 is directly connected, Loopback1
C 172.16.12.0/30 is directly connected, Port-channel2
L 172.16.12.1/32 is directly connected, Port-channel2 192.168.1.0/30 is subnetted, 1 subnets S 192.168.1.0 [1/0] via 172.16.12.2
DLS2 Switch>en Switch#conf t
Switch(config)#hostname DLS2
DLS2(config)#interface loopback 1
DLS2(config-if)#ip address 192.168.1.1 255.255.255.0
DLS2(config-if)#exit
DLS2(config)#interface vlan 110
DLS2(config-if)#ip address 10.1.110.1 255.255.255.0
DLS2(config-if)#no shutdown
DLS2(config-if)#exi
t DLS2(config)#interface vlan 120
DLS2(config-if)#ip address 10.1.120.1 255.255.255.0
DLS2(config-if)#no shutdown
DLS2(config-if)#exit Implement a Layer 3 EtherChannel
DLS2(config)#interface range e0/2-3
DLS2(config-if-range)#no switchport
DLS2(config-if-range)#no ip
DLS2(config-if-range)#no ip address
DLS2(config-if-range)#channel-group 2 mode on Creating a port-channel interface Port-channel 2 DLS2(config-if-range)#exit
DLS2(config)#interface port-channel 2
DLS2(config-if)#ip address 172.16.12.2 255.255.255.252
DLS2(config-if)#end DLS2(config)#interface range e0/0-1
DLS2(config-if-range)#switchport trunk encapsulation dot1q
DLS2(config-if-range)#switchport mode trunk
DLS2(config-if-range)#channel-group 3 mode desirable Creating a port-channel interface Port-channel 3
DLS2(config-if-range)#exit
DLS2(config)#interface range e1/0-1
DLS2(config-if-range)#switchport trunk encapsulation dot1q
DLS2(config-if-range)#switchport mode trunk

DLS2(config-if-range)#channel-group 4 mode desirable Creating a port-channel interface Port-channel 4
DLS2(config-if-range)#end
DLS2#sh interfaces trunk Port Mode Encapsulation Status Native vlan Po3 on 802.1q trunking 1 Po4 on 802.1q trunking 1 Port Vlans allowed on trunk Po3 1-4094 Po4 1-4094 Port Vlans allowed and active in management domain Po3 1,110,120 Po4 1,110,120 Port Vlans in spanning tree forwarding state and not pruned Po3 1,110,120 Po4 1,110,120 Implement Static Routing
DLS2(config)#ip routing DLS2(config)#ip route 172.16.1.0 255.255.255.252 172.16.12.1
DLS2(config)# ip route 172.16.1.0 255.255.255.0 10.1.99.1 Configure the host ports for the appropriate VLANs according to the diagram
DLS2(config)#interface e1/2
DLS2(config-if)#switchport mode access
DLS2(config-if)#switchport access vlan 110
DLS2#sh ip route
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set 10.0.0.0/8 is variably subnetted, 4 subnets, 2 masks
C 10.1.110.0/24 is directly connected, Vlan110
L 10.1.110.1/32 is directly connected, Vlan110
C 10.1.120.0/24 is directly connected, Vlan120
L 10.1.120.1/32 is directly connected, Vlan120 172.16.0.0/16 is variably subnetted, 3 subnets, 2 masks S 172.16.1.0/30 [1/0] via 172.16.12.1
C 172.16.12.0/30 is directly connected, Port-channel2
L 172.16.12.2/32 is directly connected, Port-channel2 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Loopback1
L 192.168.1.1/32 is directly connected, Loopback1 ALS1
Switch>en Switch#conf t S
witch(config)#hostname ALS1
ALS1(config)#ip default-gateway 10.1.99.1
ALS1(config)#ip default-gateway 10.1.110.1
ALS1(config)#ip default-gateway 10.1.100.2 Implement a Layer 3 EtherChannel
ALS1(config)#int range e0/0-1
ALS1(config-if-range)#switchport trunk encapsulation dot1q
ALS1(config-if-range)#switchport mode trunk
ALS1(config-if-range)#channel-group 1 mode desirable Creating a port-channel interface Port-channel 1
ALS1(config-if-range)#exit
ALS1(config)#int range e0/2-3
ALS1(config-if-range)#switchport trunk encapsulation dot1q
ALS1(config-if-range)#switchport mode trunk
ALS1(config-if-range)#channel-group 4 mode desirable Creating a port-channel interface Port-channel 4
ALS1(config-if-range)#end
ALS1#sh etherchannel summary

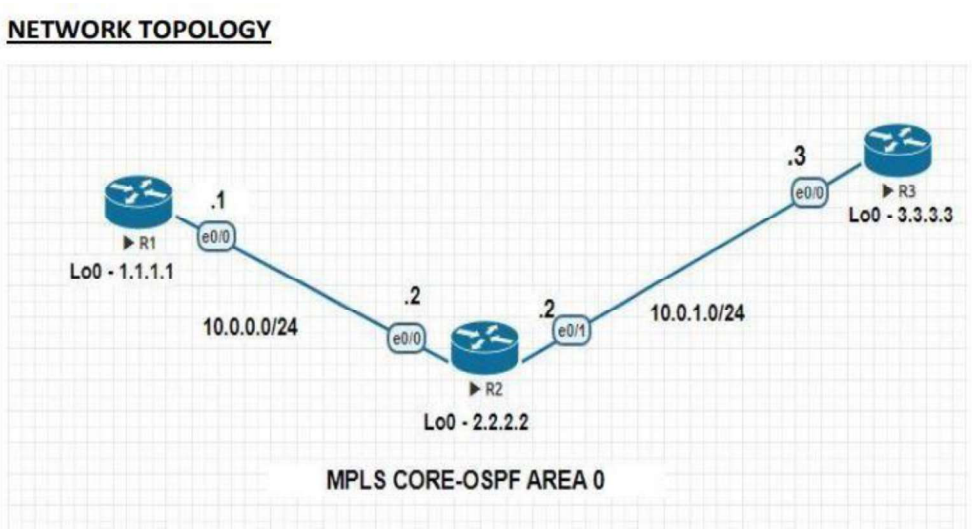
Flags: D - down P - bundled in port-channel I - stand-alone s - suspended H - Hot-standby (LACP only) R - Layer3 S - Layer2 U - in use N - not in use, no aggregation f - failed to allocate aggregator M - not in use, minimum links not met m - not in use, port not aggregated due to minimum links not met u - unsuitable for bundling w - waiting to be aggregated d - default port A - formed by Auto LAG Number of channel-groups in use: 2
Number of aggregators: 2
Group Port-channel Protocol Ports - - - - +- - - - - - - - - +- - - - - - - +- 1 Po1(SU) PAgP Et0/0(P) Et0/1(P) 4 Po4(SU) PAgP Et0/2(P) Et0/3(P)
Configure the host ports for the appropriate VLANs according to the diagram
ALS1(config)#interface e1/0
ALS1(config-if)#switchport mode access
ALS1(config-if)#switchport access vlan 100 ALS2
Switch>en Switch#conf t
Switch(config)#hostname ALS2
ALS2(config)#ip default-gateway 10.1.120.1 Implement a Layer 3 EtherChannel
ALS2(config)#int range e0/0-1
ALS2(config-if-range)#switchport trunk encapsulation dot1q
ALS2(config-if-range)#switchport mode trunk
ALS2(config-if-range)#channel-group 3 mode desirable Creating a port-channel interface Port-channel 3
ALS2(config-if-range)#end
ALS2#sh etherchannel summary
Flags: D - down P - bundled in port-channel I - stand-alone s - suspended H - Hot-standby (LACP only) R - Layer3 S - Layer2 U - in use N - not in use, no aggregation f - failed to allocate aggregator M - not in use, minimum links not met m - not in use, port not aggregated due to minimum links not met u - unsuitable for bundling w - waiting to be aggregated d - default port A - formed by Auto LAG Number of channel-groups in use: 1
Number of aggregators: 1
Group Port-channel Protocol Ports - - - - +- - - - - - - - - +- - - - - - - +- 3 Po3(SU) PAgP Et0/0(P) Et0/1(P) Configure the host ports for the appropriate VLANs according to the diagram
ALS2(config)#interface e0/2
ALS2(config-if)#switchport mode access
ALS2(config-if)#switchport access vlan 120 HOST A VPCS> ip 10.1.100.1 255.255.255.0 10.1.100.2 HOST B
VPCS> ip 10.1.120.2 255.255.255.0 10.1.120.1 HOST D
VPCS> ip 10.1.110.2 255.255.255.0 10.1.110.1

Practical 8

Aim: Simulating an MPLS environment and Simulating VRF.

Code:

-



R1 Router>enable

Router#conf t

Router(config)#hostname R1

R1(config)# interface loopback 0

R1(config-if)#ip address 1.1.1.1 255.255.255.255

R1(config-if)#exit R1(config)#int e0/0

R1(config-if)#ip address 10.0.0.1 255.255.255.0

R1(config-if)#no shut

R1(config)#router ospf 1

R1(config-router)#network 1.1.1.0 0.0.0.255 area 0

R1(config-router)#network 10.0.0.0 0.0.0.255 area 0

R1(config-router)#exit R

1#show ip route ospf

Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override

Gateway of last resort is not set

2.0.0.0/32 is subnetted, 1 subnets O 2.2.2.2 [110/11] via 10.0.0.2, 00:15:40, Ethernet0/0

3.0.0.0/32 is subnetted, 1 subnets

O 3.3.3.3 [110/21] via 10.0.0.2, 00:04:01, Ethernet0/0 10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks

O 10.0.1.0/24 [110/20] via 10.0.0.2, 00:09:25, Ethernet0/0

R1#sh ip cef Prefix Next Hop Interface 0.0.0.0/0 no route 0.0.0.0/8 drop 0.0.0.0/32 receive 1.1.1.1/32 receive Loopback0 2.2.2.2/32 10.0.0.2

Ethernet0/0 3.3.3.3/32 10.0.0.2 Ethernet0/0 10.0.0.0/24 attached Ethernet0/0 10.0.0.0/32 receive Ethernet0/0 10.0.0.1/32 receive Ethernet0/0 10.0.0.2/32 attached Ethernet0/0 10.0.0.255/32 receive Ethernet0/0 10.0.1.0/24 10.0.0.2
Ethernet0/0 127.0.0.0/8 drop 224.0.0.0/4 drop 224.0.0.0/24 receive 240.0.0.0/4 drop 255.255.255.255/32 receive
R1#sh ip route 2.2.2.2
Routing entry for 2.2.2.2/32 Known via "ospf 1", distance 110, metric 11, type intra area Last update from 10.0.0.2 on Ethernet0/0, 00:30:34 ago Routing Descriptor Blocks: * 10.0.0.2, from 2.2.2.2, 00:30:34 ago, via Ethernet0/0 Route metric is 11, traffic share count is 1 R1#sh ip route 3.3.3.3 Routing entry for 3.3.3.3/32 Known via "ospf 1", distance 110, metric 21, type intra area Last update from 10.0.0.2 on Ethernet0/0, 00:11:43 ago Routing Descriptor Blocks: * 10.0.0.2, from 3.3.3.3, 00:11:43 ago, via Ethernet0/0 Route metric is 21, traffic share count is 1 R1#sh ip cef 2.2.2.2 2.2.2.2/32 nexthop 10.0.0.2 Ethernet0/0
R1#sh ip cef 3.3.3.3 3.3.3.3/32 nexthop 10.0.0.2 Ethernet0/0
R1(config)#mpls label range 100 199
R1(config)#mpls label protocol ldp
R1(config)#mpls ldp router-id loopback 0
R1(config)#int e0/0
R1(config-if)#mpls ip
R1#sh mpls interfaces Interface IP Tunnel BGP Static Operational Ethernet0/0 Yes (ldp) No No No Yes
R1#sh mpls ldp neighbor Peer LDP Ident: 2.2.2.2:0; Local LDP Ident 1.1.1.1:0 TCP connection: 2.2.2.2.27963 - 1.1.1.1.646 State: Oper; Msgs sent/rcvd: 13/14; Downstream Up time: 00:05:21 LDP discovery sources: Ethernet0/0, Src IP addr: 10.0.0.2 Addresses bound to peer LDP Ident: 10.0.0.2 10.0.1.2 2.2.2.2
R1#sh ip cef 3.3.3.3 3.3.3.3/32 nexthop 10.0.0.2 Ethernet0/0 label 201
R1#sh ip cef 2.2.2.2 2.2.2.2/32 nexthop 10.0.0.2 Ethernet0/0
R1#sh mpls forwarding-table
Local Outgoing Prefix Bytes Label Outgoing Next Hop Label Label or Tunnel Id Switched interface 100 Pop Label 2.2.2.2/32 0 Et0/0 10.0.0.2 101 201 3.3.3.3/32 0 Et0/0 10.0.0.2 102 Pop Label 10.0.1.0/24 0 Et0/0 10.0.0.2
R1#sh mpls ldp bindings lib entry: 1.1.1.1/32, rev 2
local binding: label: imp-null
remote binding: lsr: 2.2.2.2:0, label: 200
lib entry: 2.2.2.2/32, rev 4 local binding: label: 100 remote binding: lsr: 2.2.2.2:0, label: imp-null
lib entry: 3.3.3.3/32, rev 6 local binding: label: 101 remote binding: lsr: 2.2.2.2:0, label: 201 lib entry: 10.0.0.0/24, rev 8 local binding: label: imp-null remote binding: lsr: 2.2.2.2:0, label: imp-null lib entry: 10.0.1.0/24, rev 10 local binding: label: 102 remote binding: lsr: 2.2.2.2:0, label: imp-null
R1#ping 3.3.3.3 source 10.0.0.1
Type escape sequence to abort. Sending 5, 100-byte
ICMP Echos to 3.3.3.3, timeout is 2 seconds:
Packet sent with a source address of 10.0.0.1 !!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms
R1#traceroute 3.3.3.3 source 10.0.0.1 Type escape sequence to abort.
Tracing the route to 3.3.3.3 VRF info: (vrf in name/id, vrf out name/id) 1 10.0.0.2 [MPLS: Label 201 Exp 0] 1 msec 1 msec 0 msec 2 10.0.1.3 1 msec 2 msec

* R1#ping 2.2.2.2 source 10.0.0.1 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 2.2.2.2, timeout is 2 seconds:
Packet sent with a source address of 10.0.0.1 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 5/5/6 ms R1#traceroute 2.2.2.2 source 10.0.0.1 Type escape sequence to abort.
Tracing the route to 2.2.2.2 VRF info: (vrf in name/id, vrf out name/id) 1 10.0.0.2 2 msec 1 msec

*** R2**

Router>enable
Router#conf t
Router(config)#hostname R2
R2(config)# interface loopback 0
R2(config-if)#ip address 2.2.2.2 255.255.255.255
R2(config-if)# exit
R2(config)#int e0/0
R2(config-if)#ip address 10.0.0.2 255.255.255.0 R2(config-if)#no shut
R2(config)#int e0/1
R2(config-if)#ip address 10.0.1.2 255.255.255.0
R2(config-if)#no shut
R2(config)#router ospf 1
R2(config-router)#network 2.2.2.0 0.0.0.255 area 0
R2(config-router)#network 10.0.0.0 0.0.0.255 area 0 R2(config-router)#network 10.0.1.0 0.0.0.255 area 0
R2(config-router)#exit
R2#show ip route
ospf Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
1.0.0.0/32 is subnetted, 1 subnets O 1.1.1.1 [110/11] via 10.0.0.1, 00:15:32, Ethernet0/0
3.0.0.0/32 is subnetted, 1 subnets O 3.3.3.3 [110/11] via 10.0.1.3, 00:03:58, Ethernet0/1
R2#sh ip cef Prefix Next Hop Interface 0.0.0.0/0 no route 0.0.0.0/8 drop 0.0.0.0/32
receive 1.1.1.1/32 10.0.0.1 Ethernet0/0 2.2.2.2/32
receive Loopback0 3.3.3.3/32 10.0.1.3 Ethernet0/1 10.0.0.0/24 attached Ethernet0/0 10.0.0.0/32
receive Ethernet0/0 10.0.0.1/32 attached Ethernet0/0 10.0.0.2/32
receive Ethernet0/0 10.0.0.255/32
receive Ethernet0/0 10.0.1.0/24 attached Ethernet0/1 10.0.1.0/32
receive Ethernet0/1 10.0.1.2/32
receive Ethernet0/1 10.0.1.3/32 attached Ethernet0/1 10.0.1.255/32 r
eceive Ethernet0/1 127.0.0.0/8 drop 224.0.0.0/4 drop 224.0.0.0/24
receive 240.0.0.0/4 drop 255.255.255.255/32 receive
R2#sh ip route 1.1.1.1
Routing entry for 1.1.1.1/32 Known via "ospf 1", distance 110, metric 11, type intra area Last update from 10.0.0.1 on Ethernet0/0, 00:33:11 ago
Routing Descriptor Blocks: * 10.0.0.1, from 1.1.1.1, 00:33:11 ago, via Ethernet0/0 Route metric is 11, traffic share count is 1

R2#sh ip route 3.3.3.3
Routing entry for 3.3.3.3/32 Known via "ospf 1", distance 110, metric 11, type intra area Last update from 10.0.1.3 on Ethernet0/1, 00:21:49 ago R
outing Descriptor Blocks: * 10.0.1.3, from 3.3.3.3, 00:21:49 ago, via Ethernet0/1 Route metric is 11, traffic share count is 1
R2#sh ip cef 1.1.1.1 1.1.1.1/32 nexthop 10.0.0.1 Ethernet0/0 R2#sh ip cef 3.3.3.3 3.3.3.3/32 nexthop 10.0.1.3 Ethernet0/1
R2(config)#mpls label range 200 299
R2(config)#mpls label protocol ldp
R2(config)#mpls ldp router-id loopback 0
R2(config)#int e0/0
R2(config-if)#mpls ip
R2(config-if)#int e0/1
R2(config-if)#mpls ip
R2#sh mpls interfaces
Interface IP Tunnel BGP Static Operational Ethernet0/0 Yes (ldp) No No No Yes Ethernet0/1 Yes (ldp) No No No Yes
R2#sh mpls forwarding-table
Local Outgoing Prefix Bytes Label Outgoing Next Hop Label Label or Tunnel Id Switched interface 200 Pop Label 1.1.1.1/32 0 Et0/0 10.0.0.1 201 Pop Label 3.3.3.3/32 1266 Et0/1 10.0.1.3
R2#sh mpls ldp neighbor
Peer LDP Ident: 1.1.1.1:0; Local LDP Ident 2.2.2.2:0 TCP connection: 1.1.1.1.646 - 2.2.2.2.27963 State: Oper; Msgs sent/rcvd: 41/42; Downstream Up time: 00:29:24 LDP discovery sources: Ethernet0/0, Src IP addr: 10.0.0.1 Addresses bound to peer LDP Ident: 10.0.0.1 1.1.1.1 Peer LDP Ident: 3.3.3.3:0; Local LDP Ident 2.2.2.2:0 TCP connection: 3.3.3.3.44196 - 2.2.2.2.646 State: Oper; Msgs sent/rcvd: 38/38; Downstream Up time: 00:27:24 LDP discovery sources: Ethernet0/1, Src IP addr: 10.0.1.3 Addresses bound to peer LDP Ident: 10.0.1.3 3.3.3.3
R2#sh mpls ldp bindings
lib entry: 1.1.1.1/32, rev 2 local binding: label: 200 remote binding: lsr: 1.1.1.1:0, label: imp-null remote binding: lsr: 3.3.3.3:0, label: 300
lib entry: 2.2.2.2/32, rev 4 local binding: label: imp-null remote binding: lsr: 1.1.1.1:0, label: 100 remote binding: lsr: 3.3.3.3:0, label: 301
lib entry: 3.3.3.3/32, rev 6 local binding: label: 201 remote binding: lsr: 1.1.1.1:0, label: 101 remote binding: lsr: 3.3.3.3:0, label: imp-null
lib entry: 10.0.0.0/24, rev 8 local binding: label: imp-null remote binding: lsr: 1.1.1.1:0, label: imp-null remote binding: lsr: 3.3.3.3:0, label: 302
lib entry: 10.0.1.0/24, rev 10 local binding: label: imp-null remote binding: lsr: 1.1.1.1:0, label: 102 remote binding: lsr: 3.3.3.3:0, label: imp-null
R2#ping 1.1.1.1 source 10.0.0.2
Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds:
Packet sent with a source address of 10.0.0.2 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R2#traceroute 1.1.1.1 source 10.0.0.2 Type escape sequence to abort. Tracing the route to 1.1.1.1
VRF info: (vrf in name/id, vrf out name/id) 1 10.0.0.1 2 msec 1 msec *

R2#ping 3.3.3.3 source 10.0.1.2 Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 3.3.3.3, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.2 !!!!! Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R2#traceroute 3.3.3.3 source 10.0.1.2 Type escape sequence to abort. Tracing the route to 3.3.3.3 VRF info: (vrf in name/id, vrf out name/id) 1 10.0.1.3 0 msec 1 msec *
R3
Router>enable Router#conf t
Router(config)#hostname R
3 R3(config)#interface loopback 0
R3(config-if)#ip address 3.3.3.3 255.255.255.255
R3(config-if)#exit
R3(config)#int e0/0
R3(config-if)#ip address 10.0.1.3 255.255.255.0
R3(config-if)#no shu
t R3(config-if)#exit
R3(config)#router ospf 1
R3(config-router)#network 3.3.3.0 0.0.0.255 area 0
R3(config-router)#network 10.0.1.0 0.0.0.255 area 0
R3(config-router)#exit
R3#sh ip route osp
f Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary,
L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
1.0.0.0/32 is subnetted, 1 subnets O 1.1.1.1 [110/21] via 10.0.1.2, 00:03:45,
Ethernet0/0 2.0.0.0/32 is subnetted, 1 subnets O 2.2.2.2 [110/11] via 10.0.1.2, 00:03:45,
Ethernet0/0 10.0.0.0/8 is variably subnetted, 3 subnets, 2 masks O 10.0.0.0/24 [110/20] via 10.0.1.2, 00:03:45,
Ethernet0/0
R3#sh ip cef Prefix
Next Hop Interface 0.0.0.0/0 no route 0.0.0.0/8 drop 0.0.0.0/32 receive 1.1.1.1/32 10.0.1.2 Ethernet0/0 2.2.2.2/32 10.0.1.2 Ethernet0/0 3.3.3.3/32 receive Loopback0 10.0.0.0/24 10.0.1.2 Ethernet0/0 10.0.1.0/24 attached
Ethernet0/0 10.0.1.0/32 receive Ethernet0/0 10.0.1.2/32 attached Ethernet0/0 10.0.1.3/32 receive Ethernet0/0 10.0.1.255/32 receive
Ethernet0/0 127.0.0.0/8 drop 224.0.0.0/4 drop 224.0.0.0/24 receive 240.0.0.0/4 drop 255.255.255.255/32 receive
R3#sh ip route 1.1.1.1
Routing entry for 1.1.1.1/32 Known via "ospf 1", distance 110, metric 21, type intra area Last update from 10.0.1.2 on Ethernet0/0, 00:23:51 ago Routing Descriptor Blocks: * 10.0.1.2, from 1.1.1.1, 00:23:51 ago, via Ethernet0/0 Route metric is 21, traffic share count is 1
R3#sh ip route 2.2.2.2 Routing entry for 2.2.2.2/32 Known via "ospf 1", distance 110, metric 11, type intra area Last update from 10.0.1.2 on Ethernet0/0, 00:23:58 ago Routing Descriptor

