DC Project

Mayank Gupta

4899-8256

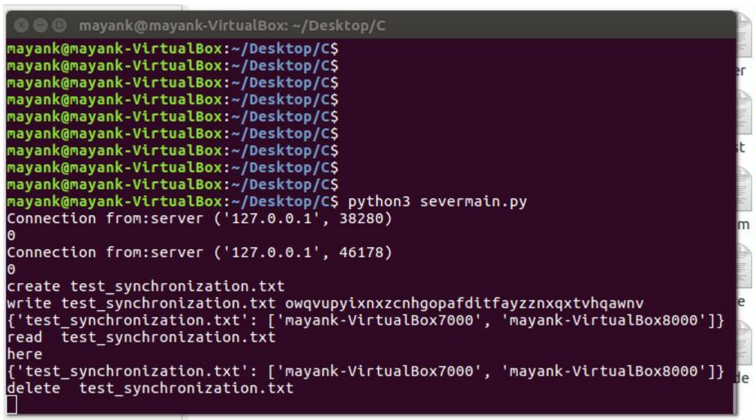
i. Describe how you would implement synchronization. Explain using pseudo-code.

For synchronization locks would have to be used in the server.Since there are 5 seperate threads running at one instance,
For this we need 2 locks,
one to prevent writes or deletes while reading
One to prevent deletion while writing

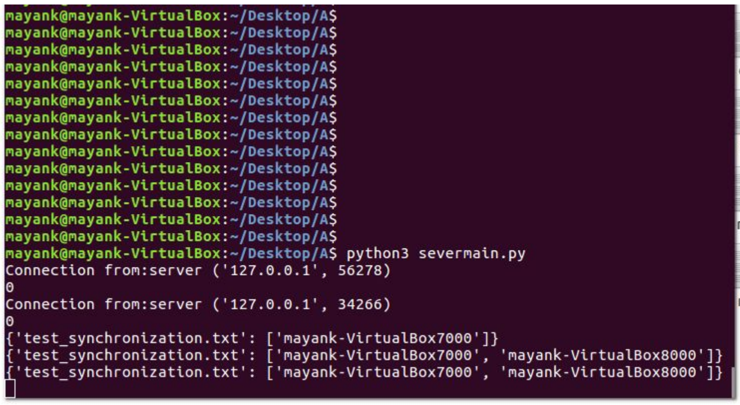
```
acquire.Lock()
read(filename)
release.Lock()

acquire.Lock2()
write(filename)
release.Lock2()
```

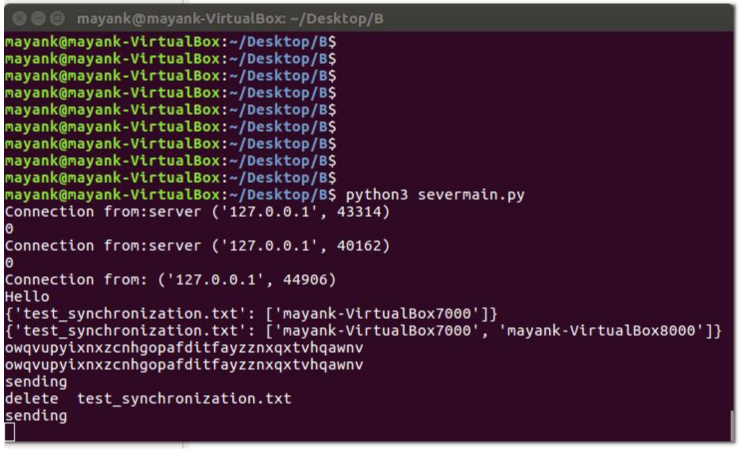
This lock will be shared the threads.



