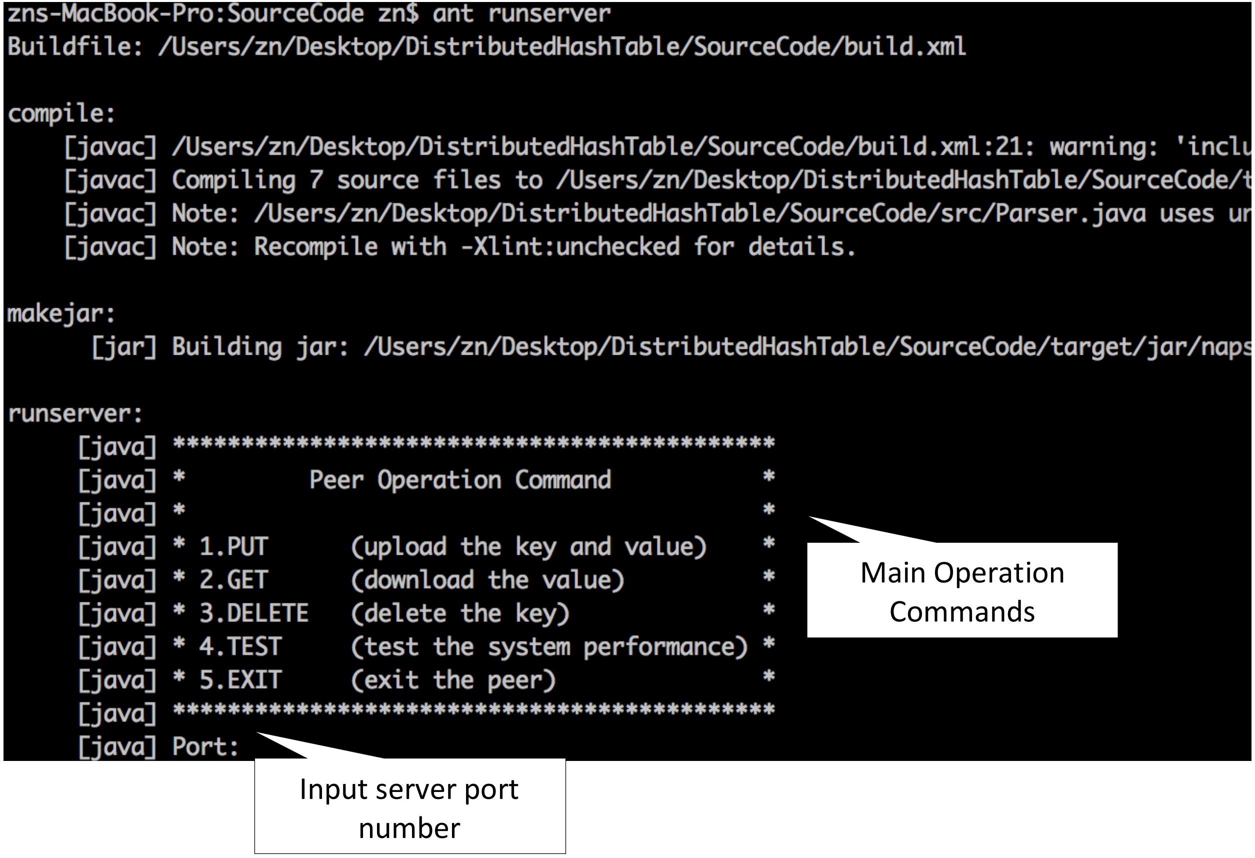
Output file

Name: Ning Zhang
CWID: A20336916
Department of Computer Science, Illinois Institute of Technology

1.when the server is running, it will output this results as following.