Blocks: * 10.0.1.2, from 2.2.2.2, 00:23:58 ago, via Ethernet0/0 Route metric is 11, traffic share count is 1
R3#sh ip cef 1.1.1.1 1.1.1.1/32 nexthop 10.0.1.2 Ethernet0/0
R3#sh ip cef 2.2.2.2 2.2.2.2/32 nexthop 10.0.1.2 Ethernet0/0
R3(config)#mpls label range 300 399
R3(config)#mpls lab
el protocol ldp
R3(config)#mpls ldp router-id loopback 0
R3(config)#int e0/0
R3(config-if)#mpls ip R3#sh mpls interfaces Interface IP Tunnel BGP Static Operational
Ethernet0/0 Yes (ldp) No No No Yes R3#sh mpls ldp binding
lib entry: 1.1.1.1/32, rev 2 local binding: label: 300 remote
binding: lsr: 2.2.2.2:0, label: 200 lib entry: 2.2.2.2/32, rev 4 local
binding: label: 301 remote binding: lsr: 2.2.2.2:0, label: imp-null lib entry: 3.3.3.3/32,
rev 6 local binding: label: imp-null remote binding: lsr: 2.2.2.2:0, label: 201 lib entry:
10.0.0.0/24,
rev 8 local binding: label: 302 remote binding: lsr: 2.2.2.2:0, label: imp-null lib entry:
10.0.1.0/24,
rev 10 local binding: label: imp-null remote binding: lsr: 2.2.2.2:0, label: imp-null
R3#sh mpls ldp neighbor Peer LDP Ident: 2.2.2.2:0; Local LDP Ident 3.3.3.3:0
TCP connection: 2.2.2.2.646 - 3.3.3.3.44196 State: Oper; Msgs sent/rcvd: 51/51;
Downstream Up time: 00:38:15
LDP discovery sources:
Ethernet0/0, Src IP addr: 10.0.1.2 Addresses bound to peer LDP Ident: 10.0.0.2 10.0.1.2 2.2.2.2
R3#sh mpls forwarding-table
Local Outgoing Prefix Bytes
Label Outgoing Next Hop Label Label or Tunnel Id Switched interface 300 200 1.1.1.1/32 0
Et0/0 10.0.1.2 301 Pop Label 2.2.2.2/32 0 Et0/0 10.0.1.2 302 Pop Label 10.0.0.0/24 0 Et0/0
10.0.1.2
R3#sh ip cef 1.1.1.1 1.1.1.1/32 nexthop 10.0.1.2 Ethernet0/0 label 200
R3#sh ip cef 2.2.2.2 2.2.2.2/32 nexthop 10.0.1.2 Ethernet0/0
R3#ping 1.1.1.1 source 10.0.1.3 Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 1.1.1.1, timeout is 2 seconds: Packet sent with a source address of 10.0.1.3 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/3 ms R3#traceroute 1.1.1.1 source 10.0.1.3
Type escape sequence to abort.
Tracing the route to 1.1.1.1
VRF info:

(vrf in name/id, vrf out name/id) 1 10.0.1.2 [MPLS: Label 200 Exp 0] 1 msec 2 msec 1 msec 2 10.0.0.1 2 msec 2 msec *
R3#ping 2.2.2.2 source 10.0.1.3 Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2.2.2.2, timeout is 2 seconds:
Packet sent with a source address of 10.0.1.3 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R3#traceroute 2.2.2.2
source 10.0.1.3
Type escape sequence to abort. Tracing the route to 2.2.2.2

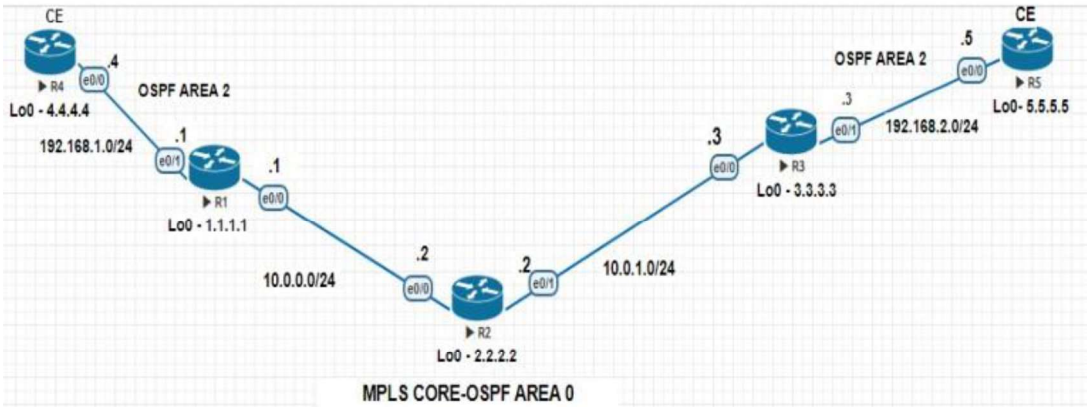
Practical 9

Aim: - Simulating SDN with
• Open Daylight SDN Controller with the Mininet Network Emulator
• OF Net SDN network emulator

Code:

-

NETWORK TOPOLOGY



R1
Router>enable
Router#conf t
Router(config)#hostname R1
R1(config)# interface loopback 0
R1(config-if)#ip address 1.1.1.1 255.255.255.255
R1(config-if)#exit
R1(config)#int e0/0
R1(config-if)#ip address 10.0.0.1 255.255.255.0
R1(config-if)#no shut
R1(config)#int e0/1
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shut
R1(config)#router ospf 1
R1(config-router)#network 1.1.1.0 0.0.0.255 area 0
R1(config-router)#network 10.0.0.0 0.0.0.255 area 0 R1(config-router)#exit
R1(config)#mpls label range 100 199
R1(config)#mpls label protocol ldp
R1(config)#mpls ldp router-id loopback 0
R1(config)#int e0/0
R1(config-if)#mpls ip
R1(config)#ip vrf A-1
R1(config-vrf)#rd 500:1
R1(config-vrf)#route-target import 500:1
R1(config-vrf)#route-target export 500:1
R1(config-vrf)#exit
R1(config)#exit
R1#sh ip vrf Name Default RD Interfaces A-1 500:1 R1#sh ip vrf detail VRF A-1 (VRF Id = 1);

default RD 500:1;
default VPNID Old CLI format, supports IPv4 only Flags: 0xC No interfaces Address family ipv4 unicast (Table ID = 0x1): Flags: 0x0 Export VPN route-target communities RT:500:1 Import VPN route-target communities RT:500:1 No import route-map No global export route-map No export route-map VRF label distribution protocol: not configured VRF label allocation mode: per-prefix
R1(config)#int e0/1
R1(config-if)#ip vrf forwarding A-1 % Interface Ethernet0/1 IPv4 disabled and address(es) removed due to enabling VRF A-1 R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#end
R1#sh ip route vrf A-1
Routing Table: A-1 Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS
summary,
L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Ethernet0/1
L 192.168.1.1/32 is directly connected, Ethernet0/1
R1#sh ip vrf Name Default RD Interfaces A-1 500:1 Et0/1
R1(config)#router ospf 10 vrf A-1
R1(config-router)#network 192.168.1.0 0.0.0.255 area 10
R1(config-router)#end
R1#sh ip ospf neighbor Neighbor ID Pri State Dead Time Address In
terface 2.2.2.2 1 FULL/DR 00:00:39 10.0.0.2 Ethernet0/0 4.4.4.4 1 FULL/DR 00:00:38 192.168.1.4 Ethernet0/1 R1#sh ip ospf 10 neighbor Neighbor ID Pri State Dead Time Address Interface 4.4.4.4 1 FULL/DR 00:00:38 192.168.1.4 Ethernet0/1
R1#sh ip route vrf A-1 ospf
Routing Table: A-1 Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS
summary,
L1 - IS-IS level-1,
L2 - IS-IS level-2 ia - IS-IS inter area,
* - candidate default,
U - per-user static route o – ODR
, P - periodic downloaded static route, H - NHRP,
l - LISP a - application route + - replicated route,
% - next hop override Gateway of last resort is not set 4.0.0.0/32 is subnetted, 1 subnets O 4.4.4.4 [110/11] via 192.168.1.4, 00:03:58, Ethernet0/1 R1(config)#router bgp 500
R1(config-router)#no bgp default ipv4-unicast
R1(config-router)#neighbor 3.3.3.3 remote-as 500
R1(config-router)#neighbor 3.3.3.3 update-source loopback 0
R1(config-router)#address-family vpnv4 unicast

R1(config-router-af)#neighbor 3.3.3.3 activate R1(config-router-af)#neighbor 3.3.3.3 send-community extended R1(config-router-af)#neighbor 3.3.3.3 next-hop-self R1(config-router-af)#end
R1#sh ip bgp vpnv4 all summary
BGP router identifier 1.1.1.1, local AS number 500 BGP table version is 1, main routing table version 1 Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd 3.3.3.3 4 500 6 7 1 0 0 00:03:19 0
R1(config)#router bgp 500
R1(config-router)#address-family ipv4 vrf A-1
R1(config-router-af)#redistribute ospf 10 vrf A-1 match internal external 1 external 2
R1(config-router-af)#exit R1(config-router)#exit
R1(config)#router ospf 10 vrf A-1
R1(config-router)#redistribute bgp 500 subnets
R1(config-router)#end R1#sh ip bgp vpnv4 all BGP table version is 7, local router ID is 1.1.1.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete RPKI validation codes: V valid, I invalid, N Not found
Network Next Hop Metric LocPrf Weight Path Route Distinguisher: 500:1 (default for vrf A-1)
*> 4.4.4.4/32 192.168.1.4 11 32768 ?
*>i 5.5.5.5/32 3.3.3.3 11 100 0 ?
*> 192.168.1. 0 0.0.0.0 0 32768 ?
*>i 192.168.2.0 3.3.3.3 0 100 0 ?
R1#sh ip route vrf A-1
Routing Table: A-1 Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS
summary,
L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
4.0.0.0/32 is subnetted, 1 subnets O 4.4.4.4 [110/11] via 192.168.1.4, 07:36:09, Ethernet0/1 5.0.0.0/32 is subnetted, 1 subnets B 5.5.5.5 [200/11] via 3.3.3.3, 00:06:15 192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C 192.168.1.0/24 is directly connected, Ethernet0/1 L 192.168.1.1/32 is directly connected, Ethernet0/1 B 192.168.2.0/24 [200/0] via 3.3.3.3, 00:06:15
R1#sh ip route vrf A-1 bgp
Routing Table: A-1 Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set 5.0.0.0/32 is subnetted, 1 subnets B 5.5.5.5 [200/11] via 3.3.3.3, 00:07:31 B 192.168.2.0/24 [200/0] via 3.3.3.3, 00:07:31
R1#ping vrf A-1 4.4.4.4

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 4.4.4.4, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/6 ms
R2
Router>enable
Router#conf t
Router(config)#hostname R2
R2(config)# interface loopback 0
R2(config-if)#ip address 2.2.2.2 255.255.255.255
R2(config-if)# exit
R2(config)#int e0/0
R2(config-if)#ip address 10.0.0.2 255.255.255.0
R2(config-if)#no shut
R2(config)#int e0/1
R2(config-if)#ip address 10.0.1.2 255.255.255.0
R2(config-if)#no shut
R2(config)#router ospf 1
R2(config-router)#network 2.2.2.0 0.0.0.255 area 0
R2(config-router)#network 10.0.0.0 0.0.0.255 area 0
R2(config-router)#network 10.0.1.0 0.0.0.255 area 0
R2(config-router)#exit
R2(config)#mpls label range 200 299 R2(config)#mpls label protocol ldp
R2(config)#mpls ldp router-id loopback 0
R2(config)#int e0/0 R2(config-if)#mpls ip
R2(config-if)#int e0/1
R2(config-if)#mpls ip
R3
Router>enable
Router#conf t
Router(config)#hostname

R3
R3(config)#interface loopback 0
R3(config-if)#ip address 3.3.3.3 255.255.255.255
R3(config-if)#exit
R3(config)#int e0/0
R3(config-if)#ip address 10.0.1.3 255.255.255.0
R3(config-if)#no shut
R3(config-if)#exit
R3(config)#interface e0/1
R3(config-if)#ip address 192.168.2.3 255.255.255.0
R3(config-if)#no shut
R3(config-if)#exit
R3(config)#router ospf 1
R3(config-router)#network 3.3.3.0 0.0.0.255 area 0
R3(config-router)#network 10.0.1.0 0.0.0.255 area 0

R3(config-router)#exit
R3(config)#mpls label range 300 399
R3(config)#mpls label protocol ldp
R3(config)#mpls ldp router-id loopback 0
R3(config)#int e0/0
R3(config-if)#mpls ip
R3(config)#ip vrf A-2
R3(config-vrf)#rd 500:1
R3(config-vrf)#route-target import 500:1
R3(config-vrf)#route-target export 500:1
R3#sh ip vrf Name Default RD Interfaces A-2 500:1
R3#sh ip vrf detail

VRF A-2 (VRF Id = 1);
default RD 500:1; default VPNID Old CLI format, supports IPv4 only Flags: 0xC No interfaces Address family ipv4 unicast (Table ID = 0x1): Flags: 0x0 Export VPN route-target communities RT:500:1 Import VPN route-target communities RT:500:1 No import route-map No global export route-map No export route-map VRF label distribution protocol: not configured VRF label allocation mode: per-prefix R3(config)#int e0/1 R3(config-if)#ip vrf forwarding A-2 % Interface Ethernet0/1 IPv4 disabled and address(es) removed due to enabling VRF A-2
R3(config-if)#ip address 192.168.2.3 255.255.255.0
R3(config-if)#end
R3#sh ip route vrf A-2 Routing Table: A-2
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks C 192.168.2.0/24 is directly connected, Ethernet0/1 L 192.168.2.3/32 is directly connected, Ethernet0/1

R3#sh ip vrf Name Default RD Interfaces A-2 500:1 Et0/1
R3(config)#router ospf 10 vrf A-2
R3(config-router)#network 192.168.2.0 0.0.0.255 area 0 R3(config-router)#end R3#sh ip ospf 10 neighbor Neighbor ID Pri State Dead Time Address Interface 5.5.5.5 1 FULL/DR 00:00:33 192.168.2.5 Ethernet0/1
R3#sh ip route vrf A-2 ospf
Routing Table: A-2
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS

summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
5.0.0.0/32 is subnetted, 1 subnets O 5.5.5.5 [110/11] via 192.168.2.5, 00:06:37, Ethernet0/1
R3(config)#router bgp 500
R3(config-router)#no bgp default ipv4-unicast R3(config-router)#neighbor 1.1.1.1 remote-as 500
R3(config-router)#neighbor 1.1.1.1 update-source loopback 0 R3(config-router)#address-family vpnv4 unicast
R3(config-router-af)#neighbor 1.1.1.1 activate
R3(config-router-af)#neighbor 1.1.1.1 send-community extended
R3(config-router-af)#neighbor 1.1.1.1 next-hop-self
R3#sh ip bgp vpnv4 all
summary
BGP router identifier 3.3.3.3, local AS number 500 BGP table version is 1, main routing table version 1 Neighbor V AS MsgRcvd MsgSent TblVer InQ OutQ Up/Down State/PfxRcd 1.1.1.1 4 500 7 6 1 0 0 00:03:01
R3(config)#router bgp 500
R3(config-router)#address-family ipv4 vrf A-2
R3(config-router-af)#redistribute ospf 10 vrf A-2 match internal external 1 external 2
R3(config-router-af)#exit R
3(config-router)#exit
R3(config)#router ospf 10 vrf A-2
R3(config-router)#redistribute bgp 500 subnets
R3(config-router)#end
R3#sh ip bgp vpnv4 all
BGP table version is 7, local router ID is 3.3.3.3
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal, r RIB-failure, S Stale, m multipath, b backup-path, f RT-Filter, x best-external, a additional-path, c RIB-compressed, Origin codes: i - IGP, e - EGP, ? - incomplete RPKI validation codes: V valid, I invalid, N Not found
Network Next Hop Metric LocPrf Weight Path Route Distinguisher: 500:1 (default for vrf A-2)

*>i 4.4.4.4/32 1.1.1.1 11 100 0 ?
*> 5.5.5.5/32 192.168.2.5 11 32768 ?
*>i 192.168.1.0 1.1.1.1 0 100 0 ?
*> 192.168.2.0 0.0.0.0 0 32768 ?
R3#sh ip route vrf A-2
Routing Table: A-2
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF

NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
4.0.0.0/32 is subnetted, 1 subnets B 4.4.4.4 [200/11] via 1.1.1.1, 00:55:23 5.0.0.0/32 is subnetted, 1 subnets O 5.5.5.5 [110/11] via 192.168.2.5, 01:50:21,
Ethernet0/1 B 192.168.1.0/24 [200/0] via 1.1.1.1, 00:55:23 192.168.2.0/24 is variably subnetted, 2 subnets, 2 masks C 192.168.2.0/24 is directly connected,
Ethernet0/1 L 192.168.2.3/32 is directly connected, Ethernet0/1 R3#ping vrf A-2 5.5.5.5 T ype escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 5.5.5.5, timeout is 2 seconds: !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/1 ms
R4 Router>enable
Router#conf t
Router(config)#hostname R4
R4(config)#int loopback 0
R4(config-if)#ip address 4.4.4.4 255.255.255.255
R4(config-if)#exit
R4(config)#int e0/0
R4(config-if)#ip address 192.168.1.4 255.255.255.0
R4(config-if)#no shutdown
R4(config-if)#exit
R4(config)#router ospf 1
R4(config-router)#network 4.4.4.0 0.0.0.255 area 10
R4(config-router)#network 192.168.1.0 0.0.0.255 area 10
R4(config-router)#exit
R4#sh ip route ospf
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set
5.0.0.0/32 is subnetted, 1 subnets O IA 5.5.5.5 [110/21] via 192.168.1.1, 00:23:41,
Ethernet0/0 O IA 192.168.2.0/24 [110/11] via 192.168.1.1, 00:23:41,
Ethernet0/0 R4#ping 5.5.5.5 source lo 0
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 5.5.5.5, timeout is 2 seconds:
Packet sent with a source address of 4.4.4.4 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/1/2 ms R5
Router>enable

Router#conf t
Router(config)#hostname R5
R5(config)#int loopback 0

R5(config-if)#ip address 5.5.5.5 255.255.255.255
R5(config-if)#exit
R5(config)#int e0/0
R5(config-if)#ip address 192.168.2.5 255.255.255.
R5(config-if)#no shutdown
R5(config-if)#exit
R5(config)#router ospf 1
R5(config-router)#network 5.5.5.0 0.0.0.255 area 0
R5(config-router)#network 192.168.2.0 0.0.0.255 area 0 R5(config-router)#exit
R5#sh ip route ospf Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2 E1 - OSPF external type 1, E2 - OSPF external type 2 i - IS-IS, su - IS-IS
summary, L1 - IS-IS level-1, L2 - IS-IS level-2 ia - IS-IS inter area, * - candidate default, U - per-user static route o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP a - application route + - replicated route, % - next hop override
Gateway of last resort is not set 4.0.0.0/32 is subnetted, 1 subnets O IA 4.4.4.4 [110/21] via 192.168.2.3, 00:23:51, Ethernet0/0 O IA 192.168.1.0/24 [110/11] via 192.168.2.3, 00:23:51, Ethernet0/0
R5#ping 4.4.4.4 source lo 0
Type escape sequence to abort. Sending 5, 100-byte ICMP Echos to 4.4.4.4, timeout is 2 seconds: Packet sent with a source address of 5.5.5.5 !!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 2/2/3 ms

Practical 10

Aim: Simulating OpenFlow Using MININET.

Code:

-

To simulate an OpenFlow-based SDN network using Mininet.

Step-by-Step Procedure:

Step 1: Update and Install Mininet

Open Terminal and run:
sudo apt update
sudo apt install mininet -y
Alternatively, install from source:

git clone https://github.com/mininet/mininet.git
cd mininet
sudo ./util/install.sh -a

Step 2: Run a Basic OpenFlow Topology

Start a simple topology with 1 switch and 2 hosts:

sudo mn --topo=single,2 --controller=remote,ip=127.0.0.1 --switch ovsk

- This creates:
  - 1 Open vSwitch (ovsk)
  - 2 Hosts
  - Connected to a remote controller (yet to be started)

Step 3: Verify Network Connectivity

Inside the Mininet CLI (you'll see mininet> prompt):
mininet> pingall

- This will check connectivity between the hosts.

To show switch info:
mininet> sh ovs-vsctl show

Step 4: Install and Run the POX Controller

Open a new terminal (do not close the Mininet window)

git clone https://github.com/noxrepo/pox.git
cd pox
./pox.py forwarding.l2_learning

- This will start a Layer 2 learning switch controller.

Step 5: Run Mininet with POX Controller

In the original terminal (or a new one if needed), run this command:
sudo mn --topo=tree,depth=2 --controller=remote,ip=127.0.0.1,port=6633 --switch ovsk

- Creates a **tree topology** with depth 2.
- Connects to the POX controller on the default OpenFlow port 6633.

Step 6: Monitor OpenFlow Communication

Open a new terminal and run:
sudo tcpdump -i any port 6633

- This will show OpenFlow messages between the switch and controller.

Step 7: View Flow Rules on Switch

In Mininet CLI, check the switch name with:
mininet> sh ovs-vsctl show
Then, dump flow table (replace br0 with switch name if different):
sudo ovs-ofctl dump-flows br0

BUNTS SANGHA MUMBAI

ANNA LEELA COLLEGE OF COMMERCE & ECONOMICS

SHOBHA JAYARAM SHETTY COLLEGE FOR BMS

(UNIVERSITY OF MUMBAI)

Shashi Manmohan Shetty Higher Education Complex,

Buntara Bhavan Marg, Kurla (E), Mumbai – 400 070.

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

Exam. Seat No. _____

Date _____

CERTIFIED that the practical, and assignments duly signed, were performed by

Mr./Ms. _____

Roll No. _____ *of* _____*class in the Information Technology Laboratory of Anna Leela College of Commerce & Economics Shobha Jayaram Shetty College for BMS, Mumbai during the academic year 20* _____ *-20* _____

He/she has completed the course of laboratory assignments in Information Technology as contained in the course prescribed by the University of Mumbai.

Head of Dept.

Information Technology

Date _____

Sign. of the student

Date _____

Professor-in-charge

1) _____

Date _____

Sign. of Examiner's

1) _____

Date _____

M.Sc. IT Sem II **Computer Vision Practical.**

Sr. No	PRACTICAL	DATE	SIGN
1	Perform Geometric transformations		
	Perform Image Stitching		
	Perform Camera Calibration		
2	Perform the following: a. Face detection b. Object detection c. Pedestrian detection d. Face recognition		
	Construct a 3D model from images I		
	Implement object detection and tracking from video		
3	Perform Feature extraction using RANSAC		
	Perform Colorization		
4	Perform text detection and recognition		
	Perform Image matting and composting		

Practical No. 1

Part A:

Aim: Perform Geometric Transformations

Concept:

Geometric transformations are mathematical operations that move or manipulate objects in space. These transformations are fundamental concepts in fields like computer graphics, geometry, and physics. The primary aim is to alter the position, orientation, and scale of objects while preserving certain properties like angles, lengths, or areas.

Translation: Translation moves every point of an object in the same direction by the same distance. if T(x,y) represents the translation of a point P(x,y), the new coordinates after translation would be:
T(x,y)=(x+tx,y+ty), where tx and ty are the translation distances along the x and y axes, respectively.

Scaling: Scaling changes the size of an object, either enlarging or shrinking it, while maintaining its shape. If S(sx,sy) represents the scaling of a point P(x,y),the new coordinates after scaling are: S(sx,sy)=(x·sx, y·sy)

where sx and sy are the scaling factors along the x and y axes, respectively. **Rotation:** Involves turning an object around a fixed point(called the center of rotation) by a specified angle. if $R(\theta)$ represents the rotation of a point P(x,y) by an angle θ, the new coordinates after rotation are:

R(θ)=(x′ =xcos(θ)− ysin(θ),y=xsin(θ)+ycos(θ)), where θ is the angle of rotation.