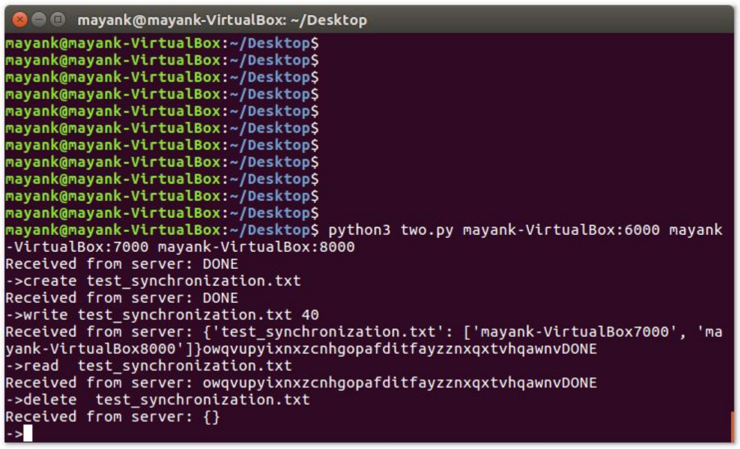
```
mayank@mayank-VirtualBox: ~/Desktop/C
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$ python3 severmain.py
Connection from:server ('127.0.0.1', 38280)
0
Connection from:server ('127.0.0.1', 46178)
0
create test_synchronization.txt
write test_synchronization.txt owqvupyixnxzcnhgopafditfayzznxqxtvhqawnv
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
read  test_synchronization.txt
here
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
delete  test_synchronization.txt
```



```
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$ python3 severmain.py
Connection from:server ('127.0.0.1', 56278)
0
Connection from:server ('127.0.0.1', 34266)
0
{'test_synchronization.txt': ['mayank-VirtualBox7000']}
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
```

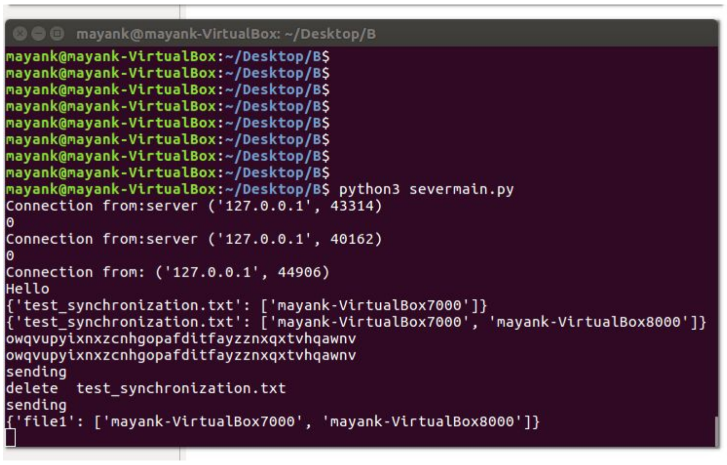


```
mayank@mayank-VirtualBox: ~/Desktop/B
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$ python3 severmain.py
Connection from:server ('127.0.0.1', 43314)
0
Connection from:server ('127.0.0.1', 40162)
0
Connection from: ('127.0.0.1', 44906)
Hello
{'test_synchronization.txt': ['mayank-VirtualBox7000']}
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
owqvupyixnxzcnhgopafditfayzznxqxtvhqawnv
owqvupyixnxzcnhgopafditfayzznxqxtvhqawnv
sending
delete  test_synchronization.txt
sending
```

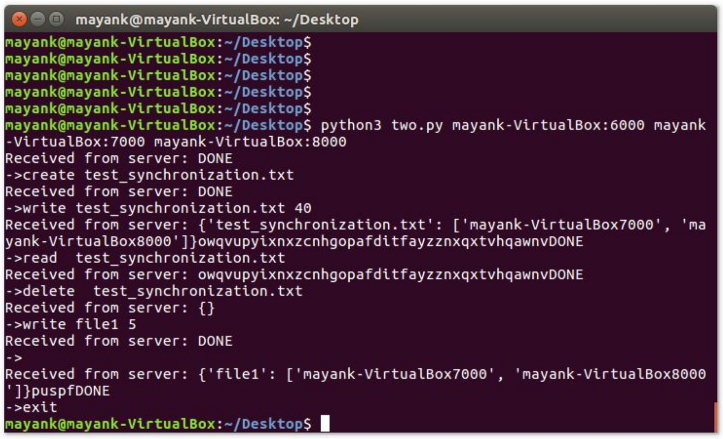


```
mayank@mayank-VirtualBox: ~/Desktop
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$ python3 two.py mayank-VirtualBox:6000 mayank
-VirtualBox:7000 mayank-VirtualBox:8000
Received from server: DONE
->create test_synchronization.txt
Received from server: DONE
->write test_synchronization.txt 40
Received from server: {'test_synchronization.txt': ['mayank-VirtualBox7000', 'ma
yank-VirtualBox8000']}owqvupyixnxzcnhgopafditfayzznxqxtvhqawnvDONE
->read  test_synchronization.txt
Received from server: owqvupyixnxzcnhgopafditfayzznxqxtvhqawnvDONE
->delete  test_synchronization.txt
Received from server: {}
->
```