Reflection: Reflection creates a mirror image of an object across a specific line, known as the line of reflection.

Shearing: Shearing involves shifting one part of an object while the other part stays fixed, resulting in a skewed appearance. A shear transformation along the x-axis is represented as:

Sh(x,y)=(x+shx·y,y) where $s\ h\ x$ is the shear factor in the x direction. A similar transformation can be applied along the y-axis. Sh(x,y)=(x,y +shy·x)

Packages Used :

numpy is a powerful library in Python used for working with numbers and arrays (lists of numbers). It stands for Numerical Python and is widely used in data science, machine learning, and scientific computing. NumPy is essential for anyone working with large amounts of numerical data in Python.

OpenCV (Open Source Computer Vision Library), often imported in Python as cv2, is a powerful library used for computer vision tasks. It provides tools to work with images and videos, such as analyzing, processing, and manipulating visual data.

def translate(image, x, y):
M = np.float32([[1, 0, x], [0, 1, y]])
translated_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0])) return translated_image

Translates the image, i.e., shifts the image by x pixels along the horizontal axis and y pixels along the vertical axis.

- image : The input image that you want to translate (a NumPy array).
- M : The transformation matrix M defines the translation in terms of the affine transformation.

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$$

- x : The number of pixels to shift the image along the x-axis (horizontal direction).
- y : The number of pixels to shift the image along the y-axis (vertical direction).
- The cv2.warpAffine() function applies this matrix to the image, which moves every pixel by x and y.

def rotate(image, angle, center=None, scale=1.0):
(h, w) = image.shape[:2] if
center is None:
center = (w // 2, h // 2)
M = cv2.getRotationMatrix2D(center, angle, scale)
rotated_image = cv2.warpAffine(image, M, (w, h)) return rotated_image

Rotates the image by a specified angle around a center point. You can also scale the image during rotation.

- image : The input image to rotate (NumPy array).
- angle : The angle (in degrees) by which to rotate the image. A positive value rotates counterclockwise, and a negative value rotates clockwise.
- center : The point around which to rotate. It defaults to the center of the image (None means the center will be used).
- scale : A scaling factor. 1.0 means no scaling (same size), values greater than 1 scale the image up, and values between 0 and 1 scale it down.
- M : The transformation matrix M defines the rotation in terms of the affine transformation.

$$M = \begin{bmatrix} \square\square\square\square & \square\square\square\square & 0 \\ -\square\square\square\square & \square\square\square\square & 0 \end{bmatrix}$$

$For\ Clockwise\ Rotation,\ -\theta\ is\ negative\ (-\theta)$
$For\ Anti-clockwise\ Rotation,\ -\theta\ is\ positive\ (\theta)$

- cv2.getRotationMatrix2D() generates the transformation matrix for rotation. It takes the center, angle, and scale as inputs.
- The cv2.warpAffine() function then applies this transformation to the image, resulting in a rotated version.

def scale(image, scale_x, scale_y):
scaled_image = cv2.resize(image, None, fx=scale_x, fy=scale_y) return scaled_image
Scales the image by different factors along the x and y axes.

- image : The input image (NumPy array).
- scale_x : The scaling factor along the x-axis (horizontal).
- scale_y : The scaling factor along the y-axis (vertical).
- M : The transformation matrix M defines the scaling in terms of the affine transformation.

$$M = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- cv2.resize() changes the size of the image based on the given scaling factors fx and fy.

def shear(image, shear_x, shear_y):
M = np.float32([[1, shear_x, 0], [shear_y, 1, 0]])
sheared_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0])) return sheared_image
Shears the image along the x and y axes. Shearing is a transformation that slants the image along one direction.

- image : The input image to shear (NumPy array).
- shear_x : The shear factor along the x-axis. This value determines how much the image will shift horizontally based on the vertical position of each pixel.
- shear_y : The shear factor along the y-axis. This value determines how much the image will shift vertically based on the horizontal position of each pixel.
- M : The transformation matrix M defines the shearing in terms of the affine transformation.

$$For\ X-Shear\ (M) = \begin{bmatrix} 1 & 0 & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$For\ Y-Shear\ (M) = \begin{bmatrix} 1 & sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- cv2.imread('img.jpg') : Reads the image from the file 'img.jpg' (make sure the image exists).
- The image undergoes translation, rotation, scaling, and shearing using the functions defined earlier.
- cv2.imshow() : Displays the original and transformed images in separate windows.
- cv2.waitKey(0) : Waits indefinitely for a key press to close the image windows.

reflected_x = cv2.flip(image, 0)

- The second argument (0) specifies that the image should be flipped along the X-axis.
- This results in a vertical flip, meaning the top and bottom parts of the image are swapped.

reflected_y = cv2.flip(image, 1)

- The second argument (1) specifies that the image should be flipped along the Y-axis.
- This results in a horizontal flip, meaning the left and right parts of the image are swapped.

reflected_origin = cv2.flip(image, -1)

- The second argument (-1) flips the image along both axes (X and Y).
- This results in a diagonal flip, meaning the image is rotated 180 degrees (upside-down and mirrored).

Part B:
Aim: Perform Image Stitching

Concept:
Image stitching is the process of merging multiple overlapping images to create a seamless, larger image, commonly used in panoramic photography, medical imaging, and satellite mapping. The process involves capturing images with overlapping regions, detecting key features using techniques like SIFT or ORB, matching these features between images, and aligning them using homography estimation. Once aligned, blending techniques such as multi-band blending or feathering are applied to smooth out seams and color differences. The final result is a unified image with minimal distortions. Image stitching is widely used in applications like virtual reality, geographic mapping, and surveillance systems, enabling the creation of high-resolution and immersive visuals.

- imutils.paths: Helps in listing image file paths from a given directory.
- imutils: A utility library for image processing tasks.
- cv2: OpenCV, used for image processing.
- argparse: Used for command-line argument parsing (not utilized in this script)

imagePaths = sorted(list(paths.list_images('P1'))) #'P1' is a folder containing images

- paths.list_images('P1') collects all image file paths in the folder 'P1'.
- list() converts the generator output into a list.
- sorted() arranges the image paths in ascending order to ensure they are processed in sequence.

images=[] : An empty list images is created to store loaded images. **for imagePath in imagePaths:**

```
    image = cv2.imread(imagePath)
    images.append(image)
```

- A for loop iterates through each imagePath.
- cv2.imread(imagePath): Reads the image from the file.
- The loaded image is appended to the images list.

stitcher = cv2.createStitcher() if imutils.is_cv3() else cv2.Stitcher_create():

- imutils.is_cv3() checks whether OpenCV version 3 is being used:
- If OpenCV 3, it uses cv2.createStitcher().
- Otherwise, it uses cv2.Stitcher_create().
- The stitcher object is responsible for combining multiple images into a panorama.

(status, stitched) = stitcher.stitch(images):

- stitcher.stitch(images): Attempts to combine the images into a panoramic image.
- Returns:
- status: Indicates if stitching was successful (0 = success, non-zero = failure).
- stitched: The resulting stitched image if successful.

Part C:
Aim: Perform Camera Calibration

Concept:
Camera calibration is the process of determining a camera's intrinsic (internal) and extrinsic (external) parameters to correct distortions and accurately map 3D world coordinates to 2D image coordinates. It is essential for applications like computer vision, robotics, augmented reality (AR), and 3D reconstruction.
Need for Camera Calibration

- Lens Distortion Correction: Lenses introduce distortions (barrel, pincushion), which calibration corrects.
- Accurate Depth Perception: Helps reconstruct real-world distances from images.
- Robust Object Detection: Improves precision in object tracking and recognition.
- Augmented Reality (AR): Ensures virtual objects align correctly with real-world scenes.

Packages Used :
The **os** module for interacting with the operating system, typically used for file and directory operations.
The **glob** module in Python is a powerful utility used for file name pattern matching. Its primary purpose is to allow users to retrieve files with a certain name pattern, without having to manually specify each file.

CHECKERBOARD = (6, 9)
The CHECKERBOARD variable defines the number of **internal corners** in the checkerboard. The first value (6) is the number of internal corners along the rows, and the second value (9) is the number of internal corners along the columns. These internal corners (not squares) are used for calibration.

criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)
This sets the termination criteria for the corner subpixel refinement process.

- cv2.TERM_CRITERIA_EPS: The stopping criterion based on the change in corner positions.
- cv2.TERM_CRITERIA_MAX_ITER: The stopping criterion based on the maximum number of iterations.

The refinement method iterates over the corner positions until one of the following criteria is met:

- EPS (0.001): The algorithm stops if the corner positions stop changing by more than 0.001.
- MAX_ITER (30): The algorithm stops after a maximum of 30 iterations.

threedpoints = [ ]
A list to store the real-world coordinates of the detected corners in 3D space.

twodpoints = [ ]
A list to store the detected 2D pixel coordinates of the corners in each image.

objectp3d = np.zeros((1, CHECKERBOARD[0] * CHECKERBOARD[1], 3), np.float32)
objectp3d[0,:,:2] = np.mgrid[0:CHECKERBOARD[0],0:CHECKERBOARD[1]].T.reshape(-1, 2)
This creates an array to store the 3D real-world coordinates of the checkerboard corners. Since the checkerboard is flat, the Z-coordinates are set to 0 for all corners. The mgrid function creates a grid of corner positions in the X and Y directions.

images = glob.glob('.jpg')
This retrieves all .jpg files in the current directory. These images will be used to extract the checkerboard corners for camera calibration.

image = cv2.imread(filename)
grayColor = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
Reads each image file and converts it to grayscale. Grayscale images are easier to process when detecting corners.

- cv2.imread() : Reads the image from disk.
- cv2.cvtColor() : Converts the image to grayscale from BGR (since OpenCV reads images in BGR format).

ret, corners = cv2.findChessboardCorners(grayColor, CHECKERBOARD, cv2.CALIB_CB_ADAPTIVE_THRESH + cv2.CALIB_CB_FAST_CHECK + cv2.CALIB_CB_NORMALIZE_IMAGE)
This method detects the internal corners of the checkerboard pattern.

- grayColor : The grayscale image in which the corners will be detected.
- CHECKERBOARD : The expected number of internal corners (rows x columns).
- cv2.CALIB_CB_flags : These flags help improve the corner detection by enabling adaptive thresholding, fast checking, and image normalization.

if ret == True:

threedpoints.append(objectp3d)

corners2 = cv2.cornerSubPix(grayColor, corners, (11, 11), (-1, -1), criteria) twodpoints.append(corners2)

image = cv2.drawChessboardCorners(image, CHECKERBOARD, corners2, ret) If the corners are detected, the code refines the corner positions to subpixel accuracy and stores the results.

- cv2.cornerSubPix() : Refines the corner positions by iterating over them and adjusting their subpixel accuracy using the provided termination criteria (criteria).
- cv2.drawChessboardCorners() : Draws the refined corner positions on the image.

cv2.imshow('img', image) cv2.waitKey(0)

Displays the image with drawn corners.

- cv2.imshow() : Displays the image in a window.
- cv2.waitKey() : Waits for a key press to proceed (0 means wait indefinitely).

ret, matrix, distortion, r_vecs, t_vecs = cv2.calibrateCamera(threedpoints, twodpoints, grayColor.shape[::-1], None, None)

This method calibrates the camera using the 3D object points (threedpoints) and the corresponding 2D image points (twodpoints).

- threedpoints : List of 3D object coordinates from the checkerboard.
- twodpoints : List of detected 2D coordinates corresponding to the 3D object points.
- grayColor.shape[::-1] : The dimensions of the image (height, width) in reverse order.
- r_vecs: Rotation vectors (camera orientation).
- t_vecs: Translation vectors (camera position).

print(" Camera matrix:")
print(matrix)
print("\n Distortion coefficient:") print(distortion)
print("\n Rotation Vectors:")
print(r_vecs)
print("\n Translation Vectors:") print(t_vecs)

Prints out the calibration results, including the camera matrix, distortion coefficients, rotation vectors, and translation vectors.

Explanation of Output:

Camera matrix:

[[58.17561698 0. 102.67813943]
[0. 60.45272444 93.55060487]
[0. 0. 1.]]

- Focal Lengths: fx=58.18, fy =60.45 (in pixels) – these scale the image from real-world dimensions.
- Principal Point: (cx=102.68,cy=93.55) – this is the optical center of the camera sensor.
- Skew: The 0 in the second column indicates no skew (ideal camera).

Distortion coefficient:

[[0.02229387 -0.02371451 0.00047977 -0.00194139 0.00608793]]

These values represent radial and tangential distortions:

- Radial Distortion (k1,k2 ,k3): Causes straight lines to appear curved.
- Tangential Distortion (p1,p2): Caused by lens misalignment.
- Small values indicate minimal distortion.

Rotation Vectors:

(array([[-0.13607382], [
0.08651059],
[1.49228225]]),)

Defines how the camera is rotated in 3D space relative to the calibration pattern.

Translation Vectors:

(array([[3.51290647],
[-3.70110923],
[3.98457147]]),)

- Indicates the camera' s position relative to the calibration object in real-world units (e.g., meters).
- **Positive x-value**: Camera is shifted to the right.
- **Negative y-value**: Camera is slightly below the reference plane.
- **Positive z-value**: Camera is in front of the reference plane.

Practical No. 1

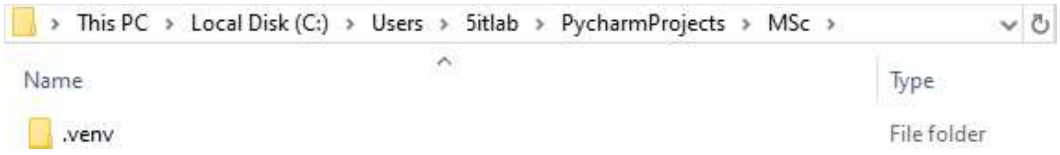
Part A
Aim: Perform Geometric Transformation Packages
Used: numpy, cv2
Command:

pip install numpy

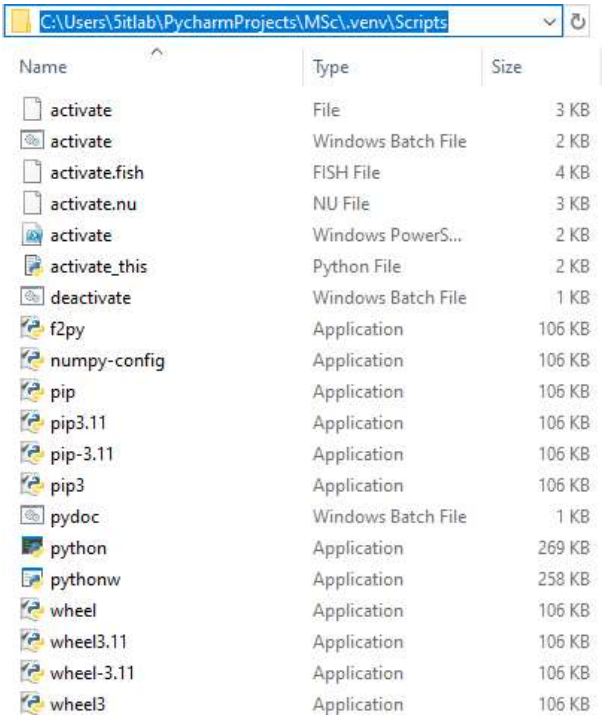
On Pycharm Terminal

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install numpy
Collecting numpy
  Downloading numpy-2.2.1-cp311-cp311-win_amd64.whl.metadata (60 kB)
Downloading numpy-2.2.1-cp311-cp311-win_amd64.whl (12.9 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.9/12.9 MB 4.3 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-2.2.1
```

Note: If an error is found in installing packages from Pycharm Terminal, then open the project in the File Explorer.

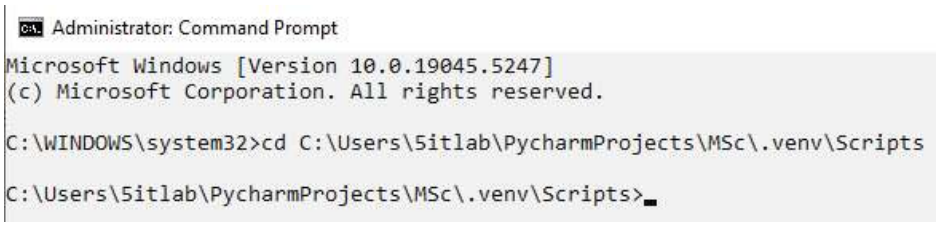


Open the **Scripts** folder and copy the path



Open a command propt with administrative privileges, and change the working directory using the following command:

cd *path*



Try using the command again

```
C:\Users\5itlab\PycharmProjects\MSc\.venv\Scripts>pip install numpy
Collecting numpy
  Using cached numpy-2.2.1-cp311-cp311-win_amd64.whl.metadata (60 kB)
Using cached numpy-2.2.1-cp311-cp311-win_amd64.whl (12.9 MB)
Installing collected packages: numpy
Successfully installed numpy-2.2.1
```

pip install opencv-python

On Pycharm Terminal

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install opencv-python
Collecting opencv-python
  Using cached opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl.metadata (20 kB)
Requirement already satisfied: numpy>=1.21.2 in c:\users\5itlab\pycharmprojects\msc\.venv\lib\site-packages
(from opencv-python) (2.2.1)
Using cached opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl (38.8 MB)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.10.0.84
```

On Command Propt (Administrative Privileges)

```
C:\Users\5itlab\PycharmProjects\MSc\.venv\Scripts>pip install opencv-python
Collecting opencv-python
  Using cached opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl.metadata (20 kB)
Requirement already satisfied: numpy>=1.21.2 in c:\users\5itlab\pycharmprojects\msc\.venv\lib\site-packages (from opencv
-python) (2.2.1)
Using cached opencv_python-4.10.0.84-cp37-abi3-win_amd64.whl (38.8 MB)
Installing collected packages: opencv-python
Successfully installed opencv-python-4.10.0.84
```

Images Used:
img.jpeg, moon.jpg

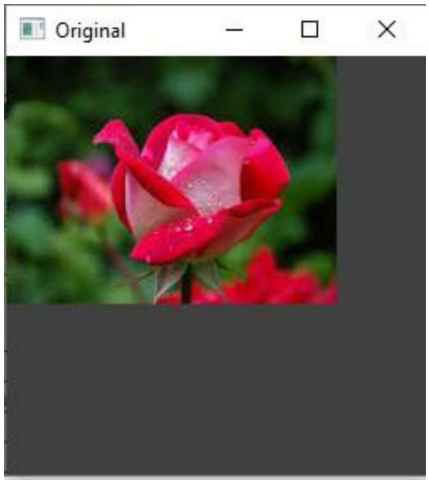
Code (Translation, Rotation, Scaling and Shearing):

```
import numpy as np import
cv2

def translate(image, x, y): M =
  np.float32([
    [1, 0, x],
    [0, 1, y]])
  translated_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0])) return translated_image

def rotate(image, angle, center=None, scale=1.0): (h, w) =
  image.shape[:2]
  if center is None:
     center = (w // 2, h // 2)
  M = cv2.getRotationMatrix2D(center, angle, scale)
  rotated_image = cv2.warpAffine(image, M, (w, h)) return
  rotated_image

def scale(image, scale_x, scale_y):
  scaled_image = cv2.resize(image, None, fx=scale_x, fy=scale_y) return
  scaled_image

def shear(image, shear_x, shear_y): M =
  np.float32([
    [1, shear_x, 0],
    [shear_y, 1, 0]])
```

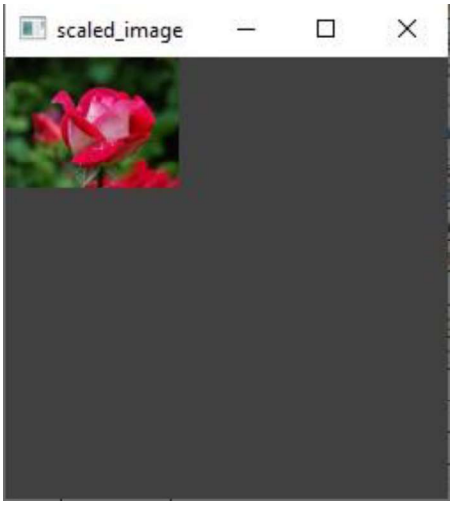
```
    sheared_image = cv2.warpAffine(image, M, (image.shape[1], image.shape[0])) return
    sheared_image
image = cv2.imread('img.jpg') #Main
Logic
translated_image = translate(image, 100, 100)
rotated_image = rotate(image, 45) scaled_image =
scale(image, 0.5, 0.5)
sheared_image = shear(image, 0.2, 0.3)

#Display of images cv2.imshow('Original',image)
cv2.imshow('translated_image',translated_image) cv2.imshow('rotated_image',rotated_image)
cv2.imshow('scaled_image',scaled_image) cv2.imshow('sheared_image',sheared_image)

#Resize Images cv2.resizeWindow('Original',250,250)
cv2.resizeWindow('translated_image',250,250)
cv2.resizeWindow('rotated_image',250,250)
cv2.resizeWindow('scaled_image',250,250)
cv2.resizeWindow('sheared_image',250,250)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:

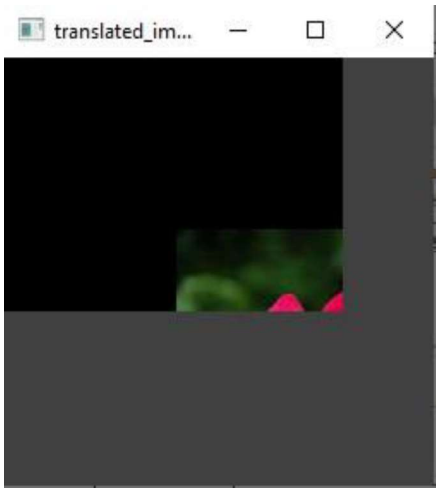
Original



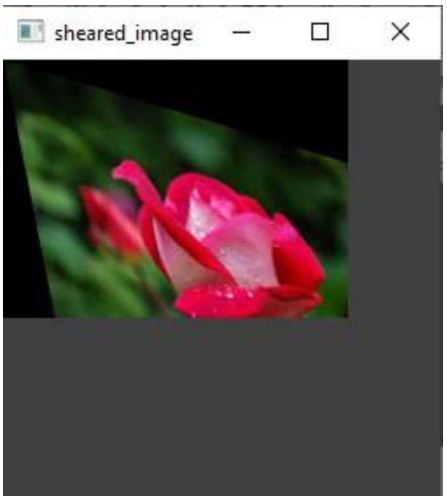
scaled_image



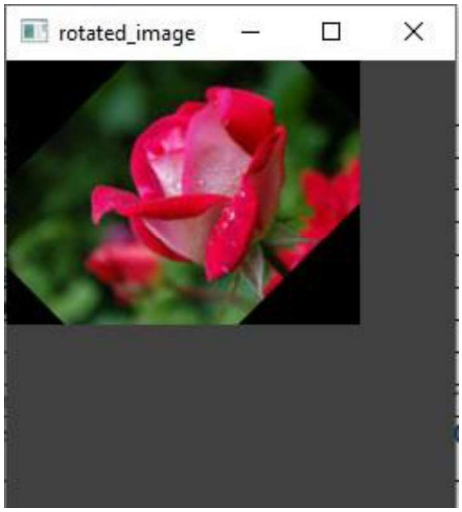
translated_image



sheared_image



rotated_image

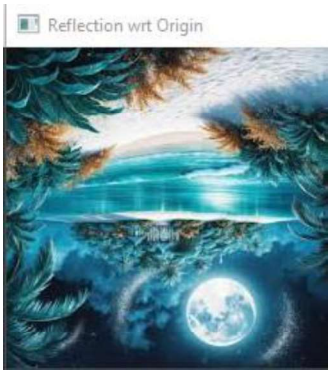


Code (Reflection):