Q2 Screenshot for create,write,read, delete

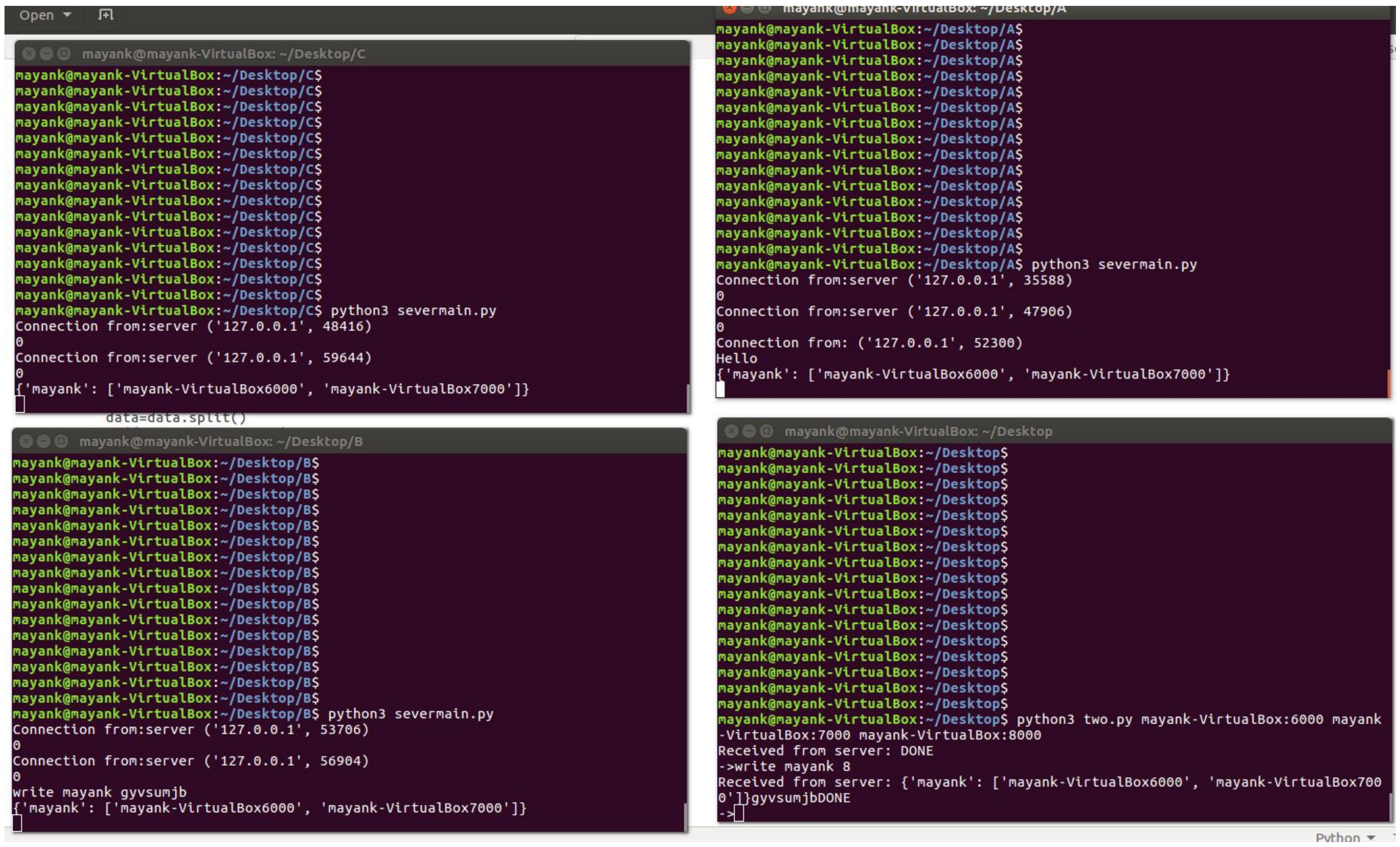


```
mayank@mayank-VirtualBox: ~/Desktop/B
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$ python3 severmain.py
Connection from:server ('127.0.0.1', 43314)
0
Connection from:server ('127.0.0.1', 40162)
0
Connection from: ('127.0.0.1', 44906)
Hello
{'test_synchronization.txt': ['mayank-VirtualBox7000']}
{'test_synchronization.txt': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
owqvupyixnxzcnhgopafditfayzznxqxtvhqawnv
owqvupyixnxzcnhgopafditfayzznxqxtvhqawnv
sending
delete  test_synchronization.txt
sending
{'file1': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000']}
```



```
mayank@mayank-VirtualBox: ~/Desktop
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$ python3 two.py mayank-VirtualBox:6000 mayank
-VirtualBox:7000 mayank-VirtualBox:8000
Received from server: DONE
->create test_synchronization.txt
Received from server: DONE
->write test_synchronization.txt 40
Received from server: {'test_synchronization.txt': ['mayank-VirtualBox7000', 'ma
yank-VirtualBox8000']}owqvupyixnxzcnhgopafditfayzznxqxtvhqawnvDONE
->read  test_synchronization.txt
Received from server: owqvupyixnxzcnhgopafditfayzznxqxtvhqawnvDONE
->delete  test_synchronization.txt
Received from server: {}
->write file1 5
Received from server: DONE
->
Received from server: {'file1': ['mayank-VirtualBox7000', 'mayank-VirtualBox8000
']}puspfDONE
->exit
mayank@mayank-VirtualBox:~/Desktop$
```

Q2 Screenshot showing hashtable after deleting test_synchronization.txt



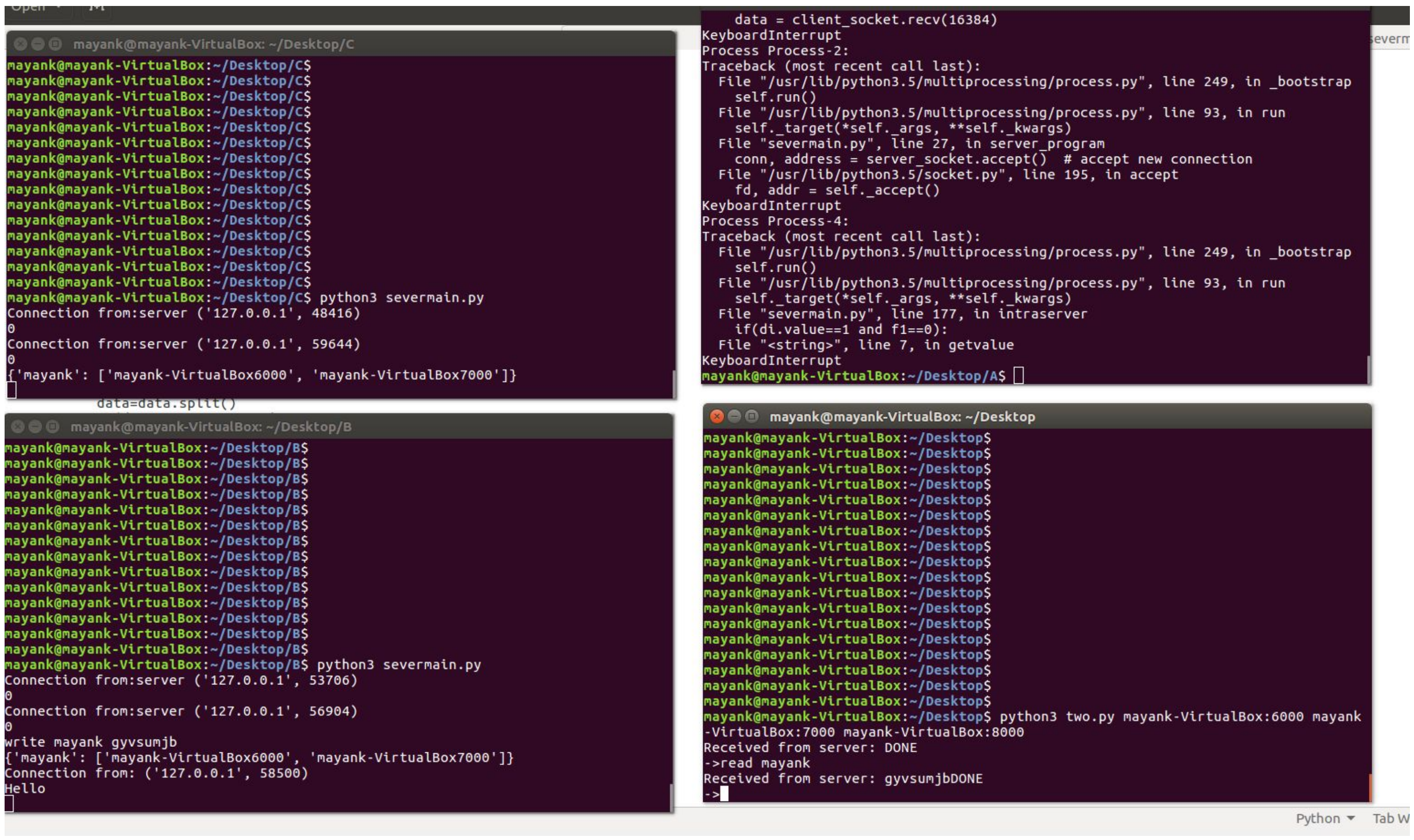
```
mayank@mayank-VirtualBox: ~/Desktop/C
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$ python3 severmain.py
Connection from:server ('127.0.0.1', 48416)
0
Connection from:server ('127.0.0.1', 59644)
0
{'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox7000']}
```

```
mayank@mayank-VirtualBox: ~/Desktop/A
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$
mayank@mayank-VirtualBox:~/Desktop/A$ python3 severmain.py
Connection from:server ('127.0.0.1', 35588)
0
Connection from:server ('127.0.0.1', 47906)
0
Connection from: ('127.0.0.1', 52300)
Hello
{'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox7000']}
```

```
        data=data.split()
```

```
mayank@mayank-VirtualBox: ~/Desktop/B
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$ python3 severmain.py
Connection from:server ('127.0.0.1', 53706)
0
Connection from:server ('127.0.0.1', 56904)
0
write mayank gyvsumjb
{'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox7000']}
```

```
mayank@mayank-VirtualBox: ~/Desktop
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$ python3 two.py mayank-VirtualBox:6000 mayank
-VirtualBox:7000 mayank-VirtualBox:8000
Received from server: DONE
->write mayank 8
Received from server: {'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox700
0']}gyvsumjbDONE
->
```



```
mayank@mayank-VirtualBox: ~/Desktop/C
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$
mayank@mayank-VirtualBox:~/Desktop/C$ python3 severmain.py
Connection from:server ('127.0.0.1', 48416)
0
Connection from:server ('127.0.0.1', 59644)
0
{'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox7000']}