```
import cv2

# Load the image
image = cv2.imread("moon.jpg")

# Reflection w.r.t. X-axis (vertical flip) reflected_x =
cv2.flip(image, 0)
# Reflection w.r.t. Y-axis (horizontal flip)
reflected_y = cv2.flip(image, 1)
# Reflection w.r.t. Origin (both axes flip)
reflected_origin = cv2.flip(image, -1)

# Display the results cv2.imshow("Original
Image",image)
cv2.imshow("Reflection wrt X",reflected_x)
cv2.imshow("Reflection wrt Y",reflected_y)
cv2.imshow("Reflection wrt Origin",reflected_origin)

#Resize Images cv2.resizeWindow('Original
Image',350,350)
cv2.resizeWindow('Reflection wrt X',350,350)
cv2.resizeWindow('Reflection wrt Y',350,350)
cv2.resizeWindow('Reflection wrt Origin',350,350)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:


Reflection wrt Y


Reflection wrt Origin

Part B
Aim: Perform Image Stitching Steps:
Packages Used: imutils, cv2
Command:
pip install imutils

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install imutils
Collecting imutils
  Using cached imutils-0.5.4-py3-none-any.whl
Installing collected packages: imutils
Successfully installed imutils-0.5.4
```

pip install argparse

```
(.venv) PS C:\Users\5itlab\PycharmProjects\01_MHD_MSc_CVP> pip install argparse
Collecting argparse
  Using cached argparse-1.4.0-py2.py3-none-any.whl.metadata (2.8 kB)
Using cached argparse-1.4.0-py2.py3-none-any.whl (23 kB)
Installing collected packages: argparse
Successfully installed argparse-1.4.0
```

Images Used:
P1>>building1.jpg, building2.jpg, building3.jpg, building4.jpg, building5.jpg P2>>p21.png, p22.png, p23.png
P3>>001.jpg, 002.jpg, 003.jpg, 004.jpg, 005.jpg **Code:**

```
from imutils import paths
import imutils
import cv2 import
argparse
imagePaths = sorted(list(paths.list_images('P1'))) #'P..' is folder containing images images=[]
# loop over the image paths, load each one, and add them to our
# images to stitch list
for imagePath in imagePaths:
        image = cv2.imread(imagePath)
        images.append(image)
```

```
# initialize OpenCV's image stitcher object and then perform the image # stitching
print("[INFO] stitching images...")
#stitcher = cv2.Stitcher_create(cv2.Stitcher_PANORAMA)
stitcher = cv2.createStitcher() if imutils.is_cv3() else cv2.Stitcher_create() (status, stitched) =
stitcher.stitch(images)

# if the status is '0', then OpenCV successfully performed image # stitching
if status == 0:
        # write the output stitched image to disk
        cv2.imwrite("P1Stitched.jpg", stitched)
        # display the output stitched image to our screen
        cv2.imshow("Stitched", stitched)
        cv2.waitKey(0)
else:
        print("[INFO] image stitching failed ({})".format(status))
```

Output: P1Stitched.jpg

P2Stitched.jpg

P3Stitched.jpg

Note: Output image is also stored in the folder

Part C
Aim: Perform Camera Calibration

Steps:
Packages Used: numpy, cv2, os, glob

Images Used:
p3.jpg
p32.jpg
p33.jpg

Code:

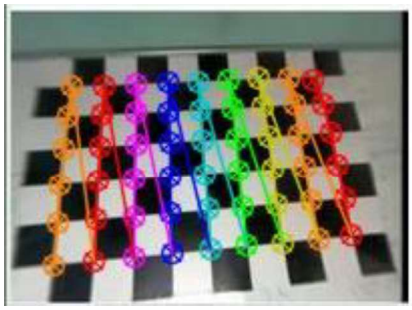
```
import cv2
import numpy as np import os
import glob

# Define the dimensions of checkerboard
CHECKERBOARD = (6, 9)

# stop the iteration when specified #
accuracy, epsilon, is reached or
# specified number of iterations are completed.
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)

# Vector for 3D points
threedpoints = []

# Vector for 2D points
```

```
twodpoints = []

# 3D points real world coordinates
objectp3d = np.zeros((1, CHECKERBOARD[0] * CHECKERBOARD[1], 3), np.float32) objectp3d[0, :,  :2]
                        =          np.mgrid[0:CHECKERBOARD[0],
0:CHECKERBOARD[1]].T.reshape(-1, 2)
prev_img_shape = None

# Extracting path of individual image stored # in a
given directory. Since no path is
# specified, it will take current directory # jpg
files alone
images = glob.glob('*.jpg')

for filename in images:
        image = cv2.imread(filename)
        grayColor = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

        # Find the chess board corners
        # If desired number of corners are # found
        in the image then ret = true
        ret, corners = cv2.findChessboardCorners( grayColor,
    CHECKERBOARD, cv2.CALIB_CB_ADAPTIVE_THRESH
                + cv2.CALIB_CB_FAST_CHECK + cv2.CALIB_CB_NORMALIZE_IMAGE)

        # If desired number of corners can be detected then, # refine the
        pixel coordinates and display
        # them on the images of checker board if ret ==
        True:
                threedpoints.append(objectp3d)

                # Refining pixel coordinates # for
                given 2d points. corners2 =
                cv2.cornerSubPix(
                        grayColor, corners, (11, 11), (-1, -1), criteria)

                twodpoints.append(corners2)


                # Draw and display the corners
                image = cv2.drawChessboardCorners(image,          CHECKERBOARD,
corners2, ret)
```

```
        cv2.imshow('img', image) cv2.waitKey(0)

cv2.destroyAllWindows() h, w
= image.shape[:2]
# Perform camera calibration by
# passing the value of above found out 3D points (threedpoints) # and its
corresponding pixel coordinates of the
# detected corners (twodpoints)
ret, matrix, distortion, r_vecs, t_vecs = cv2.calibrateCamera( threedpoints,
        twodpoints, grayColor.shape[::-1], None, None)

# Displaying required output
print(" Camera matrix:")
print(matrix)

print("\n Distortion coefficient:") print(distortion)

print("\n Rotation Vectors:") print(r_vecs)

print("\n Translation Vectors:") print(t_vecs)
```

Output:

Case – I (p3.jpg):



Camera matrix:

```
[[ 58.17561698   0.        102.67813943]
[  0.         60.45272444  93.55060487]
[  0.          0.          1.        ]]
```

Distortion coefficient:

[[0.02229387 -0.02371451 0.00047977 -0.00194139 0.00608793]]

Rotation Vectors:
(array([[-0.13607382], [
0.08651059],
[1.49228225]]),)

Translation Vectors:
(array([[3.51290647],
[-3.70110923],
[3.98457147]]),)

Case – II (p32.jpg):

Camera matrix:
[[8.8650501 0. 75.00007197]
[0. 8.86817689 99.50002698]
[0. 0. 1.]]

Distortion coefficient:
[[1.32152374e-04 -2.46703289e-06 -5.41268873e-05 -2.85386206e-05 1.27621105e-08]]

Rotation Vectors:
(array([[-0.03259909], [-
0.00766796],
[3.07373454]]),)

Translation Vectors:
(array([[2.9027577],
[3.72600396],
[0.60404124]]),)

Case – III (p33.jpg):

Camera matrix:
[[28.92668812 0. 57.87918422]
[0. 30.55971048 103.85696559]
[0. 0. 1.]]

Distortion coefficient:
[[3.21879713e-03 -5.56108458e-04 1.09806782e-04 1.98033459e-04 2.63324031e-05]]

Rotation Vectors:
(array([[-0.01773896], [
0.07318732],
[-0.05676499]]),)

Translation Vectors:
(array([[-1.47326778], [-
4.01274967],
[2.08423295]]),

Practical No. 2

Part A.1:

Aim: Perform the Face Detection

Concept:

Face detection is the process of identifying human faces in an image or video. It is a fundamental step in various applications like facial recognition, security systems, emotion analysis, and augmented reality.
The code implements **face detection** using the **Haar Cascade Classifier** in OpenCV
which is computationally lightweight and suitable for real-time applications. However, it has limitations, such as sensitivity to lighting conditions and difficulty detecting faces at extreme angles.
This algorithm is based on **Viola-Jones object detection**, which uses Haar-like features to identify patterns in an image. The process can be broken down into several key steps.

1. **Image Preprocessing**
   The first step is to read the input image using cv2.imread() and convert it to grayscale using cv2.cvtColor(). The grayscale image simplifies computation since color information is not necessary for face detection. This step enhances processing speed because grayscale images require only a **single channel** instead of three (RGB).
1. **Loading the Pre-trained Haar Cascade Classifier**
   The classifier is loaded using cv2.CascadeClassifier(), which loads the **pre-trained Haar cascade XML file** (haarcascade_frontalface_default.xml). This file contains data trained on multiple face images, enabling the classifier to recognize frontal faces effectively.
2. **Detecting Faces in the Image**
   The function detectMultiScale() scans the grayscale image to detect faces. It takes several parameters:
   - scaleFactor=1.01: This parameter determines how much the image size is reduced at each scale. A smaller value increases accuracy but slows down processing.
   - minNeighbors=2: Specifies how many neighbors a detected region must have to be considered a face. Lower values detect more objects but increase false positives.
   - minSize=(5,5): Defines the minimum face size to be detected, helping to filter out very small objects.

   The function returns a list of detected faces in the form of bounding box coordinates (x, y, w, h), where (x, y) represents the top-left corner, and (w, h) represent the width and height of the detected face.

3. **Drawing a Bounding Box Around Faces**
   For each detected face, a green rectangle is drawn using cv2.rectangle(), which takes the starting and ending coordinates (x, y, x+w, y+h), the color (0, 255, 0) (green), and the thickness of the rectangle (4 pixels). This visualizes the detected faces in the image.
4. **Displaying the Image with Detected Faces**
   The image is converted from **BGR to RGB** using cv2.cvtColor(), as OpenCV loads images in **BGR format**, whereas most display libraries (like Matplotlib) use **RGB format**. Finally, cv2.imshow() is used to display the image, and cv2.waitKey(0) ensures the window remains open until a key is pressed.

Packages Used:
cv2 is the OpenCV library used for image processing tasks like loading, manipulating, and displaying images.

img = cv2.imread('C:/Users/5itlab/Downloads/input4.jpeg')
This line reads an image from the file path provided and stores it in the variable img.

- 'C:/Users/aditi/Downloads/input4.jpeg' : The file path to the image that needs to be loaded. It could be any valid image path.

gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
This converts the color image to grayscale. Face detection is often easier and faster in grayscale, as it reduces the complexity of the image (removing color information).

- img : The original image (in BGR color format).
- cv2.COLOR_BGR2GRAY : This is the color conversion code that tells OpenCV to convert the image from BGR (Blue-Green-Red) to grayscale.

face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_frontalface_default.xml ")
This initializes a Haar Cascade Classifier for face detection. OpenCV comes with pre-trained models for various objects like faces, eyes, etc.

- cv2.data.haarcascades : This is a predefined path to OpenCV's directory of Haar cascade files.
- "haarcascade_frontalface_default.xml" : This is the XML file that contains the trained data for detecting frontal faces.

face = face_classifier.detectMultiScale(gray_image, scaleFactor=1.01, minNeighbors=2, minSize=(5, 5))
This method detects faces in the given grayscale image.

- gray_image : The image on which face detection is applied (in grayscale).
- scaleFactor=1.01 : This compensates for faces appearing larger or smaller due to scaling in the image. It adjusts the image size at each scale.

- minNeighbors=2 : This parameter defines how many neighbors each rectangle should have to retain it. Higher values result in fewer detections but more accurate ones.
- minSize=(5, 5) : Specifies the minimum size of the detected face (width, height). Faces smaller than this size are ignored.

for (x, y, w, h) in face:
cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 4)
This loop draws rectangles around each detected face in the image.

- img : The image on which to draw the rectangle (this modifies the image directly).
- (x, y) : The top-left corner of the rectangle.
- (x + w, y + h) : The bottom-right corner of the rectangle, which is calculated by adding the width (w) and height (h) to the top-left corner.
- (0, 255, 0) : This is the color of the rectangle in BGR format (green in this case).
- 4 : This specifies the thickness of the rectangle's border.

img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
This line converts the image from BGR color format (used by OpenCV) to RGB color format (commonly used in matplotlib for display purposes).

- img : The original image in BGR format.
- cv2.COLOR_BGR2RGB : This conversion code tells OpenCV to convert from BGR to RGB.

cv2.imshow('i', img_rgb)
cv2.waitKey(0)
This part of the code displays the image with the detected faces and rectangles on the screen using OpenCV's imshow() function.

Part A.2:
Aim: Perform Object Detection

Concept :
In this practical we perform stop sign detection using OpenCV' s Haar Cascade Classifier. This approach is based on pattern recognition using pre-trained features from the stop_data.xml file. The algorithm follows a systematic process to identify and highlight stop signs in an image.

1. **Image Preprocessing**
   The first step involves loading the input image (stop.png) using cv2.imread(). Since OpenCV loads images in BGR format, they are converted to RGB using cv2.cvtColor(img, cv2.COLOR_BGR2RGB). Additionally, the image is converted to

grayscale (cv2.COLOR_BGR2GRAY), as grayscale images reduce computational complexity while preserving important features for detection.

2. **Loading the Pre-trained Haar Cascade Classifier**
   The classifier is initialized using cv2.CascadeClassifier('stop_data.xml'), which loads a pre-trained Haar cascade file for detecting stop signs. This XML file contains pre-extracted Haar-like features based on multiple positive and negative stop sign images.
3. **Detecting Stop Signs in the Image**
   The function detectMultiScale() is applied to the grayscale image. It detects multiple stop signs and returns their coordinates. The parameter minSize=(20,20) ensures that very small objects (noise) are ignored, focusing only on objects that are at least 20× 20 pixels in size.
   - If no stop signs are detected, found remains empty, and the program does nothing.
   - If one or more stop signs are detected, their bounding box coordinates (x, y, width, height) are stored in found.
4. **Drawing Bounding Boxes Around Detected Stop Signs**
   For each detected stop sign, a green rectangle is drawn using cv2.rectangle(). The parameters (x, y) define the top-left corner, while (x + height, y + width) define the bottom-right corner. The color (0, 255, 0) represents green, and the thickness is set to 5 pixels. This ensures that each detected stop sign is clearly highlighted.
5. **Displaying the Image with Detected Stop Signs**
   The processed image is displayed using cv2.imshow('img', img_rgb). The function plt.show() is included, but it is redundant because cv2.imshow() already handles the display. The program waits for user input (cv2.waitKey(0)) before closing the window.

Packages Used :

Pyplot is a module in the Matplotlib library, which is one of the most popular data visualization libraries in Python. Pyplot provides a MATLAB-like interface for creating static, animated, and interactive plots and graphs. It is specifically designed to work with 2D plotting, and it allows you to easily generate a wide range of plots and charts, including line plots, bar charts, histograms, scatter plots, and many others.

img = cv2.imread("stop.png")

The cv2.imread() method loads an image from the specified file path. In this case, it loads the image stop.png into the variable img.

img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

The cv2.cvtColor() method is used to convert an image from one color space to another. Here, it's used to convert the original BGR image into a grayscale version, which simplifies the detection process.

- img : This is the source image (in BGR format, which is the default for images loaded via OpenCV).
- cv2.COLOR_BGR2GRAY : This is a color conversion code that tells OpenCV to convert from BGR to grayscale.

img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

This method converts the BGR image to RGB format. OpenCV reads images in BGR by default, but many visualization libraries like Matplotlib expect images in RGB.

- img : This is the original BGR image loaded using cv2.imread().
- cv2.COLOR_BGR2RGB : The color conversion code to convert from BGR to RGB format.

stop_data = cv2.CascadeClassifier('stop_data.xml')

The cv2.CascadeClassifier() method loads a pre-trained classifier from an XML file. In this case, 'stop_data.xml' contains the data needed to recognize stop signs. OpenCV's Haar Cascade Classifier is commonly used for object detection in images.

- 'stop_data.xml' : A file containing a pre-trained classifier in XML format. It is usually generated using machine learning techniques to detect specific objects (e.g., stop signs).

found = stop_data.detectMultiScale(img_gray, minSize=(20, 20))

The detectMultiScale() method detects objects (in this case, stop signs) in an image by scanning the image at multiple scales. The method uses a sliding window approach to detect objects of various sizes.

- img_gray : The input image in grayscale. Object detection typically works better with grayscale images because they remove unnecessary color information, simplifying the detection process.
- minSize=(20, 20) : This argument defines the minimum size of the object to be detected. In this case, it's set to 20x20 pixels. Objects smaller than this size will be ignored to avoid detecting noise or irrelevant small objects.

for (x, y, width, height) in found:

    cv2.rectangle(img_rgb, (x, y), (x + width, y + height), (0, 255, 0), 5)

For each detected stop sign (bounding box), this code draws a green rectangle around it to highlight the detected object.

- img_rgb : The image in RGB format where the rectangle will be drawn.
- (x, y) : The top-left corner of the bounding box.
- (x + width, y + height) : The bottom-right corner of the bounding box, calculated by adding the width and height to the top-left coordinates.

- (0, 255, 0) : The color of the rectangle in RGB format (green in this case).
- 5 : The thickness of the rectangle's border.

Part A.3:
Aim: Perform Pedestrian Detection

Concept :
In this practical we perform pedestrian detection in an image using the Histogram of Oriented Gradients (HOG) descriptor with a pre-trained SVM (Support Vector Machine) classifier. This method is widely used for human detection in surveillance, autonomous vehicles, and crowd analysis.

1. **Initializing the HOG Descriptor for Person Detection**
   The first step involves initializing the HOG descriptor, a feature extraction technique that captures object shapes and edge orientations. The hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector()) function loads a pre-trained SVM model specifically trained to detect humans. This combination of HOG + SVM is highly effective for pedestrian detection.
2. **Reading and Resizing the Image**
   The input image (people2.jpeg) is loaded using cv2.imread(). Since larger images slow down processing, the image is resized using imutils, ensuring the width does not exceed 400 pixels (imutils.resize()). This helps optimize performance without significantly impacting detection accuracy.
3. **Detecting Pedestrians in the Image**
   The function hog.detectMultiScale() is used to detect pedestrians in the image. It returns bounding box coordinates of detected persons. Several key parameters control detection accuracy:
   - winStride=(4,4) – Defines the step size for the sliding window; smaller values increase accuracy but slow down processing.
   - padding=(10,10) – Adds extra pixels around detected regions to improve detection in cluttered environments.
   - scale=1.05 – Determines how much the image size is reduced at each step; a lower value increases accuracy but increases computation time.

   The function returns a list of regions, where each detected pedestrian is represented by (x, y, width, height).
4. **Drawing Bounding Boxes Around Detected Pedestrians**
   For each detected pedestrian, a red bounding box is drawn using cv2.rectangle(). The parameters (x, y) define the top-left corner, while (x + w, y + h) define the bottom-right corner. The color (0, 0, 255) represents red, and the rectangle thickness is set to 2 pixels. This visually highlights the detected pedestrians in the image.
5. **Displaying the Image with Detected Pedestrians**

The processed image is displayed using cv2.imshow("Image", image), and cv2.waitKey(0) ensures the window remains open until a key is pressed. Finally, cv2.destroyAllWindows() closes all OpenCV windows to free up memory.

Packages Used :

imutils is a Python library designed to provide convenience functions and utilities for image processing tasks, particularly when using OpenCV. While OpenCV itself is a powerful library for computer vision tasks, imutils is a lightweight, helper library that makes it easier to work with OpenCV by abstracting common and repetitive tasks into simple, easy-to-use functions.

hog = cv2.HOGDescriptor() hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

This part initializes the HOG (Histogram of Oriented Gradients) person detector in OpenCV. HOG is a feature descriptor used for object detection, and here, it's specifically used for detecting people (pedestrians).

- cv2.HOGDescriptor() : Creates an instance of the HOGDescriptor class. The HOG descriptor is responsible for extracting features from an image that can be used to detect objects (in this case, people).
- setSVMDetector() : This method sets the SVM (Support Vector Machine) detector for the HOG descriptor. It tells the HOG object which SVM to use for classifying the features as either a person or not.
- cv2.HOGDescriptor_getDefaultPeopleDetector() : This function provides a pre-trained SVM detector specifically for detecting people.

image = cv2.imread('people2.jpeg')

This line loads an image (people2.jpeg) into the variable image. It's the first step in processing the image for object detection.

image = imutils.resize(image, width=min(400, image.shape[1]))

This line resizes the image to have a width of 400 pixels or the width of the original image (whichever is smaller). It ensures that the image is appropriately scaled for faster processing during object detection.

- image : The input image to be resized.
- width=min(400, image.shape[1]) : This argument specifies the new width for resizing. The height is calculated proportionally to maintain the aspect ratio.
- image.shape[1] : Original width of the image.

(regions, _) = hog.detectMultiScale(image, winStride=(4, 4), padding=(10, 10), scale=1.05)

The detectMultiScale() method detects objects (people) in the image by sliding a window over the image and analyzing the features using the HOG descriptor and the pre-trained SVM.

- image : The input image in which people are to be detected.
- winStride=(4, 4) : The step size for the sliding window (in both x and y directions). Larger values mean fewer windows and faster processing, but smaller values can improve detection accuracy.
- padding=(10, 10) : Padding added to each window during detection. This helps in preventing boundary issues when detecting objects near the edges of the image. Larger padding can allow for a more robust detection, but it may also slow down the process.
- scale=1.05 : The scaling factor that compensates for objects at different sizes (zooming in or out during detection). A value of 1.05 typically works well, though increasing the scale can speed up the detection (at the cost of precision).
- regions: A list of rectangles (x, y, w, h) where a person is detected. The coordinates (x, y) represent the top-left corner, and (w, h) represent the width and height of the bounding box.
- _: This is an unused variable for additional information returned by the method (e.g., the weights of the classifier), so it's discarded by using the underscore.

for (x, y, w, h) in regions:

cv2.rectangle(image, (x, y), (x + w, y + h), (0, 0, 255), 2)

Draws rectangles around the detected people in the image to visualize the detection result.

- image : The image on which the rectangle will be drawn.
- (x, y) : The top-left corner of the rectangle.
- (x + w, y + h) : The bottom-right corner of the rectangle (calculated from width w and height h).
- (0, 0, 255) : The color of the rectangle (in BGR format). (0, 0, 255) represents red.
- 2 : The thickness of the rectangle' s border.

Part A.4 -
Aim - Perform Face Recognition

Concept :
This practical implements face recognition using the face_recognition library, which employs deep learning-based feature extraction. It detects and compares faces between two images (nana.jpg and non.jpg) to determine if they belong to the same person. This process is widely used in security systems, identity verification, and biometric authentication.

1. **Loading and Preprocessing Images**
   The images are loaded using face_recognition.load_image_file(). Since OpenCV reads images in BGR format, they are converted to RGB using cv2.cvtColor(). This conversion is necessary because the face_recognition library expects images in RGB format for accurate face detection.
2. **Detecting Faces in Both Images**
   Face detection is performed using face_recognition.face_locations(), which returns the bounding box coordinates (top, right, bottom, left) of the detected face. These coordinates are used to draw a purple bounding box ((255, 0, 255)) around the detected face using cv2.rectangle().
3. **Encoding Facial Features for Recognition**
   Once a face is located, the function face_recognition.face_encodings() extracts unique facial features and represents them as a 128-dimensional feature vector.
   - encodeElon stores the encoded face data for nana.jpg (main image).
   - encodeTest stores the encoded face data for non.jpg (test image).

   These encodings allow the algorithm to numerically compare the two faces.
4. **Comparing Faces and Measuring Similarity**
   The function face_recognition.compare_faces() checks whether the two encodings match, returning True if they do, otherwise False.
   Additionally, face_recognition.face_distance() calculates the similarity score between the two faces.
   - Lower values indicate higher similarity (closer to 0 means a better match).
   - Higher values indicate lower similarity (greater than 0.6 usually means different faces).

   The results are printed and overlaid on the test image using cv2.putText(), displaying:
   - results (whether the faces match or not).
   - faceDis (the computed similarity score, rounded to 2 decimal places).
5. **Displaying the Images with Face Recognition Results**
   Both images, now annotated with bounding boxes and text, are displayed using cv2.imshow(). The program waits for user input (cv2.waitKey(0)) before closing the windows.

import cv2
import face_recognition
Imports the OpenCV and face recognition libraries for image processing and facial recognition.

- No arguments.
- No return value.

imgmain1 = face_recognition.load_image_file('nana.jpg') Loads the image 'nana.jpg' into a NumPy array.

- 'nana.jpg' : The filename of the image.

Returns a NumPy array representing the image.

imgmain = cv2.cvtColor(imgmain1, cv2.COLOR_BGR2RGB)
Converts the loaded image from BGR to RGB format for compatibility with face_recognition.

- imgmain1 : The original image in BGR format.
- cv2.COLOR_BGR2RGB : The color conversion flag.

Returns a NumPy array of the image in RGB format.

imgTest1 = face_recognition.load_image_file('non.jpg') Loads the test image 'non.jpg' into a NumPy array.

- 'non.jpg' : The filename of the test image.

Returns a NumPy array representing the test image.

imgTest = cv2.cvtColor(imgTest1, cv2.COLOR_BGR2RGB) Converts the test image from BGR to RGB format.

- imgTest1 : The test image in BGR format.
- cv2.COLOR_BGR2RGB : The color conversion flag.

Returns a NumPy array of the test image in RGB format.

faceLoc = face_recognition.face_locations(imgmain)[0]
Detects the face location in the main image and returns its bounding box.

- imgmain : The input image in RGB format.

Returns a tuple (top, right, bottom, left) representing the face's coordinates.

encodeElon = face_recognition.face_encodings(imgmain)[0]
Encodes the detected face into a 128-dimensional numerical representation.

- imgmain : The input image containing a face.

Returns a NumPy array representing the face encoding.

cv2.rectangle(imgmain, (faceLoc[3], faceLoc[0]), (faceLoc[1], faceLoc[2]), (255, 0, 255), 2)
Draws a rectangle around the detected face in the main image.

- imgmain : The image on which the rectangle will be drawn.
- (faceLoc[3], faceLoc[0]) : The top-left corner of the rectangle.
- (faceLoc[1], faceLoc[2]) : The bottom-right corner of the rectangle.
- (255, 0, 255) : The color of the rectangle (purple in BGR format).
- 2 : The thickness of the rectangle' s border.

faceLocTest = face_recognition.face_locations(imgTest)[0] Detects the face location in the test image.

- imgTest : The input test image in RGB format.

Returns a tuple (top, right, bottom, left) representing the face's coordinates.

encodeTest = face_recognition.face_encodings(imgTest)[0]
Encodes the detected face in the test image into a 128-dimensional numerical representation.

- imgTest : The test image containing a face.

Returns a NumPy array representing the face encoding.

cv2.rectangle(imgTest,(faceLocTest[3],faceLocTest[0]),(faceLocTest[1],faceLocTest[2]),(255, 0, 255), 2)
Draws a rectangle around the detected face in the test image.

- imgTest : The image on which the rectangle will be drawn.
- (faceLocTest[3], faceLocTest[0]) : The top-left corner of the rectangle.
- (faceLocTest[1], faceLocTest[2]) : The bottom-right corner of the rectangle.
- (255, 0, 255) : The color of the rectangle (purple in BGR format).
- 2 : The thickness of the rectangle' s border.

results = face_recognition.compare_faces([encodeElon], encodeTest)
Compares the face encodings of the main and test images to determine if they match.

- [encodeElon] : The encoding of the known face.
- encodeTest : The encoding of the test face.

Returns a list containing True if the faces match, otherwise False.

faceDis = face_recognition.face_distance([encodeElon], encodeTest)
Computes the Euclidean distance between the two face encodings. Lower values indicate greater similarity.

- [encodeElon] : The encoding of the known face.
- encodeTest : The encoding of the test face.

Returns a NumPy array containing the similarity score.

print(results, faceDis)
Prints the comparison result and face distance score to the console.

- results : The Boolean result of face comparison.
- faceDis : The distance between face encodings.

No return value.

cv2.putText(imgTest, f'{results} {round(faceDis[0],2)}', (20,20), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
Overlays the match result and similarity score on the test image.

- imgTest : The image to modify.
- f'{results} {round(faceDis[0], 2)}' : The text displaying the match result and similarity score.
- (20, 20) : The position of the text.
- cv2.FONT_HERSHEY_COMPLEX : The font type.
- 1 : The font scale.
- (0, 0, 255) : The color of the text (red in BGR format).
- 2 : The thickness of the text.

Part B :

Aim : Construct 3D model from images

Concept :

The practical implements a depth-based image shifting effect using PIL (Pillow) and NumPy. This effect is useful in 3D rendering, parallax effects, and image transformations where pixel positions are shifted based on depth information. The algorithm works by using a base image (cube1.jpeg) and a depth image (cube2.jpeg), where pixel positions in the base image are adjusted according to depth values.

1. **Loading Images and Preparing Data** The program loads the two images:
   - The base image (cube1.jpeg) is loaded and converted to RGBA format, ensuring an alpha channel (transparency support).
   - The depth image (cube2.jpeg) is loaded and converted to grayscale ("L" mode), ensuring single-channel pixel values.

   Both images are then converted to NumPy arrays for efficient pixel-wise operations.
2. **Computing Shift Amounts from Depth Data**
   The grayscale depth values range from 0 to 255, representing different depth levels. These values are normalized by dividing by 255, then multiplied by shift_amount (default = 10). The result is an integer array (deltas) that determines how much each pixel in the base image should be shifted horizontally.
   For example:
   - White pixels (255 in depth map) shift by 10 pixels.
   - Black pixels (0 in depth map) remain unchanged.
   - Gray pixels (between 0 and 255) shift by a fraction of 10 pixels, depending on their intensity.
3. **Shifting the Image Pixels**
   A new blank image (shifted_data) is created to store the transformed pixels. Thealgorithm iterates through each pixel in the depth map:
   It reads the shift amount (dx) from deltas.
   It moves the pixel at (x, y) to a new position (x + dx, y) in shifted_data, ensuring it stays within image boundaries.
   Pixels without assigned values remain transparent (black in the final output).
4. **Converting the Shifted Data Back to an Image**
   After all pixels are repositioned, the NumPy array is converted back into an image (Image.fromarray()), making it viewable. The final result is displayed using shifted_img.show().

Packages Used :

Pillow is a Python Imaging Library (PIL) fork and a powerful tool for image processing in Python. It is widely used for handling and manipulating images in various ways. Using Pillow in PyCharm combines the robust image-processing capabilities of Pillow with the productivity-enhancing features of PyCharm.

from PIL import Image # pip install Pillow import numpy as np
Imports necessary libraries for image manipulation and numerical operations.

- Image : Used to load, process, and manipulate images.
- numpy : Used for efficient numerical computations on image arrays. **def shift_image(img, depth_img, shift_amount=10):**

Defines a function to shift pixels in an image based on depth information.

- img : The base image to be shifted.
- depth_img : The depth map image, determining pixel shifts.
- shift_amount=10 : The maximum shift applied based on depth intensity.

Returns an image with shifted pixels.

img = img.convert("RGBA")
Ensures the base image has an alpha (transparency) channel.

- img : The input image.

Returns an image in "RGBA" mode (Red, Green, Blue, Alpha).

data = np.array(img)
Converts the image into a NumPy array for pixel-wise manipulation.

- img : The input image in RGBA format.

Returns a NumPy array representing pixel data.

depth_img = depth_img.convert("L")
Converts the depth image to grayscale to extract depth values.

- depth_img : The depth image.

Returns an image in "L" mode (grayscale).

depth_data = np.array(depth_img)
Converts the depth image into a NumPy array.

- depth_img : The grayscale depth image.

Returns a NumPy array where pixel intensity represents depth.

deltas = ((depth_data / 255.0) * float(shift_amount)).astype(int) Computes pixel shifts based on depth intensity.

- depth_data : The depth image as an array.
- 255.0 : Normalizes pixel intensity values between 0 and 1.
- shift_amount : The maximum shift applied.

Returns a NumPy array of integer shift values.

shifted_data = np.zeros_like(data)
Creates an empty image array filled with zeros (black, fully transparent).

- data : The original image array.

Returns an array of the same shape as data, initialized with zeros.

height, width, _ = data.shape Extracts image dimensions.

- data.shape : The shape of the original image array.
- height : The number of rows (image height).
- width : The number of columns (image width).

for y, row in enumerate(deltas): for x, dx in enumerate(row):
Iterates over every pixel in the image, using deltas to determine shifts.

- y : Row index (vertical position).
- x : Column index (horizontal position).
- dx : Computed shift value for the pixel at (x, y).

if x + dx<width and x + dx>= 0:
Ensures the shifted pixel remains within the image boundaries.

- x + dx<width : Ensures the new position does not exceed image width.
- x + dx>= 0 : Ensures the new position is not negative.

shifted_data[y, x + dx] = data[y, x]
Shifts the pixel from (x, y) to its new position (x + dx, y).

- data[y, x] : The original pixel value.
- shifted_data[y, x + dx] : The new position of the pixel in the output image.

shifted_image = Image.fromarray(shifted_data.astype(np.uint8)) Converts the shifted pixel data back into an image.

- shifted_data.astype(np.uint8) : Ensures pixel values are in the valid range (0-255).

Returns a PIL image object.

return shifted_image
Returns the final image with shifted pixels. **img = Image.open("cube1.jpeg")**

Loads the main image.

- "cube1.jpeg" : The filename of the input image.

Returns a PIL image object.

depth_img = Image.open("cube2.jpeg") Loads the depth map image.

- "cube2.jpeg" : The filename of the depth image.

Returns a PIL image object.

shifted_img = shift_image(img, depth_img, shift_amount=10) Applies the shift_image function to shift pixels based on depth.

- img : The base image.
- depth_img : The depth map image.
- shift_amount=10 : Maximum shift distance.

Returns a new image with shifted pixels.

shifted_img.show()

Displays the final shifted image.

- shifted_img : The modified image.

No return value.

Part C :

Aim : Implement Object Detection and Tracking from video.

Concept :

This practical implements **real-time face detection in a video file (v4.mp4)** using OpenCV' s **Haar cascade classifier**. The algorithm processes each video frame, detects faces, and draws bounding boxes around detected faces. This technique is widely used in **surveillance, automated monitoring, and facial recognition systems**.

1. **Loading the Pre-Trained Face Detection Model**
   The CascadeClassifier function loads the Haar cascade face detection model, a pre-trained XML file (haarcascade_frontalface_default.xml). This model is designed to recognize frontal faces based on pre-learned patterns of edges, textures, and intensity variations.
2. **Capturing Video Frames**
   Instead of using a webcam, the program reads frames from a video file (v4.mp4) using cv2.VideoCapture(). The frames are processed sequentially in a loop until the video ends.
3. **Detecting Faces in Each Frame**
   The function detect_bounding_box(vid) processes each frame by:
   - Converting it to grayscale (cv2.cvtColor()), as Haar cascades work more efficiently with grayscale images.
   - Applying face detection using detectMultiScale(), which scans the image at multiple scales and detects faces based on predefined features.
     - scaleFactor=1.1: Shrinks the image by 10% per iteration to detect faces at different sizes.
     - minNeighbors=5: Eliminates false positives by ensuring only strongly detected faces remain.
     - minSize=(40, 40): Ignores very small detections to avoid false positives.
   - Drawing a bounding box around detected faces using cv2.rectangle(), with a green (0, 255, 0) rectangle of thickness 4.
4. **Displaying the Processed Video**
   Each processed frame is displayed in a window using cv2.imshow(). The loop continues reading and processing frames until the user presses 'q', which triggers cv2.waitKey(1)&0xFF == ord("q"), breaking the loop.
5. **Releasing Resources**
   Once the video ends or the user exits, the system releases the video file (video_capture.release()) and closes all OpenCV windows (cv2.destroyAllWindows()) to free up memory.

Packages Used:

OpenCV is the most widely used library for computer vision tasks, including image processing, object detection, and object tracking. It supports both traditional computer vision methods and deep learning-based models.
NumPy is a library for numerical computing in Python. It is used heavily in computer vision tasks for manipulating arrays and matrices.
TensorFlow or **PyTorch** are used for advanced deep learning-based object detection models.
Pillow is a Python Imaging Library (PIL) fork that adds image processing capabilities. **Matplotlib** is often used for visualizing data and displaying images or videos with bounding boxes, labels, or other annotations.

face_classifier = cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_frontalface_default.xml ")
Loads the pre-trained Haar Cascade face detection model for detecting faces in images or videos.

- cv2.data.haarcascades + "haarcascade_frontalface_default.xml" : The path to the pre-trained face detection model.

Returns a CascadeClassifier object used for detecting faces.

video_capture = cv2.VideoCapture("v4.mp4") Opens the specified video file for processing.

- "v4.mp4" : The filename of the video.

Returns a VideoCapture object to read frames from the video.

def detect_bounding_box(vid):
Defines a function to detect faces and draw bounding boxes around them.

- vid : The input frame from the video.

Returns a list of detected face bounding boxes.

gray_image = cv2.cvtColor(vid, cv2.COLOR_BGR2GRAY)
Converts the video frame to grayscale for more efficient face detection.

- vid : The input frame in BGR format.
- cv2.COLOR_BGR2GRAY : Converts the frame to grayscale.

Returns a grayscale version of the frame.

faces = face_classifier.detectMultiScale(gray_image, 1.1, 5, minSize=(40, 40)) Detects faces in the grayscale frame using the Haar Cascade classifier.

- gray_image : The grayscale frame.
- 1.1 : Scale factor (reduces image size at each scale to detect faces at different sizes).

- 5 : The number of neighbors each candidate rectangle must have to be considered a face.
- minSize=(40, 40) : Minimum size of detected faces.

Returns a list of detected face bounding boxes as (x, y, w, h).

for (x, y, w, h) in faces:
Iterates through each detected face and extracts its bounding box coordinates.

- (x, y) : The top-left corner of the face.
- (w, h) : The width and height of the face.

cv2.rectangle(vid, (x, y), (x + w, y + h), (0, 255, 0), 4) Draws a rectangle around the detected face.

- vid : The video frame where the rectangle is drawn.
- (x, y) : The top-left corner of the rectangle.
- (x + w, y + h) : The bottom-right corner of the rectangle.
- (0, 255, 0) : The color of the rectangle (green in BGR format).
- 4 : The thickness of the rectangle' s border.

return faces
Returns the list of detected faces.

while True:
Starts an infinite loop to process video frames.

result, video_frame = video_capture.read() Reads the next frame from the video.

- video_capture.read() : Captures a frame from the video file.
- result : Boolean value (True if the frame is read successfully, False otherwise).
- video_frame : The captured frame.

if result is False:
Checks if the frame was not read successfully.

break
Exits the loop if the video ends or there is an error in reading the frame.

detect_bounding_box(video_frame)
Calls the face detection function to process the current frame.

- video_frame : The current video frame.

Modifies the frame by drawing bounding boxes around detected faces.

cv2.imshow("Face Detection in Video", video_frame) Displays the processed video frame with detected faces.

- "Face Detection in Video" : The name of the window.
- video_frame : The processed frame with bounding boxes.

if cv2.waitKey(1)&0xFF == ord("q"):
Waits for a key press and checks if the 'q' key is pressed to exit the loop.

- 1 : Waits 1 millisecond before capturing the next frame.
- ord("q") : Checks if the pressed key is 'q'.

break
Exits the loop when 'q' is pressed.

video_capture.release()
Releases the video capture object and frees resources.

cv2.destroyAllWindows()
Closes all OpenCV display windows.

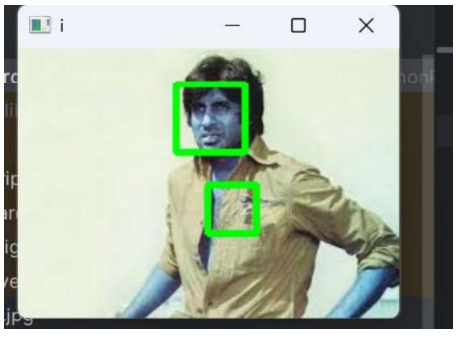
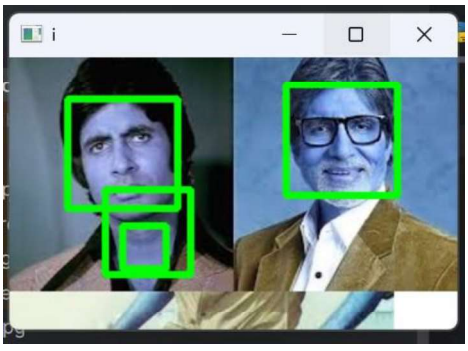
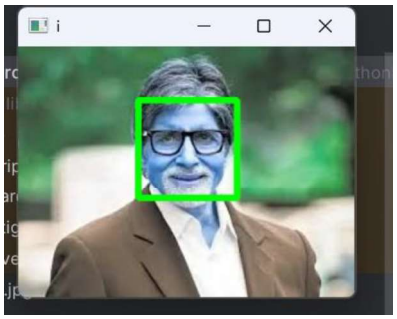
Practical No. 2

Part A.1 -
Aim - Perform Face Detection Packaged Used - CV2, Matplotlib
Files Used - input1.jpg, input2.jpg, input3.jpg **Command -** Terminal

- pip install opencv-python (cv2)
- Pip install matplotlib (Matplotlib)

Code - import cv2
import matplotlib.pyplot as plt img = cv2.imread('input4.jpeg')
gray_image = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
face_classifier = cv2.CascadeClassifier(
cv2.data.haarcascades + "haarcascade_frontalface_default.xml") face = face_classifier.detectMultiScale(
gray_image, scaleFactor=1.01, minNeighbors=2, minSize=(5, 5))
for (x, y, w, h) in face:
cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 4) img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) cv2.imshow('i',img_rgb)
#plt.axis('off')
cv2.waitKey(0)

Output -

Computer Vision Practical

Part A.2 -
Aim - Perform Object Detection

Image Used - stop.png, stop1.jpeg, stop2.jpeg **Files used -** stop_data.xml
Steps -

- **Open pycharm**
- **Create Project / Open Project**
- **Upload the images on the project**
- **Upload stop_data.xml in the project**
- **Start Execution**

Code -

```
import cv2
from matplotlib import pyplot as plt
# Opening image
img = cv2.imread("stop.png")
# OpenCV opens images as BRG so it must be converted to RGB and grayscale version
img_gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB) # Use minSize
because for not
# bothering with extra-small
# dots that would look like STOP signs
stop_data = cv2.CascadeClassifier('stop_data.xml')
found = stop_data.detectMultiScale(img_gray, minSize =(20, 20)) # Don't do anything if
there's
# no sign
amount_found = len(found) if
amount_found != 0:
        # There may be more than one
        # sign in the image
        for (x, y, width, height) in found:

                # We draw a green rectangle around # every
                recognized sign
```

```
                cv2.rectangle(img_rgb, (x, y),
                              (x + height, y + width), (0, 255,
                              0), 5)
# Creates the environment of # the
picture and shows it
cv2.imshow('img',img_rgb) plt.show()
cv2.waitKey(0)
```

Output-

Part A.3 -
Aim - Perform Pedestrian detection

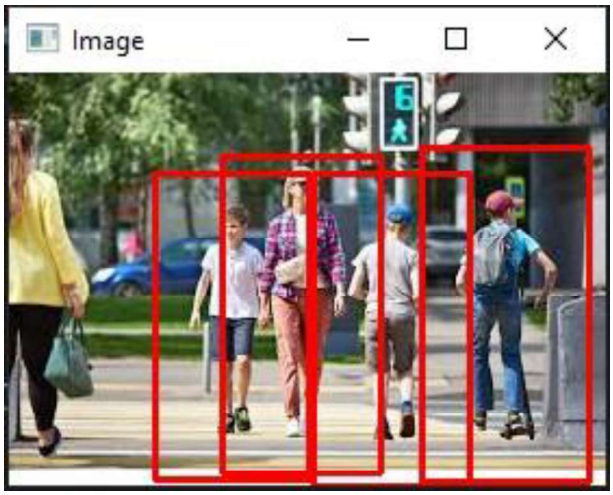
Packages Used – imutils
Command - Terminal

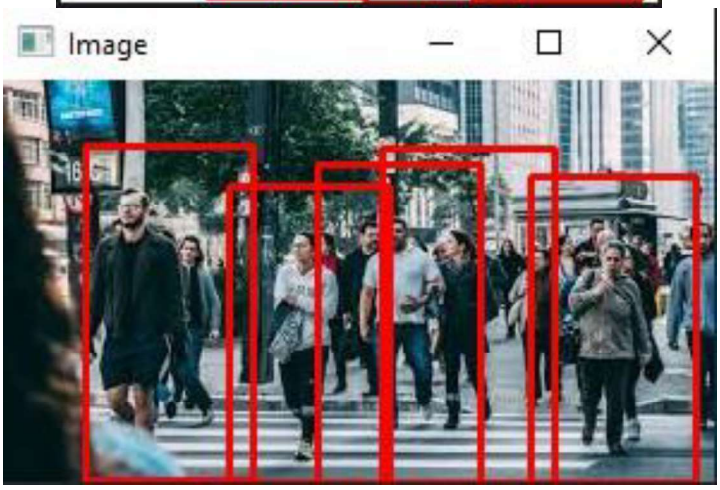
- pip install imutils

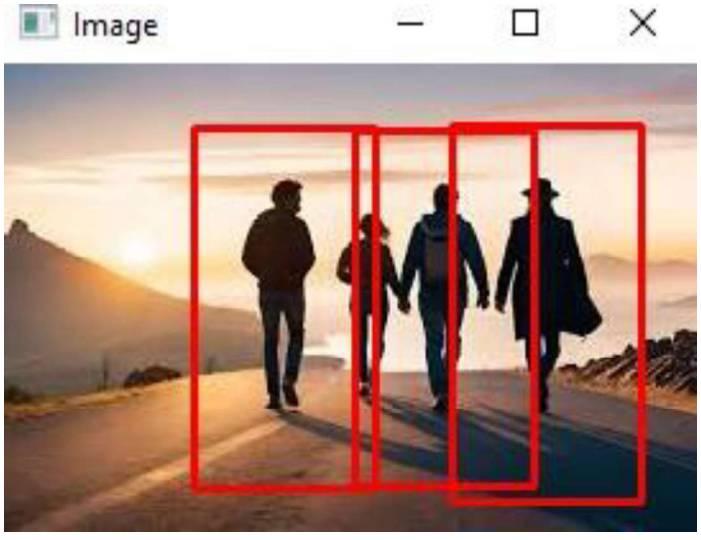
Image Used - input1.png, people1.jpeg, people2.jpeg **Code -**

```
import cv2 import
imutils
# Initializing the HOG person
# detector
hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector()) # Reading the Image
image = cv2.imread('people2.jpeg')
# Resizing the Image
image = imutils.resize(image,width=min(400, image.shape[1]))