```

```
    data = client_socket.recv(16384)
KeyboardInterrupt
Process Process-2:
Traceback (most recent call last):
  File "/usr/lib/python3.5/multiprocessing/process.py", line 249, in _bootstrap
    self.run()
  File "/usr/lib/python3.5/multiprocessing/process.py", line 93, in run
    self._target(*self._args, **self._kwargs)
  File "severmain.py", line 27, in server_program
    conn, address = server_socket.accept()  # accept new connection
  File "/usr/lib/python3.5/socket.py", line 195, in accept
    fd, addr = self._accept()
KeyboardInterrupt
Process Process-4:
Traceback (most recent call last):
  File "/usr/lib/python3.5/multiprocessing/process.py", line 249, in _bootstrap
    self.run()
  File "/usr/lib/python3.5/multiprocessing/process.py", line 93, in run
    self._target(*self._args, **self._kwargs)
  File "severmain.py", line 177, in intraserver
    if(di.value==1 and f1==0):
  File "<string>", line 7, in getvalue
KeyboardInterrupt
mayank@mayank-VirtualBox:~/Desktop/A$
```

```
        data=data.split()
```

```
mayank@mayank-VirtualBox: ~/Desktop/B
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$
mayank@mayank-VirtualBox:~/Desktop/B$ python3 severmain.py
Connection from:server ('127.0.0.1', 53706)
0
Connection from:server ('127.0.0.1', 56904)
0
write mayank gyvsumjb
{'mayank': ['mayank-VirtualBox6000', 'mayank-VirtualBox7000']}
Connection from: ('127.0.0.1', 58500)
Hello
```

```
mayank@mayank-VirtualBox: ~/Desktop
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$
mayank@mayank-VirtualBox:~/Desktop$ python3 two.py mayank-VirtualBox:6000 mayank
-VirtualBox:7000 mayank-VirtualBox:8000
Received from server: DONE
->read mayank
Received from server: gyvsumjbDONE
->
```

Q2 Screenshot for fault tolerance

Code for Server

```
import multiprocessing
import socket
import time
import pickle
def server_program(dict,count1,count2,q1,q2,r1,r2,rq1,rq2,req,di,dli,port=6000):

    # get the hostname
    host = socket.gethostname()
    host_str=str(host)+str(port)

    if(port==6000):
        port2=7000
    if(port==7000):
        port2=8000
    if(port==8000):
        port2=6000
    host_str2=str(host)+str(port2)
    server_socket = socket.socket()  # get instance

    server_socket.bind((host, port))  # bind host address and port together

    # configure how many client the server can listen simultaneously
    server_socket.listen(10)

    while True:

        conn, address = server_socket.accept()  # accept new connection
        print("Connection from: " + str(address))
        while True:
            data = conn.recv(16384).decode()
            if not data:
            # if data is not received break
                break

            temp=data

            data=data.split()
```

```
if(data[0]=='connect'):
        print("Hello")
if(data[0]=='write'):
        write(data[1],data[2])
        dict[data[1]]=[host_str,host_str2]

        q1.put(temp)
        q2.put(temp)
        count1.value=1
        count2.value=1
        rr=str(dict)
        conn.send(rr.encode())
        conn.send(data[2].encode())
if(data[0]=='read'):
        l1=dict[data[1]]
        if(l1.count(host_str)>0):
                k=read(data[1])
                conn.send(bytes(read(data[1]), 'utf-8'))

        else:
                rq1.put(temp)
                rq2.put(temp)
                r1.value=1
                r2.value=1
                while(1):
                        read1=req.get()
                        if read:
                                break

                print(read1)


                #r1.value=1
                conn.send(bytes(read1, 'utf-8'))
if(data[0]=='create'):
        create(data[1])
        dict[data[1]]=[host_str]
        q1.put(temp)
        q2.put(temp)
        count1.value=1
        count2.value=1
if(data[0]=='delete'):
        delete(data[1])
```

```
                                dict.pop(data[1])
                                rr=str(dict)
                                conn.send(rr.encode())
                                dli.put(temp)
                                di.value=1
                        data = 'DONE'
                        try:
                                conn.send(data.encode())  # send data to the client
                        except:
                                pass
        #conn.close()  # close the connection



def write(filename,content):
        f = open(filename, "a")
        f.write(content)
        f.close()



def create(filename):
        f= open(filename,"w+")
        f.close()

def read(filename):
        f = open(filename, "r")
        return(f.read())

def delete(filename):
        try:

                os.remove(filename)
        except:
                return False
def intraserver(port,q,shared_dict,count,f,r1,rq1,di,dli,req):
        host = socket.gethostname()
        server_socket = socket.socket()
        server_socket.bind((host, port))
        server_socket.listen(2)

        while 1:
                conn, address = server_socket.accept()
                print("Connection from:server " + str(address))
                print(count.value)
```

```
serialized_dict = pickle.dumps(shared_dict.copy())
#data=serialized_dict.encode()
#count.value=0
conn.send(serialized_dict)

while 1:

        if(count.value==1 and f==0):

                print(shared_dict)
                serialized_dict = pickle.dumps(shared_dict.copy())
                #data=serialized_dict.encode()
                conn.send(serialized_dict)
                ("here")

                count.value=0

        if(count.value==1 and f==1):
                #serialized_dict = pickle.dumps(shared_dict.copy())
                #data=serialized_dict.encode()
                #conn.send(serialized_dict )
                #("here")
                while(q.empty() is False):
                        out=q.get()
                #print(out)
                out=pickle.dumps(out)
                #out=out.encode()
                conn.send(out)

                count.value=0

        if(r1.value==1 and f==0):
                #serialized_dict = pickle.dumps(shared_dict.copy())
                #data=serialized_dict.encode()
                #conn.send(serialized_dict)
                while(rq1.empty() is False):
                        out1=rq1.get()
                #print("out1")
                out1=pickle.dumps(out1)
                #out1=out1.encode()

                conn.send(out1)
                while 1:
```

```
                                read=conn.recv(16384)

                                if read:
                                        break
                        read=pickle.loads(read)
                        req.put(read)
                        #print("hello")
                        print(read)
                        r1.value=0

                if(di.value==1 and f1==0):
                        serialized_dict = pickle.dumps(shared_dict.copy())
                        conn.send(serialized_dict)
                        di.value=0
                if(di.value==1 and f1==1):