# Detecting all the regions in the
# Image that has a pedestrians inside it (regions, _) =
hog.detectMultiScale(image,
        winStride=(4, 4),
        padding=(10, 10), scale=1.05)
print(regions)
# Drawing the regions in the Image for (x, y, w, h)
in regions:
        cv2.rectangle(image, (x, y),
        (x + w, y + h), (0,
        0, 255), 2)
# Showing the output Image cv2.imshow("Image",
image) cv2.waitKey(0) cv2.destroyAllWindows()
```

Output -



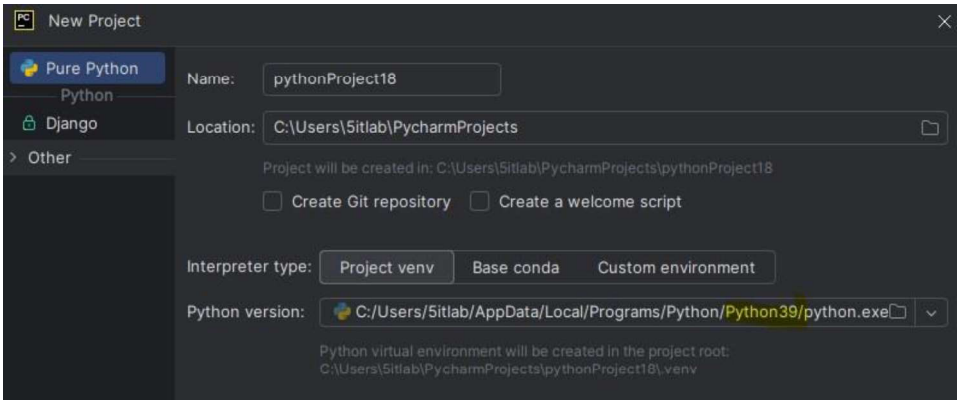




Part A.4 -
Aim - Perform Face Recognition

Packages Used - cv2, cmake, face_recognition **Instruction**:

1) Install Python 3.9>>Set Python path on python 3.9



2) Copy **dlib-19.23.0-cp39-cp39-win_amd64.whl** file in 3 locations

- Pycharm Project
- C:\Users\5itlab\PycharmProjects\pythonProject17\.venv\Scripts
- C:\Users\5itlab\AppData\Local\Programs\Python\Python39\Scripts

3) Install all the packages in Pycharm Projects pip install
cmake
pip install dlib-19.23.0-cp39-cp39-win_amd64.whl
pip install opencv-python (version 3.9) pip install face_recognition
pip uninstall pillow pip install pillow pip uninstall numpy
pip install numpy==1.26.4

4) Install the packages in (PYCHARM LINK)
C:\Users\5itlab\PycharmProjects\pythonProject17\.venv\Scripts pip install cmake
pip install dlib-19.23.0-cp39-cp39-win_amd64.whl pip install opencv-python (version 3.9)
pip install face_recognition

5) Install the packages in (PROGRAM LINK)
C:\Users\5itlab\AppData\Local\Programs\Python\Python39\Scripts pip install cmake
pip install dlib-19.23.0-cp39-cp39-win_amd64.whl pip install opencv-python (version 3.9)

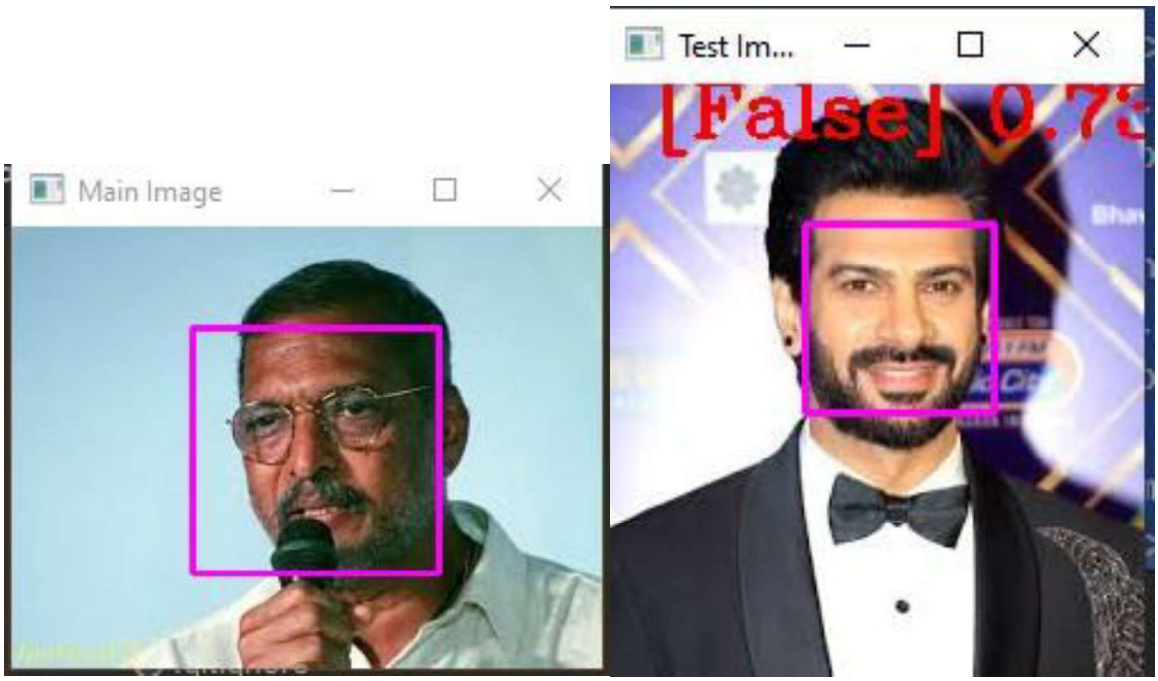
pip install face_recognition

Files Used: nana.jpg, non.jgp, dlib-19.23.0-cp39-cp39-win_amd64.whl

Code:

```
import cv2
import face_recognition
imgmain1 = face_recognition.load_image_file('nana.jpg') imgmain = cv2.cvtColor(imgmain1, cv2.COLOR_BGR2RGB) imgTest1 = face_recognition.load_image_file('non.jpg') imgTest = cv2.cvtColor(imgTest1, cv2.COLOR_BGR2RGB)

faceLoc = face_recognition.face_locations(imgmain)[0] encodeElon = face_recognition.face_encodings(imgmain)[0]
cv2.rectangle(imgmain, (faceLoc[3], faceLoc[0]), (faceLoc[1], faceLoc[2]), (255, 0, 255), 2)

faceLocTest = face_recognition.face_locations(imgTest)[0] encodeTest = face_recognition.face_encodings(imgTest)[0]
cv2.rectangle(imgTest, (faceLocTest[3], faceLocTest[0]), (faceLocTest[1], faceLocTest[2]), (255, 0, 255), 2)
results = face_recognition.compare_faces([encodeElon], encodeTest) faceDis = face_recognition.face_distance([encodeElon], encodeTest) print(results, faceDis)
cv2.putText(imgTest, f'{results} {round(faceDis[0], 2)}', (20, 20), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
cv2.imshow('Main Image', imgmain)
cv2.imshow('Test Image', imgTest) cv2.waitKey(0)
```

Output:


Main Image
Test Im...
[False] 0.73

Case 2:

Code:

```
import cv2
import face_recognition
imgmain1 = face_recognition.load_image_file('jackiedada.jpg') imgmain =
cv2.cvtColor(imgmain1, cv2.COLOR_BGR2RGB) imgTest1 =
face_recognition.load_image_file('nanap.jpg') imgTest = cv2.cvtColor(imgTest1,
cv2.COLOR_BGR2RGB) faceLoc = face_recognition.face_locations(imgmain)[0]
encodeElon = face_recognition.face_encodings(imgmain)[0]
cv2.rectangle(imgmain, (faceLoc[3], faceLoc[0]), (faceLoc[1], faceLoc[2]), (255, 0, 255), 2)
faceLocTest = face_recognition.face_locations(imgTest)[0] encodeTest =
face_recognition.face_encodings(imgTest)[0]
cv2.rectangle(imgTest, (faceLocTest[3], faceLocTest[0]), (faceLocTest[1], faceLocTest[2]), (255, 0, 255), 2)
results = face_recognition.compare_faces([encodeElon], encodeTest) faceDis =
face_recognition.face_distance([encodeElon], encodeTest) print(results, faceDis)
cv2.putText(imgTest, f'{results} {round(faceDis[0], 2)}', (20, 20),
cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
cv2.imshow('Main Image', imgmain)
cv2.imshow('Test Image', imgTest) cv2.waitKey(0)
```

Output:

Part B -
Aim - Perform Construct 3D model from images

Packages used - Pillow Command -

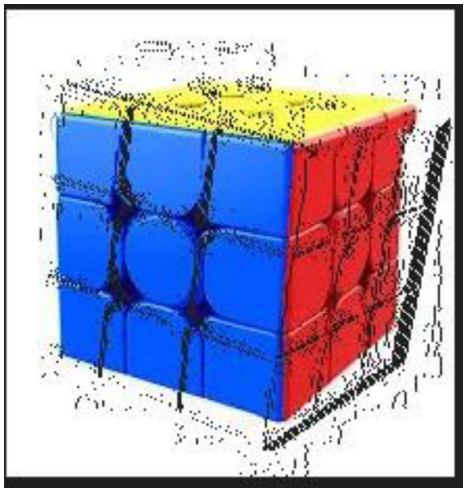
- C:\Users\5itlab\PycharmProjects\CVP\venu\Scripts\activate (To Set the given script path in the terminal.
- Pip install pillow (Pillow)

Image Used - cube1.jpeg, cube2.jpeg **Code -**

```
from PIL import Image # pip install Pillow import
numpy as np
def shift_image(img, depth_img, shift_amount=10):
    # Ensure base image has alpha img =
    img.convert("RGBA") data =
    np.array(img)
    # Ensure depth image is grayscale (for single value) depth_img =
    depth_img.convert("L")
```

```
depth_data = np.array(depth_img)
deltas = ((depth_data / 255.0) * float(shift_amount)).astype(int) # This creates the transparent resulting image.
# For now, we're dealing with pixel data. shifted_data = np.zeros_like(data) height, width, _ = data.shape

for y, row in enumerate(deltas): for x, dx in enumerate(row):
    if x + dx < width and x + dx >= 0: shifted_data[y, x + dx] = data[y, x]
# Convert the pixel data to an image.
shifted_image = Image.fromarray(shifted_data.astype(np.uint8)) return shifted_image
img = Image.open("cube1.jpeg") depth_img = Image.open("cube2.jpeg")
shifted_img = shift_image(img, depth_img, shift_amount=10) shifted_img.show()
```

Output –



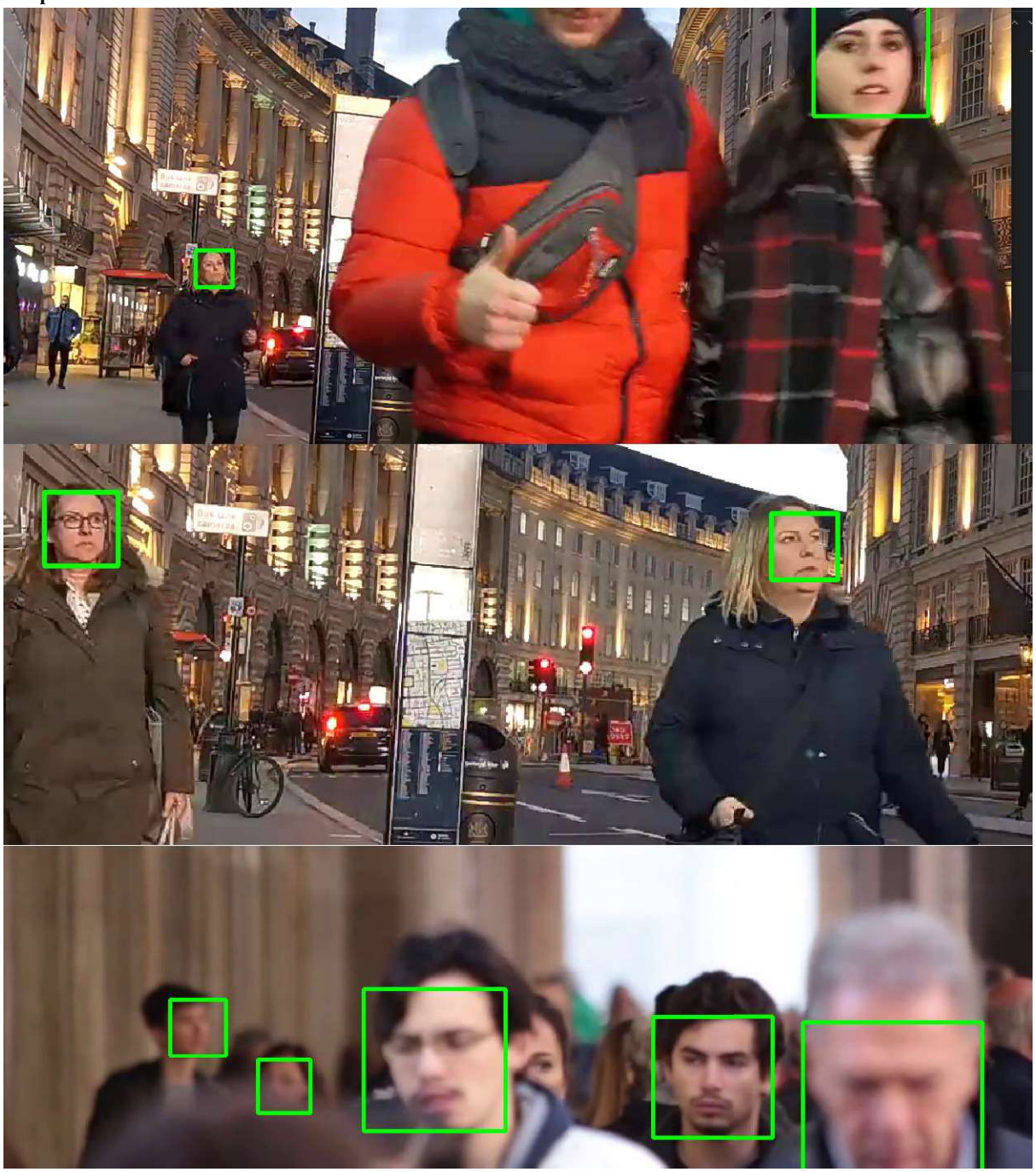
Part C

Aim - Implement object detection and tracking from video **Files used -** v2.mp4, v3.mp4

Code -

```
import cv2
# Load the pre-trained face detection model face_classifier =
cv2.CascadeClassifier(
    cv2.data.haarcascades + "haarcascade_frontalface_default.xml"
)
# Replace webcam capture with video file capture video_capture =
cv2.VideoCapture("v4.mp4")
# Function to detect faces and draw bounding boxes
def detect_bounding_box(vid):
    gray_image = cv2.cvtColor(vid, cv2.COLOR_BGR2GRAY)  # Convert to grayscale
    faces = face_classifier.detectMultiScale(gray_image, 1.1, 5, minSize=(40, 40))
    for (x, y, w, h) in faces:
        cv2.rectangle(vid, (x, y), (x + w, y + h), (0, 255, 0), 4)  # Draw rectangle return faces
# Process video frames while True:
    result, video_frame = video_capture.read() # Read frames from the video file if result is False: #
    terminate the loop if the frame is not read successfully
        break
 detect_bounding_box(video_frame) # Detect faces and draw bounding boxes cv2.imshow(
        "Face Detection in Video", video_frame
    ) # Display the processed frame in a window
   if cv2.waitKey(1) & 0xFF == ord("q"): # Exit when 'q' is pressed break
# Release video capture and close display window video_capture.release()
cv2.destroyAllWindows()
```

Output -



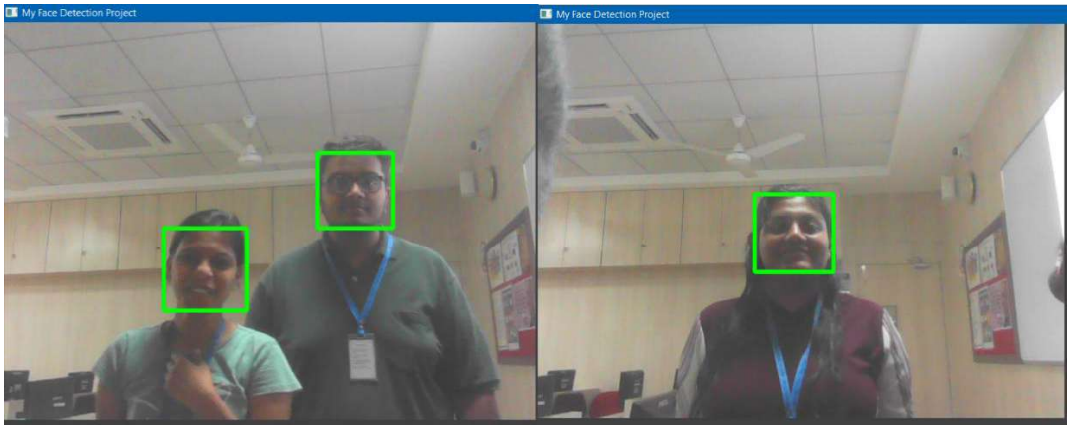
Case 2 : Implement object detection and tracking from Camera Input.

Files Used: haarcascade_frontalface_default.xml - location **(C:\Users\5itlab\PycharmProjects\pythonProject17\.venv\Lib\site-packages\cv 2\data)**

Code:

```
import cv2
face_classifier =
cv2.CascadeClassifier(cv2.data.haarcascades+"haarcascade_frontalface_defa
ult.xml")
video_capture = cv2.VideoCapture(0)

def detect_bounding_box(vid):
    gray_image = cv2.cvtColor(vid, cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray_image, 1.1, 5, minSize=(40, 40)) for (x, y, w, h) in faces:
        cv2.rectangle(vid, (x, y), (x + w, y + h), (0, 255, 0), 4) return faces

while True:
    result, video_frame = video_capture.read() # read frames from the video if result is False:
        break # terminate the loop if the frame is not read successfully faces =
    detect_bounding_box(
        video_frame
    ) # apply the function we created to the video frame cv2.imshow(
        "My Face Detection Project", video_frame
    ) # display the processed frame in a window named "My Face Detection Project"
    if cv2.waitKey(1) & 0xFF == ord("q"):
        break

video_capture.release() cv2.destroyAllWindows()
```

Output:


My Face Detection Project
My Face Detection Project

Practical No. 3

Part A :

Aim : Perform Feature Extraction using RANSAC

Concept :

1. Linear Least Squares Regression (linearleastsquare.py)

The **Linear Least Squares model** (LinearLeastSquareModel) is used for polynomial curve fitting. It solves the equation:

P=(ATA)− 1ATYP = (A^T A)^{-1} A^T YP=(ATA)− 1ATY

where:

- AAA represents the **design matrix** containing polynomial features of input xxx.
- YYY is the **output vector**.
- PPP is the **estimated model parameters** (coefficients of the polynomial).

This method **minimizes the sum of squared residuals** but is **sensitive to outliers**, which is why **RANSAC is used as an alternative**.

2. RANSAC Algorithm (ransac.py)

RANSAC (**Random Sample Consensus**) is an iterative method to **estimate a model from data with outliers**. The steps are:

1. **Randomly select a subset (num_sample=3)** of data points.
2. **Fit a model using the subset** (via the Least Squares method).
3. **Evaluate the model** using all data points by counting **inliers** (points within threshold error).
4. **Update the best model** if the new model has more inliers than the previous best.
5. **Estimate the probability of outliers** to **adaptively determine the number of iterations** needed.
6. **Repeat until the stopping condition is met** (desired probability of success). This ensures that **outliers (noise) do not heavily impact the final model**, unlike the least squares approach.

3. Model Fitting and Visualization (modelfitting.py)

- Reads datasets (data_1.csv and data_2.csv).
- Computes **second-degree polynomial features** (x^2, x, 1) for curve fitting.
- Uses both **Least Squares and RANSAC** to fit curves.
- **Plots and compares results**:
  - **Red curve**: Least Squares model.
  - **Blue curve**: RANSAC model (robust to outliers).

Packages Used :

scikit-learn is often used for robust regression tasks, including feature extraction.

OpenCV provides robust feature matching techniques, including the use of RANSAC in methods like findHomography() and estimateFundamentalMat() for applications in image matching, homography, and stereo vision.
PyTorch and **TensorFlow** are used to build custom models or layers that involve robust fitting techniques (e.g., combining RANSAC with neural networks for specific applications).
Numpy and **SciPy** are often used in conjunction with other packages like scikit-learn and opencv for basic matrix operations, data manipulation, and statistical computations when implementing RANSAC or working with feature extraction tasks.

ransac = RANSACRegressor()
This initializes the RANSACRegressor object.

- min_samples : The minimum number of data points required to estimate the model (default is 2, i.e., two data points for linear regression).
- residual_threshold : The maximum distance from a point to the model to be considered an inlier (default is None, which automatically sets a value based on the scale of the data).
- is_data_valid : Boolean value that determines if the data is valid (default is True).
- max_trials : The maximum number of iterations to run the algorithm (default is 100).
- random_state : Controls the randomization of the algorithm. This helps with reproducibility. (Default is None).

ransac.fit(X, y)
This fits the RANSAC model to the provided data.

- X : The input data, which should be a 2D array of shape (n_samples, n_features) where n_samples is the number of data points, and n_features is the number of features (in this case, it's just one feature).
- y : The target data (labels), which is the dependent variable, in this case, the values you are trying to predict.

RANSAC Algorithm (ransac.py)
class RansacModel: defines a class for implementing the RANSAC algorithm for curve fitting.
def __init__(self, curve_fitting_model): initializes the RANSAC model with a given curve fitting model.

- curve_fitting_model : An instance of a curve fitting model (e.g., least squares).
- No return value.

def fit(self, A, Y, num_sample, threshold): implements the RANSAC algorithm to estimate the best curve fitting model.

- A : Input matrix of features.
- Y : Output values corresponding to A.
- num_sample : Number of samples for model fitting (fixed at 3).
- threshold : Error threshold to determine inliers.
- Returns the best estimated model.

num_iterations = math.inf sets the number of iterations to infinity initially. **iterations_done = 0** keeps track of completed iterations. **max_inlier_count = 0** stores the highest number of inliers found. **best_model = None** holds the best model found so far.

prob_outlier = 0.5 initializes the estimated probability of an outlier. **desired_prob = 0.95** sets the desired probability of finding the best model. **total_data = np.column_stack((A, Y))** combines A and Y into a single dataset for shuffling.

data_size = len(total_data) gets the number of data points.

while num_iterations>iterations_done: runs the RANSAC loop until the required number of iterations is reached.

np.random.shuffle(total_data) shuffles the dataset to pick random samples. **sample_data = total_data[:num_sample, :]** selects num_sample random data points. **estimated_model = self.curve_fitting_model.fit(sample_data[:, :-1], sample_data[:, -1:])** fits the curve using the selected sample data.

- sample_data[:, :-1] : Extracts feature columns.
- sample_data[:, -1:] : Extracts output column.
- Returns the estimated model parameters.

y_cap = A.dot(estimated_model): calculates predicted values using the estimated model.

err = np.abs(Y - y_cap.T): computes absolute errors between actual and predicted values.

inlier_count = np.count_nonzero(err<threshold): counts inliers (data points with errors below threshold).

if inlier_count>max_inlier_count: checks if the current model is better than previous models.

- Updates max_inlier_count and best_model if more inliers are found. **prob_outlier = 1 - inlier_count / data_size**: updates the estimated probability of an outlier.

num_iterations = math.log(1 - desired_prob) / math.log(1 - (1 - prob_outlier) ** num_sample): updates the required number of iterations.

iterations_done = iterations_done + 1: increments the iteration count.

return best_model: returns the best estimated model after the loop completes. **Linear Least Squares Regression (linearleastsquare.py)**

class LinearLeastSquareModel: defines a class for solving least squares problems. **def fit(self, A, Y):** fits a linear model to the given data using least squares.

- A : Feature matrix.
- Y : Output values.
- Returns the estimated model parameters.

A_transpose = A.transpose(): computes the transpose of A. ATA = A_transpose.dot(A) computes A' * A.
ATY = A_transpose.dot(Y) computes A' * Y.
model = (np.linalg.inv(ATA)).dot(ATY) solves for model parameters using (A'A)^(-1) * A'Y.
return model returns the estimated parameters. **Model Fitting and Visualization (modelfitting.py):**
import numpy as np, import math, import pandas as pd, and from matplotlib import pyplot as plt: import necessary libraries.
from ransac import RansacModel and from linearleastsquare import LinearLeastSquareModel: import the model classes.
def fit_curve(data): fits a curve to given data using least squares and RANSAC.

- data : Pandas DataFrame containing x and y values.
- Returns fitted curves from least squares and RANSAC. **x_values = np.array(data['x']):** extracts x-values from the dataset. **y_values = np.array(data['y']):** extracts y-values from the dataset.

x_sq = np.power(x_values, 2): computes squared x-values for quadratic fitting.
A = np.stack((x_sq, x_values, np.ones((len(x_values)), dtype=int)), axis=1): constructs the feature matrix [x^2 x 1].
threshold = np.std(y_values) / 2: sets an adaptive inlier threshold based on standard deviation.
linear_ls_model = LinearLeastSquareModel(): creates an instance of the least squares model.
linear_ls_model_estimate = linear_ls_model.fit(A, y_values): fits the least squares model.
linear_model_y = A.dot(linear_ls_model_estimate): computes predicted values from the least squares model.
ransac_model = RansacModel(linear_ls_model): creates an instance of the RANSAC model using the least squares model.
ransac_model_estimate = ransac_model.fit(A, y_values, 3, threshold): fits the RANSAC model.
ransac_model_y = A.dot(ransac_model_estimate): computes predicted values from the RANSAC model.
return linear_model_y, ransac_model_y: returns both fitted curves.
if __name__ == '__main__': ensures the script runs only when executed directly.
df1 = pd.read_csv('data_1.csv') and df2 = pd.read_csv('data_2.csv') load two datasets.

ls_model_y1, ransac_model_y1 = fit_curve(df1) fits curves for dataset 1. **ls_model_y2, ransac_model_y2 = fit_curve(df2)** fits curves for dataset 2. **fig, (ax1, ax2) = plt.subplots(1, 2)** creates a figure with two subplots. **ax1.set_title('Dataset-1')** sets the title for the first subplot.
ax1.scatter(df1['x'], df1['y'], marker='o', color=(0, 1, 0), label='data points') plots original data points.
ax1.plot(df1['x'], ls_model_y1, color='red', label='Least square model') plots the least squares model.
ax1.plot(df1['x'], ransac_model_y1, color='blue', label='Ransac model') plots the RANSAC model.
ax1.set(xlabel='x-axis', ylabel='y-axis') labels axes.
ax1.legend() adds a legend.
Same is done for another data set. plt.show() displays the plots.

Part B :
Aim : Perform Colorization

Concept :
Colorization is the process of adding colors to grayscale (black-and-white) images or videos. This is done by predicting the missing chrominance (color) information while preserving the luminance (brightness details).
Traditionally, colorization was done manually by artists, but deep learning models have now automated the process with realistic results.

Most deep learning-based colorization techniques use the LAB color space instead of RGB:

- L (Lightness) → Represents brightness (grayscale image).
- A (Green-Red channel) → Represents color from green to red.
- B (Blue-Yellow channel) → Represents color from blue to yellow.

By using deep learning models, the L channel is kept as input, while the model predicts the missing A and B channels to generate a realistic color image.

ECCV 2016 Model (Deep Colorization Model):
The **ECCV 2016 Colorization Model** was introduced in a research paper titled *"Colorful Image Colorization"* by Richard Zhang, Phillip Isola, and Alexei A. Efros at the **European Conference on Computer Vision (ECCV 2016)**.

- Uses a **deep convolutional neural network (CNN)** to learn how to predict **AB color channels** from a grayscale image.
- The model is trained on **millions of images** from ImageNet, learning **natural color distributions**.
- Introduces a **probabilistic color distribution approach** to handle ambiguity in colors.
- **Working:**
  - The input grayscale image is **fed into a deep CNN**.
  - The CNN **predicts the missing AB color channels**.
  - The output LAB image is converted **back to RGB** for display.

Limitations

- The model may **struggle with unusual color distributions**.
- Sometimes, it **produces desaturated or unrealistic colors**.

SIGGRAPH 2017 Model (Enhanced Colorization Model):
The **SIGGRAPH 2017 Model** was introduced in the paper *"Learning Representations for Automatic Colorization"* by Richard Zhang et al. at **SIGGRAPH 2017**, a premier graphics conference.

- Uses a **deeper CNN architecture** to improve **sharpness and color accuracy**.

- Introduces **semantic understanding**, meaning it learns **context** to apply the correct colors (e.g., sky is blue, leaves are green).
- Can **self-correct unrealistic color predictions**.
- Uses **a classifier to refine ambiguous colors**, ensuring more **realistic and vibrant results**.
- **Working**
  - The grayscale image is passed through a **deeper CNN model**.
  - The model **analyzes object-level features** (e.g., detects faces, landscapes, etc.).
  - The **AB color channels are predicted with higher accuracy**.
  - The output is **converted to RGB** and displayed.

Advantages Over ECCV 2016

- **More realistic and natural colors**
- **Better context-awareness** (understands what objects are)
- **Sharper and more detailed output**

Packages Used :

Torch is an open-source machine learning library originally developed for Lua, which has now evolved into PyTorch, a widely used Python library. PyTorch, built on Torch's foundation, is a flexible, high-performance framework used for building and training deep learning models. Torch and PyTorch are designed to handle large-scale computational tasks efficiently, particularly in the domain of deep learning and neural networks.

argparse handles command-line arguments (though unused in this script). **matplotlib.pyplot** is used for displaying and saving images.

colorizer_eccv16 = eccv16(pretrained=True).eval() colorizer_siggraph17 = siggraph17(pretrained=True).eval() Loads two deep learning models for image colorization.

- ECCV 16 : Model from the 2016 European Conference on Computer Vision.
- SIGGRAPH 17 : Model from the 2017 SIGGRAPH Conference.

img = load_img('imgs/ansel_adams2.jpg')
(tens_l_orig, tens_l_rs) = preprocess_img(img, HW=(256,256))

Loads the black-and-white image. Converts the image to Lab color space. Extracts the L-channel (grayscale information) at two resolutions:

- tens_l_orig : Original size.
- tens_l_rs : Resized to 256x256 (needed for the models).

img_bw = postprocess_tens(tens_l_orig, torch.cat((0*tens_l_orig,0*tens_l_orig),dim=1))

out_img_eccv16 = postprocess_tens(tens_l_orig, colorizer_eccv16(tens_l_rs).cpu()) out_img_siggraph17 = postprocess_tens(tens_l_orig,colorizer_siggraph17(tens_l_rs).cpu())

- torch.cat((0*tens_l_orig, 0*tens_l_orig), dim=1): Creates a zero-filled a*b color channels (L stays the same).
- postprocess_tens(): Converts it back to an RGB image.
- colorizer_eccv16(tens_l_rs).cpu(): Predicts a*b color channels.
- colorizer_siggraph17(tens_l_rs).cpu(): Same process using a different model.
- postprocess_tens(): Converts the outputs into RGB format.

Practical No. 3

Part A

Aim: Perform Feature extraction using RANSAC

Steps:

Packages Used: numpy, math, pandas, matplotlib

Command:

pip install numpy pip install pandas pip install matplotlib

Files Used:

data_1.csv, data_2.csv

ransac.py

```
import numpy as np import math

class RansacModel:
    def __init__(self, curve_fitting_model):
        self.curve_fitting_model = curve_fitting_model

    def fit(self, A, Y, num_sample, threshold):
        num_iterations = math.inf iterations_done = 0
        num_sample = 3

        max_inlier_count = 0
        best_model = None

        prob_outlier = 0.5
        desired_prob = 0.95

        total_data = np.column_stack((A, Y)) # [A | Y] data_size =
        len(total_data)

        # Adaptively determining the number of iterations while
        num_iterations > iterations_done:

            # Shuffle the rows and take the first 'num_sample' rows as sample data np.random.shuffle(total_data)
```

```
        sample_data = total_data[:num_sample, :]

        estimated_model = self.curve_fitting_model.fit(sample_data[:, :-1], sample_data[:, -1:]) # [a b c]

        # Count the inliers within the threshold y_cap =
        A.dot(estimated_model)
        err = np.abs(Y - y_cap.T)
        inlier_count = np.count_nonzero(err<threshold)

        # check for the best model
        if inlier_count > max_inlier_count:
           max_inlier_count = inlier_count
           best_model = estimated_model

        prob_outlier = 1 - inlier_count / data_size print('#
        inliers:', inlier_count)
        print('# prob_outlier:', prob_outlier)
        num_iterations = math.log(1 - desired_prob) / math.log(1 - (1 - prob_outlier) ** num_sample)
        iterations_done = iterations_done + 1

        print('# s:', iterations_done) print('#
        n:', num_iterations)
        print('# max_inliner_count: ', max_inlier_count) return
    best_model
```

linearleastsquare.py import numpy as np

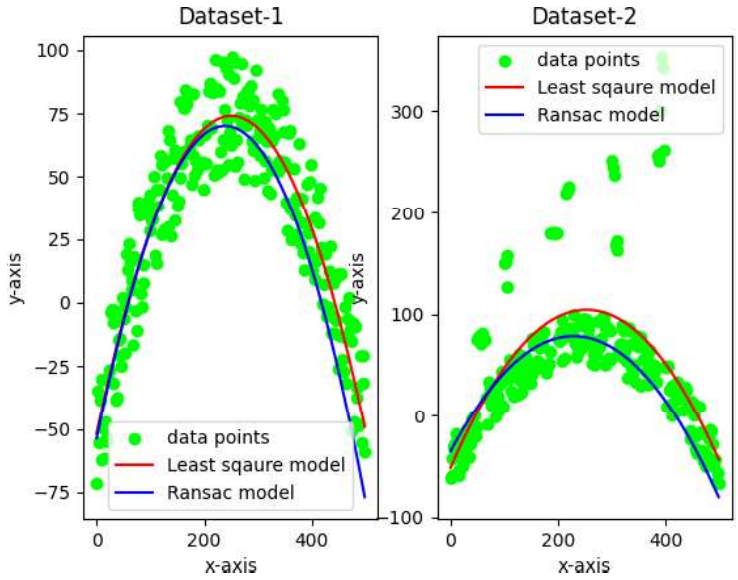
```
class LinearLeastSquareModel: def
  fit(self, A, Y):
     A_transpose = A.transpose() ATA =
     A_transpose.dot(A) ATY =
     A_transpose.dot(Y)
     model = (np.linalg.inv(ATA)).dot(ATY)
     # For a linear eq. AP = Y to solve a least sqaure problem, P = (inverse(A'A))(A'Y) return model
```

modelfitting.py import numpy as np import math

```
import pandas as pd
from matplotlib import pyplot as plt from
ransac import RansacModel
from linearleastsquare import LinearLeastSquareModel #
from<filename>import<classname>


def fit_curve(data):
    x_values = np.array(data['x']) y_values =
    np.array(data['y'])
    x_sq = np.power(x_values, 2) ## A =
    [x^2 x 1]
    A = np.stack((x_sq, x_values, np.ones((len(x_values)), dtype=int)), axis=1)
    threshold = np.std(
        y_values) / 2  # this can be tuned to sd/3 or sd/5 for various curves and better consistent results as a
result of random sampling

    # Instantiating the linear least square model linear_ls_model =
    LinearLeastSquareModel() linear_ls_model_estimate =
    linear_ls_model.fit(A, y_values) linear_model_y =
    A.dot(linear_ls_model_estimate)

    # Instantiating the ransac model
    ransac_model = RansacModel(linear_ls_model) ransac_model_estimate =
    ransac_model.fit(A, y_values, 3, threshold) ransac_model_y =
    A.dot(ransac_model_estimate)
    return linear_model_y, ransac_model_y if
__name__ =='__main__':
    # reading the values
    df1 = pd.read_csv('data_1.csv') df2 =
    pd.read_csv('data_2.csv')

    ls_model_y1, ransac_model_y1 = fit_curve(df1) ls_model_y2,
    ransac_model_y2 = fit_curve(df2)
    fig, (ax1, ax2) = plt.subplots(1, 2)
    ax1.set_title('Dataset-1')
    ax1.scatter(df1['x'], df1['y'], marker='o', color=(0, 1, 0), label='data points')
```

```
ax1.plot(df1['x'], ls_model_y1, color='red', label='Least sqaure model') ax1.plot(df1['x'],
ransac_model_y1, color='blue', label='Ransac model') ax1.set(xlabel='x-axis', ylabel='y-
axis')
ax1.legend()

ax2.set_title('Dataset-2')
ax2.scatter(df2['x'], df2['y'], marker='o', color=(0, 1, 0), label='data points') ax2.plot(df2['x'],
ls_model_y2, color='red', label='Least sqaure model') ax2.plot(df2['x'], ransac_model_y2,
color='blue', label='Ransac model') ax2.set(xlabel='x-axis', ylabel='y-axis')
ax2.legend() plt.show()
```

Output:



Console:

```
# inliers: 14
# prob_outlier: 0.944
# s: 1
# n: 17056.92657043659
# max_inliner_count: 14
# inliers: 13
# prob_outlier: 0.948
# s: 2
# n: 21304.062760691897
# max_inliner_count: 14
# inliers: 72
# prob_outlier: 0.712
# s: 3
# n: 123.90418270340892
# max_inliner_count: 72
# inliers: 39
# prob_outlier: 0.844
# s: 4
# n: 787.5960236371116
# max_inliner_count: 72
# inliers: 27
# prob_outlier: 0.892
# s: 5
# n: 2376.6106833968684
# max_inliner_count: 72
# inliers: 157
```

prob_outlier: 0.372
s: 6
n: 10.526691399992457
max_inliner_count: 157
inliers: 130
prob_outlier: 0.48
s: 7
n: 19.769880355510256
max_inliner_count: 157
inliers: 188
prob_outlier: 0.248
s: 8
n: 5.409062282538675
max_inliner_count: 188
inliers: 187
prob_outlier: 0.252
s: 1
n: 5.525552495791251
max_inliner_count: 187
inliers: 55
prob_outlier: 0.78
s: 2
n: 279.841710984081
max_inliner_count: 187
inliers: 201
prob_outlier: 0.19599999999999995
s: 3
n: 4.08481384841806
max_inliner_count: 201
inliers: 191
prob_outlier: 0.236
s: 4
n: 5.073307052973716
max_inliner_count: 201
inliers: 67
prob_outlier: 0.732
s: 5
n: 154.12918046553153
max_inliner_count: 201
inliers: 184
prob_outlier: 0.264
s: 6

n: 5.889670195389181
max_inliner_count: 201

Part B
Aim: Perform Colorization

Steps:
Packages Used: torch, argparse, ipython, scikit-image, matplotlib, colorizer (local package)

Command:

pip install torch

On PyCharm Terminal

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install torch
Installing collected packages: mpmath, typing-extensions, sympy, networkx, MarkupSafe, fsspec, filelock, jinja2, torch
Successfully installed MarkupSafe-3.0.2 filelock-3.16.1 fsspec-2024.12.0 jinja2-3.1.5 mpmath-1.3.0 networkx-3.4.2 sympy
-1.13.1 torch-2.5.1 typing-extensions-4.12.2
```

pip install argparse On PyCharm Terminal

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install argparse
Collecting argparse
  Using cached argparse-1.4.0-py2.py3-none-any.whl.metadata (2.8 kB)
Using cached argparse-1.4.0-py2.py3-none-any.whl (23 kB)
Installing collected packages: argparse
Successfully installed argparse-1.4.0
```

pip install ipython

On PyCharm Terminal

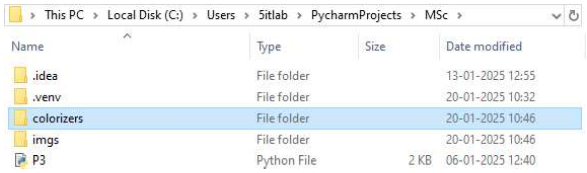
```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install ipython
Installing collected packages: wcwidth, pure-eval, traitlets, pygments, prompt_toolkit, parso, executing, decorator, co
lorama, asttokens, stack_data, matplotlib-inline, jedi, ipython
Successfully installed asttokens-3.0.0 colorama-0.4.6 decorator-5.1.1 executing-2.1.0 ipython-8.31.0 jedi-0.19.2 matplo
tlib-inline-0.1.7 parso-0.8.4 prompt_toolkit-3.0.48 pure-eval-0.2.3 pygments-2.19.1 stack_data-0.6.3 traitlets-5.14.3 w
cwidth-0.2.13
```

pip install scikit-image On PyCharm Terminal

```
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install scikit-image
Installing collected packages: tifffile, scipy, pillow, packaging, lazy-loader, imageio, scikit-image
Successfully installed imageio-2.37.0 lazy-loader-0.4 packaging-24.2 pillow-11.1.0 scikit-image-0.25.0 scipy-1.15.1 tif
(.venv) PS C:\Users\5itlab\PycharmProjects\MSc> pip install matplotlib
```

```
Installing collected packages: six, pyparsing, kiwisolver, fonttools, cycler, contourpy, python-dateutil, matplotlib
Successfully installed contourpy-1.3.1 cycler-0.12.1 fonttools-4.55.3 kiwisolver-1.4.8 matplotlib-3.10.0 pyparsing-3.2.
1 python-dateutil-2.9.0.post0 six-1.17.0
```

Open the project folder and add the **colorizer** package in the project.



Note: If error is found in installing packages on both PyCharm Terminal and Command Prompt, execute the following command and then try again to install packages:

<<Project Path>>\.venv\Scripts\activate

```
C:\Users\5itlab>C:\Users\5itlab\PycharmProjects\MSc\.venv\Scripts\activate

(.venv) C:\Users\5itlab>
```

Images Used:
imgs>>ansel_adams.jpg, ansel_adams2.jpg, ansel_adams3.jpg, ILSVRC2012_val_00041580.JPEG, ILSVRC2012_val_00046524.JPEG, ILSVRC2012_val_00046834.JPEG

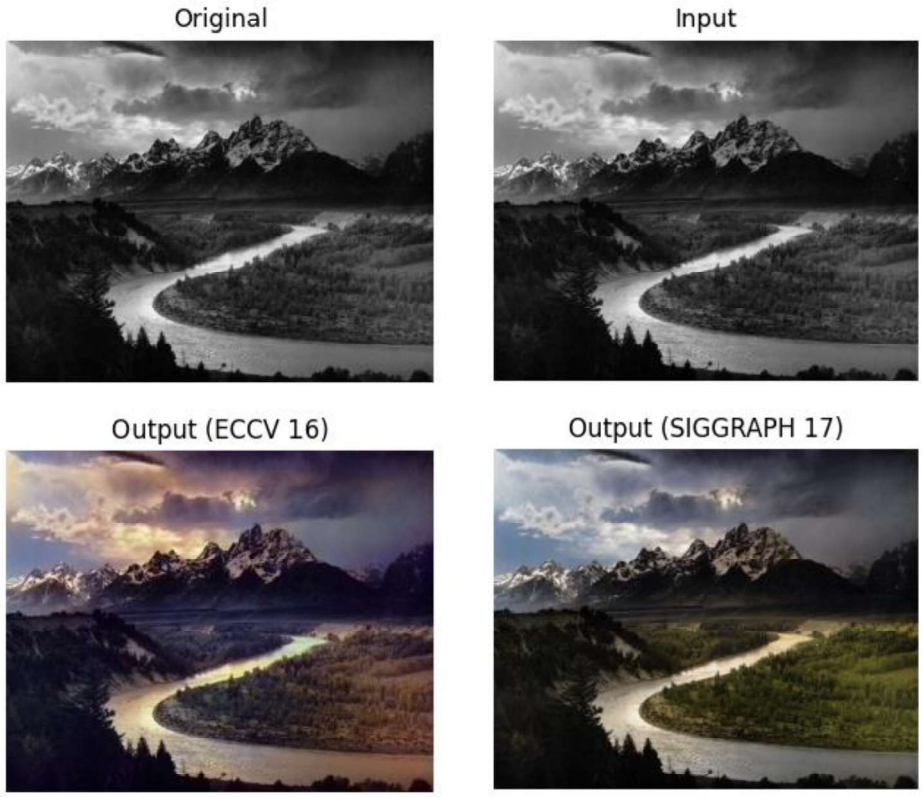
Code:

```
import argparse
import matplotlib.pyplot as plt
from colorizers import * #local package

# load colorizers
colorizer_eccv16 = eccv16(pretrained=True).eval() colorizer_siggraph17 = siggraph17(pretrained=True).eval()

# default size to process images is 256x256
# grab L channel in both original ("orig") and resized ("rs") resolutions img = load_img('imgs/ansel_adams.jpg')
(tens_l_orig, tens_l_rs) = preprocess_img(img, HW=(256,256))