                        serialized_dict = pickle.dumps(shared_dict.copy())
                        conn.send(serialized_dict)
                        print("sending")
                        try:
                                while(dli.empty() is False):
                                        out2=dli.get()
                                print(out2)
                                out2=pickle.dumps(out2)
                                conn.send(out2)
                        except:
                                pass
                        di.value=0
                if(hasher.value==1):
                        serialized_dict = pickle.dumps(shared_dict.copy())
                        conn.send(serialized_dict)
                        hasher.value=0

def intraclient(port,q,shared_dict):

        host = socket.gethostname()
        host_str=str(host)+str(port)

        port1=6000
        if(port==7002):
                port2=7000
        if(port==8001):
                port2=8000
        host_str1=str(host)+str(port1)
```

```
host_str2=str(host)+str(port2)
client_socket = socket.socket()
while 1:
        try:
                client_socket.connect((host, port))
                break

        except:
                m=1

try:
        data = client_socket.recv(16384)
        dta=pickle.loads(data)
        dit.update(dta)
        #print(dict)
        #print(dta)
except:
        pass
while 1:
        data = client_socket.recv(16384)
        if data:
                dta=pickle.loads(data)
                #print(dta)
                #print(type(dta))

                if isinstance(dta, str)==True:
                        print(dta)
                        dta=dta.split()
                        if(dta[0]=='read'):
                                k=read(dta[1])
                                print("here")
                                sreial=pickle.dumps(k)
                                client_socket.send(sreial)
                        if(dta[0]=='create'):
                                create(dta[1])
                        if(dta[0]=='write'):
                                write(dta[1],dta[2])
                                shared_dict[dta[1]]=[host_str2,host_str1]
                                print(shared_dict)
                        if(dta[0]=='delete'):
                                delete(dta[1])
                                shared_dict.pop(dta[1])
                if isinstance(dta, dict)==True:
                        shared_dict.update(dta)
                        print(shared_dict)
            #data = client_socket.recv(16384).decode()
            #print(data)
            #dta=json.loads(data)


    #client_socket.close()

if __name__ == '__main__':
    manager = multiprocessing.Manager()
    count1 = multiprocessing.Value('i', 0)
    count2 = multiprocessing.Value('i', 0)
    f1=1
    f2=0
    req=multiprocessing.Queue()
    hasher= multiprocessing.Value('i', 0)
    r1=multiprocessing.Value('i', 0)
    r2=multiprocessing.Value('i', 0)
    rq1=multiprocessing.Queue()
    rq2=multiprocessing.Queue()
    di=multiprocessing.Value('i',0)
    dli=multiprocessing.Queue()
    shared_dict = manager.dict()
    q1=multiprocessing.Queue()
    q2=multiprocessing.Queue()
    p1 = multiprocessing.Process(target=server_program,
args=(shared_dict,count1,count2,q1,q2,r1,r2,rq1,rq2,req,di,dli,6000))
    p2 = multiprocessing.Process(target=intraserver, args=(6001,q1,shared_dict,count1,f1,r1,rq1,di,dli,req))
    p3 = multiprocessing.Process(target=intraserver, args=(6002,q2,shared_dict,count2,f2,r2,rq2,di,dli,req))
    p4 = multiprocessing.Process(target=intraclient, args=(8001,q1,shared_dict))
    p5 = multiprocessing.Process(target=intraclient, args=(7002,q2,shared_dict))
    #p4 = multiprocessing.Process(target=editq, args=(q,))
    p1.start()
    p2.start()
    p3.start()
    p4.start()
    p5.start()
    p1.join()
    p2.join()
    p3.join()
    p4.join()
    p5.join()
```

Code for Client

```
import socket
import random
import string
import sys
def client_program():
    IP1=sys.argv[1]
    IP2=sys.argv[2]
    IP3=sys.argv[3]
    cell=[IP1,IP2,IP3]
    final=random.choice(cell)
    IP=final.split(':')
    host = IP[0]  # as both code is running on same pc
    port = int(IP[1])  # socket server port number

    client_socket = socket.socket()  # instantiate
    client_socket.connect((host, port))  # connect to the server
    message='connect'
    while message.lower().strip() != 'exit':
        client_socket.send(message.encode())  # send message
        data = client_socket.recv(1024).decode()  # receive response

        print('Received from server: ' + data)  # show in terminal
        data=None
        while data==None:
            data1=input("->")
            message=data1
            if(message[:5]=='write'):
                k=randomString(int(message.split()[2]))
                message = message.split()[0] + ' ' +message.split()[1] + ' ' + k
            break
        #message = input(" -> ")  # again take input
        #message='bye'
    client_socket.close()  # close the connection



def randomString(stringLength=8):
    letters = string.ascii_lowercase
    return ( ''.join(random.choice(letters) for i in range(stringLength)) )
```

```
if __name__ == '__main__':
        client_program()
```