# colorizer outputs 256x256 ab map
# resize and concatenate to original L channel img_bw = postprocess_tens(tens_l_orig,
torch.cat((0*tens_l_orig,0*tens_l_orig),dim=1))
out_img_eccv16 = postprocess_tens(tens_l_orig, colorizer_eccv16(tens_l_rs).cpu())
out_img_siggraph17 = postprocess_tens(tens_l_orig, colorizer_siggraph17(tens_l_rs).cpu())
```

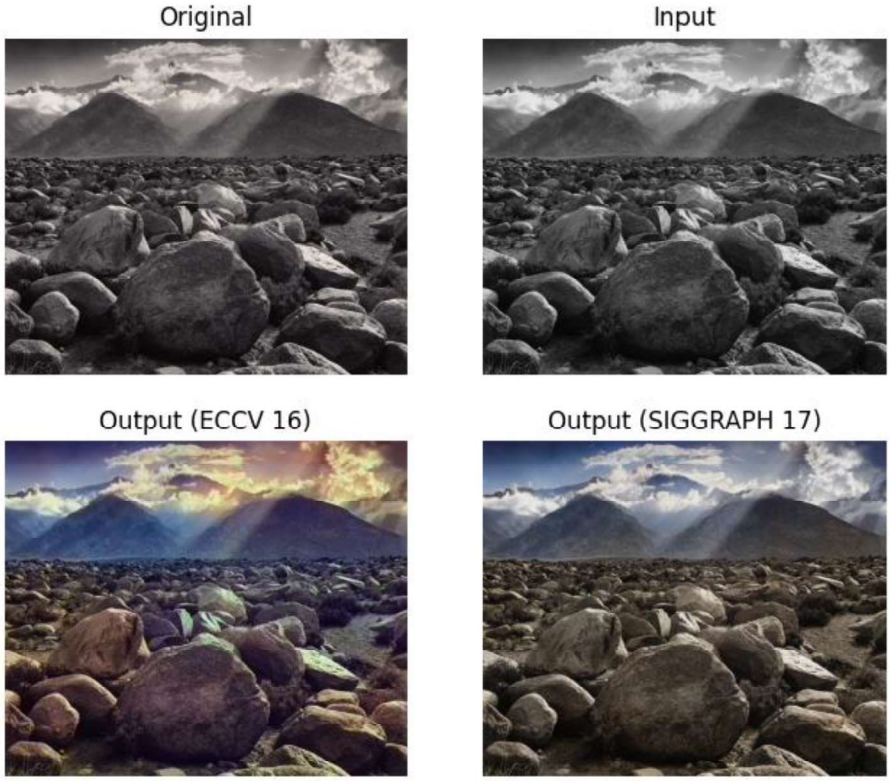
```
plt.imsave('s_eccv16.png', out_img_eccv16) plt.imsave('s_siggraph17.png',
out_img_siggraph17)

plt.figure(figsize=(12,8))
plt.subplot(2,2,1)
plt.imshow(img)
plt.title('Original')
plt.axis('off')

plt.subplot(2,2,2)
plt.imshow(img_bw)
plt.title('Input') plt.axis('off')

plt.subplot(2,2,3)
plt.imshow(out_img_eccv16)
plt.title('Output (ECCV 16)') plt.axis('off')

plt.subplot(2,2,4)
plt.imshow(out_img_siggraph17)
plt.title('Output (SIGGRAPH 17)')
plt.axis('off')
plt.show()
```

Output:

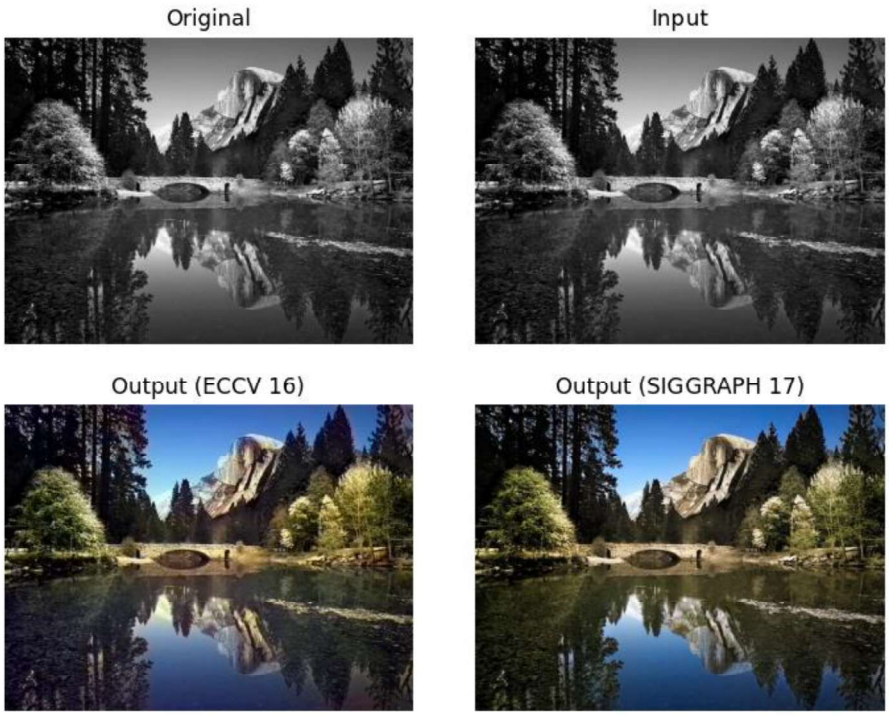
Case – I (ansel_adams.jpg)


Original
Input
Output (ECCV 16)
Output (SIGGRAPH 17)

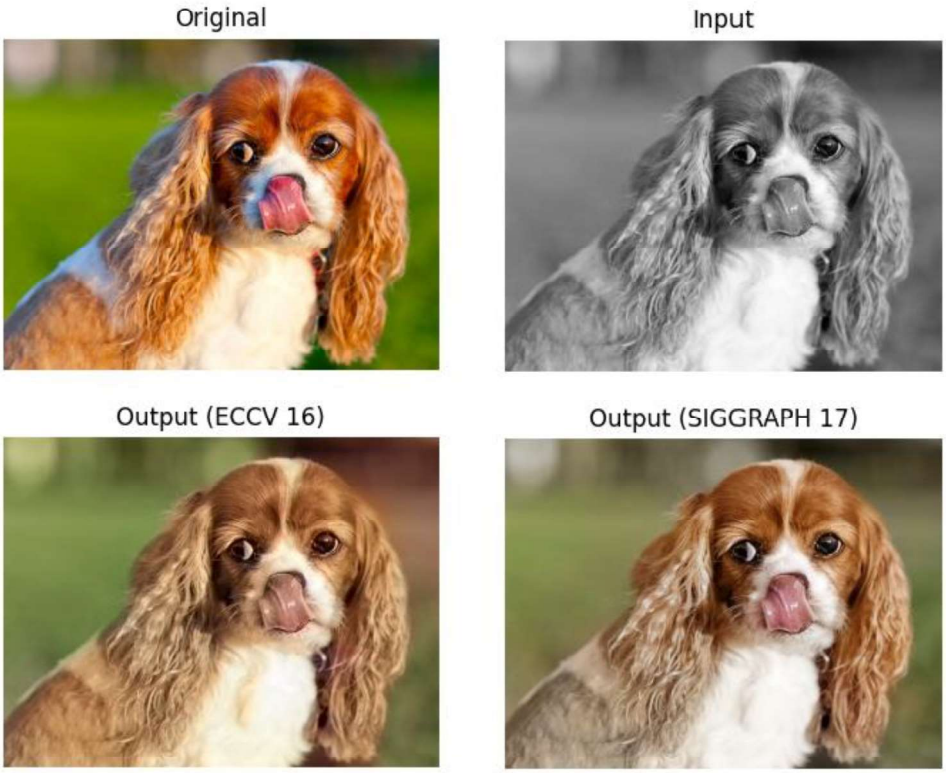
Case – II (ansel_adams2.jpg)



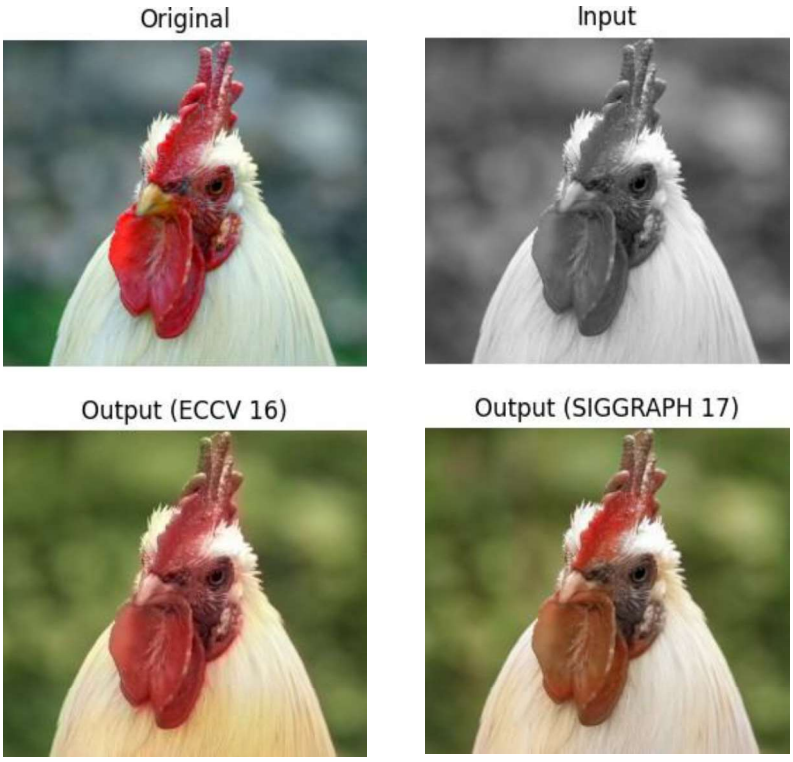
Case – III (ansel_adams3.jpg)



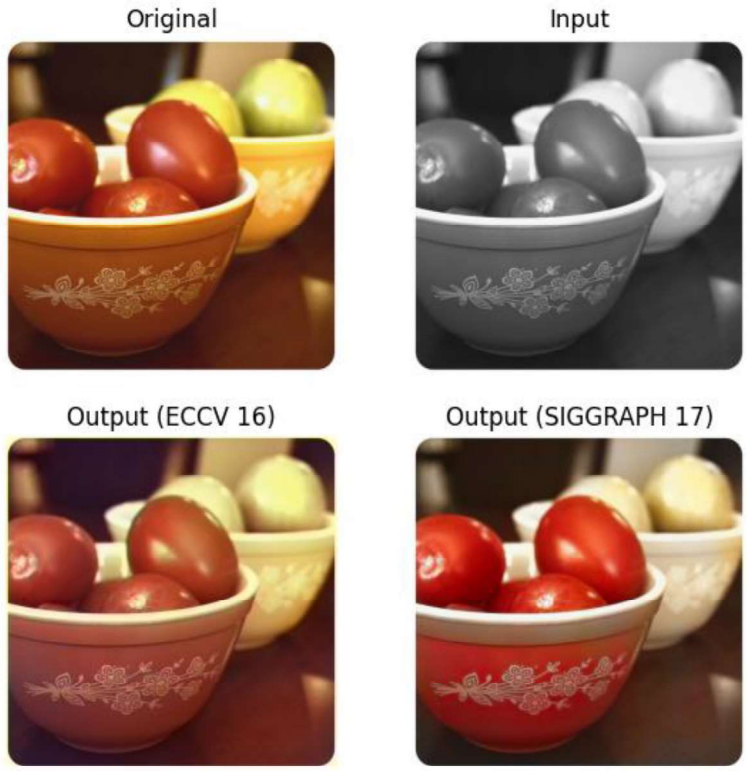
Case – IV (ILSVRC2012_val_00041580.JPEG)



Case – V (ILSVRC2012_val_00046524.JPEG)



Case – VI (ILSVRC2012_val_00046834.JPEG)



Practical No. 4

Part A :
Aim: Perform Text Detection and Recognition

Concepts :
Text detection is the process of **identifying the presence and location of text** in an image or video frame. It does **not** recognize the content of the text; it only detects **where text is present**.

Text recognition (also known as OCR – Optical Character Recognition) is the process of extracting and converting detected text from images into a machine-readable format.

Packages Used :
pytesseract is a Python wrapper for **Tesseract OCR**, an open-source Optical Character Recognition (OCR) engine developed by Google. It allows Python programs to extract text from images easily.

- **OCR for Documents –** Convert scanned PDFs and images into searchable text.
- **License Plate Recognition –** Read text from vehicle plates.
- **Text Extraction from Screenshots –** Extract text from UI screenshots.
- **Real-time OCR in Videos –** Use OCR on video frames for real-time applications.
- **Handwriting Recognition –** Recognize handwritten text (with limitations).

pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe' specifies the installed location of Tesseract OCR on the system.
img = cv2.imread("img.jpg"): reads the input image from the file "img.jpg".

- "img.jpg": The file name of the image to be processed.
- Returns an image matrix.

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY): converts the image from BGR (color) format to grayscale.

- img: Input image.
- cv2.COLOR_BGR2GRAY: Conversion code to transform the image into grayscale.
- Returns a grayscale image.

ret, thresh1 = cv2.threshold(gray, 0, 255, cv2.THRESH_OTSU | cv2.THRESH_BINARY_INV): applies thresholding to convert the grayscale image into a binary image.

- gray: Input grayscale image.
- 0: Threshold value (ignored when using Otsu's method).
- 255: Maximum pixel value.

- cv2.THRESH_OTSU | cv2.THRESH_BINARY_INV: Otsu's binarization with inverse thresholding.
- Returns a binary thresholded image.

rect_kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (20, 20)): creates a rectangular kernel for morphological operations.

- cv2.MORPH_RECT: Specifies a rectangular structuring element.
- (20, 20): Kernel size (width, height).
- Returns a structuring element.

dilation = cv2.dilate(thresh1, rect_kernel, iterations=1): applies dilation to enhance text regions.

- thresh1: Input binary image.
- rect_kernel: Structuring element used for dilation.
- iterations=1: Number of times dilation is applied.
- Returns the dilated image.

contours, hierarchy = cv2.findContours(dilation, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE): detects contours in the dilated image.

- dilation: Input binary image.
- cv2.RETR_EXTERNAL: Retrieves only the outermost contours.
- cv2.CHAIN_APPROX_NONE: Stores all contour points without approximation.
- Returns a list of detected contours.

im2 = img.copy(): creates a copy of the original image for drawing rectangles. **for cnt in contours:** iterates through each detected contour.

x, y, w, h = cv2.boundingRect(cnt): calculates the bounding box around a contour.

- cnt: A detected contour.
- Returns x (top-left x-coordinate), y (top-left y-coordinate), w (width), h (height) of the bounding box.

rect = cv2.rectangle(im2, (x, y), (x + w, y + h), (0, 255, 0), 2): draws a green rectangle around the detected text.

- im2: The image where the rectangle will be drawn.
- (x, y): Top-left corner of the rectangle.
- (x + w, y + h): Bottom-right corner of the rectangle.
- (0, 255, 0): Green color (BGR format).
- 2: Thickness of the rectangle border.

cropped = im2[y: y + h, x: x + w]: extracts the region of interest (ROI) containing the detected text.

- im2[y: y + h, x: x + w]: Crops the bounding box region.
- Returns the cropped image.

text = pytesseract.image_to_string(cropped): extracts text from the cropped region using Tesseract OCR.

- cropped: Input image containing text.
- Returns the extracted text as a string. **print(text)**:

prints the extracted text to the console.

Part B :

Aim : Perform Image Matting and Composting

Concept :

Image Matting is the process of extracting the foreground object from an image while accurately preserving fine details like hair, fur, or transparency. It is widely used in photo editing, film production, virtual backgrounds, and AR applications.

Image Compositing is the process of combining multiple images to create a seamless, realistic final image. After matting extracts the foreground, compositing blends it with a new background.

Packages Used :

PyMatting is a Python library for image matting that provides efficient implementations of classical and deep learning-based matting algorithms. It is useful for extracting the foreground from an image while preserving fine details like hair, fur, and transparency.

scale = 1.0: defines a scaling factor for resizing images.

image = load_image("in_lemur.png", "RGB", scale, "box"): loads the input image in RGB mode with the specified scaling.

- "in_lemur.png": File name of the input image.
- "RGB": Color mode for loading the image.
- scale: Resizing factor for the image.
- "box": Interpolation method used for resizing.
- Returns an RGB image.

trimap = load_image("in_lemur_trimap.png", "GRAY", scale, "box"): loads the trimap image in grayscale mode.

- "in_lemur_trimap.png": File name of the trimap.
- "GRAY": Grayscale mode for loading the image.
- scale: Resizing factor for the image.
- "box": Interpolation method used for resizing.
- Returns a grayscale image.

alpha = estimate_alpha_cf(image, trimap): estimates the alpha matte using the input image and trimap.

- image: Input RGB image.
- trimap: Grayscale trimap used to distinguish foreground, background, and unknown regions.
- Returns an estimated alpha matte.

new_background = np.zeros(image.shape): creates a new background image filled with zeros (black).

- image.shape: Dimensions of the input image.

- Returns a black image with the same shape as the input. **new_background[:, :] = [0.5, 0.5, 0.5]**: fills the new background with a gray color.
- [0.5, 0.5, 0.5]: RGB values for gray (normalized between 0 and 1). **foreground, background = estimate_foreground_ml(image, alpha, return_background=True):** estimates the foreground and background based on the alpha matte.
- image: Input RGB image.
- alpha: Estimated alpha matte.
- return_background=True: Returns both the extracted foreground and background.
- Returnsthe separated foreground and background images. **save_image("output1_alpha.png", alpha):** saves the estimated alpha matte as an image file.
- "output1_alpha.png": File name for saving the alpha matte.
- alpha: Alpha matte to be saved.

save_image("output2_foreground.png", foreground): saves the extracted foreground image.
- "output2_foreground.png": File name for saving the foreground.
- foreground: Extracted foreground image. **save_image("output3_background.png", background):** saves the extracted background image.
- "output3_background.png": File name for saving the background.
- background: Extracted background image.

cutout = stack_images(foreground, alpha): combines the foreground and alpha matte into a single image.
- foreground: Extracted foreground image.
- alpha: Alpha matte for transparency.
- Returns an image with foreground and alpha combined. **save_image("output4_cutout.png", cutout):** saves the cutout image with transparency.
- "output4_cutout.png": File name for saving the cutout.
- cutout: Image containing the foreground with transparency.

cutout = cv2.imread("output4_cutout.png", cv2.IMREAD_UNCHANGED): loads the cutout image with an alpha channel.
- "output4_cutout.png": File name of the cutout image.
- cv2.IMREAD_UNCHANGED: Reads the image while preserving the alpha channel.
- Returns an image with an alpha channel.

new_background = cv2.imread("in_beach.jpg"): loads the new background image.
- "in_beach.jpg": File name of the new background image.
- Returns a background image.

foreground = cutout[:, :, :3]: extracts the RGB channels from the cutout image.
- cutout[:, :, :3]: Extracts the first three channels (RGB).

- Returns the foreground image without the alpha channel.

am = cutout[:, :, 3] / 255.0: extracts the alpha channel and normalizes it to a range of 0 to 1.

- cutout[:, :, 3]: Extracts the alpha channel.
- / 255.0: Normalizes the alpha values.
- Returns a normalized alpha matte.

newbg_resized = cv2.resize(new_background, (foreground.shape[1], foreground.shape[0])): resizes the new background to match the size of the foreground.

- new_background: The new background image.
- (foreground.shape[1], foreground.shape[0]): Target width and height from the foreground.
- Returns a resized background image.

composite = np.zeros_like(foreground, dtype=np.uint8): creates a blank composite image with the same size as the foreground.

- np.zeros_like(foreground): Creates a zero matrix with the same shape as the foreground.
- dtype=np.uint8: Data type for image storage.
- Returns an empty black image.

for c in range(3): iterates over the three color channels (RGB).

composite[:, :, c] = foreground[:, :, c] * am + newbg_resized[:, :, c] * (1 - am): blends the foreground and background using the alpha matte.

- foreground[:, :, c] * am: Multiplies the foreground color by the alpha values.
- newbg_resized[:, :, c] * (1 - am): Multiplies the background color by the inverse alpha values.
- Adds the two values to create a blended image.

cv2.imwrite("CompositeImageBeach.png", composite): saves the final composite image with the new background.

- "CompositeImageBeach.png": File name for saving the final result.
- composite: The blended image with the new background.

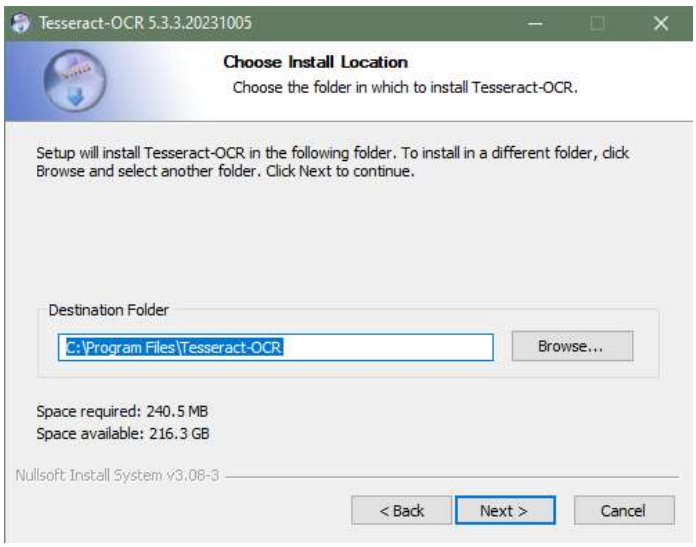
Practical No. 4

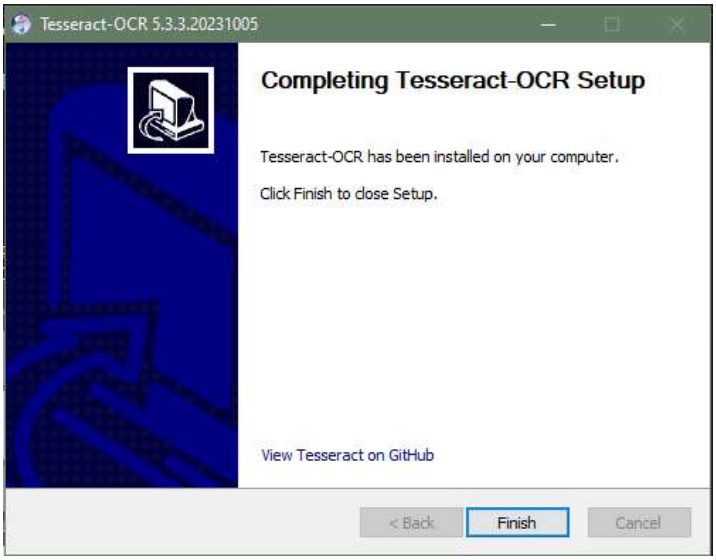
Part A
Aim: Perform Text Detection and Recognition

Steps:
Packages Used: cv2, pytesseract

Pytesseract Installation Guide: Download the PyTesseract Installer

Check the Install Location&Finish the installation





Command:

pip install pytesseract

Images Used:
img.jpg sample.jpg

Code:

```
import cv2
import pytesseract

# Mention the installed location of Tesseract OCR
pytesseract.pytesseract.tesseract_cmd=r'C:\Program
Files\Tesseract-OCR\tesseract.exe'

img = cv2.imread("img.jpg")
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

ret, thresh1 = cv2.threshold(gray, 0, 255, cv2.THRESH_OTSU | cv2.THRESH_BINARY_INV)

rect_kernel = cv2.getStructuringElement(cv2.MORPH_RECT,(20, 20)) dilation =
cv2.dilate(thresh1, rect_kernel, iterations=1)
contours, heirarchy = cv2.findContours(dilation, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_NONE)

im2 = img.copy()
```

```
for cnt in contours:
    x, y, w, h = cv2.boundingRect(cnt)
    rect = cv2.rectangle(im2, (x, y), (x+w, y+h), (0,255,0), 2) cropped =
    im2[y: y+h, x: x+w]
    text = pytesseract.image_to_string(cropped)
    print(text)
```

Output:

Case – I (img.jpg):


```
Test

Hello World


Process finished with exit code 0
```

Case – II (sample.jpg)


```
Text is at different regions

Process finished with exit code 0
```

Part B
Aim: Perform Image Matting and Composting

Steps:
Packages Used: pymatting, numpy, cv2 Command:
pip install pymatting

Images Used:
in_lemur.png
in_lemur_trimap.png
in_beach.png

Code:

```
from pymatting import *
import numpy as np import
cv2

scale = 1.0
image = load_image("in_lemur.png", "RGB", scale, "box")
trimap = load_image("in_lemur_trimap.png", "GRAY", scale, "box") #MATTING
# Estimate Alpha from Image and Trimap alpha =
estimate_alpha_cf(image, trimap)

# Make gray background new_background =
np.zeros(image.shape) new_background[:, :] = [0.5,
0.5, 0.5]

# Estimate foreground from image and alpha
foreground, background = estimate_foreground_ml(image, alpha, return_background=True)

# Save Alpha save_image("output1_alpha.png",
alpha) # Save Foreground
save_image("output2_foreground.png", foreground) # Save
Background save_image("output3_background.png", background)
```

```
# Save Cutout
cutout = stack_images(foreground, alpha)
save_image("output4_cutout.png", cutout)

# COMPOSTING

cutout = cv2.imread("output4_cutout.png", cv2.IMREAD_UNCHANGED) new_background =
cv2.imread("in_beach.jpg")

# Split the cutout into foreground and alpha matte foreground =
cutout[:, :, :3]
am = cutout[:, :, 3] / 255.0

# Resize the new background to match the size of the foreground newbg_resized =
cv2.resize(new_background, (foreground.shape[1], foreground.shape[0]))

# Create composite image
composite = np.zeros_like(foreground, dtype=np.uint8)

# Composite the foreground onto the new background for c in
range(3):
    composite[:, :, c] = foreground[:, :, c] * am + newbg_resized[: , :, c] * (1-am)

# Save the composite image cv2.imwrite("CompositeImageBeach.png", composite)
```

Output:
output1_alpha.png

output3_background.png

output4_cutout.png

CompositeImageBeach.png