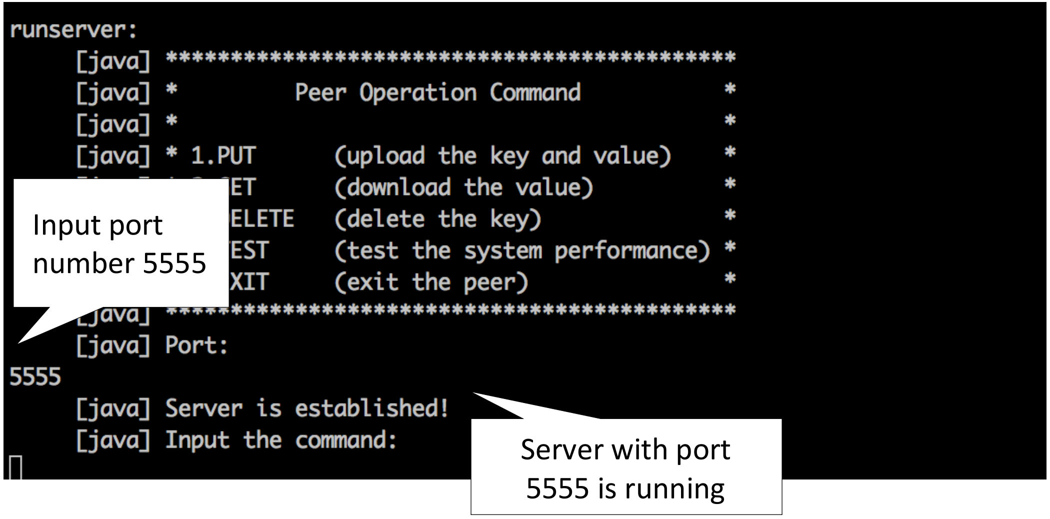


```
zns-MacBook-Pro:SourceCode zn$ ant runserver
Buildfile: /Users/zn/Desktop/DistributedHashTable/SourceCode/build.xml

compile:
    [javac] /Users/zn/Desktop/DistributedHashTable/SourceCode/build.xml:21: warning: 'inclu
    [javac] Compiling 7 source files to /Users/zn/Desktop/DistributedHashTable/SourceCode/t
    [javac] Note: /Users/zn/Desktop/DistributedHashTable/SourceCode/src/Parser.java uses ur
    [javac] Note: Recompile with -Xlint:unchecked for details.

makejar:
      [jar] Building jar: /Users/zn/Desktop/DistributedHashTable/SourceCode/target/jar/naps

runserver:
     [java] ********************************************
     [java] *         Peer Operation Command           *
     [java] *                                          *
     [java] * 1.PUT      (upload the key and value)    *
     [java] * 2.GET      (download the value)          *
     [java] * 3.DELETE   (delete the key)              *
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
     [java] ********************************************
     [java] Port:
```

Main Operation Commands

Input server port number

2. Input server port number, we can get the following result.

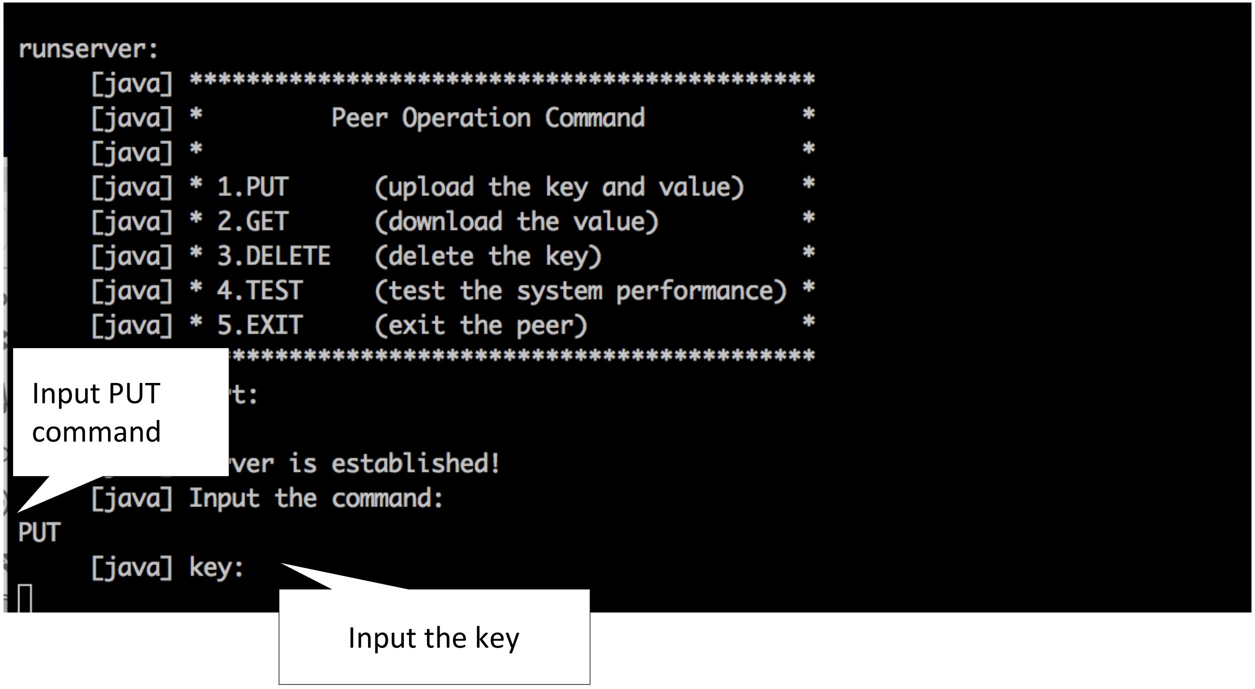


```
runserver:
     [java] ********************************************
     [java] *         Peer Operation Command           *
     [java] *                                          *
     [java] * 1.PUT      (upload the key and value)    *
                    ET      (download the value)          *
                    ELETE   (delete the key)              *
                    EST     (test the system performance) *
                    XIT     (exit the peer)               *
     java] ********************************************
     [java] Port:
5555
     [java] Server is established!
     [java] Input the command:
```

Input port number 5555

Server with port 5555 is running

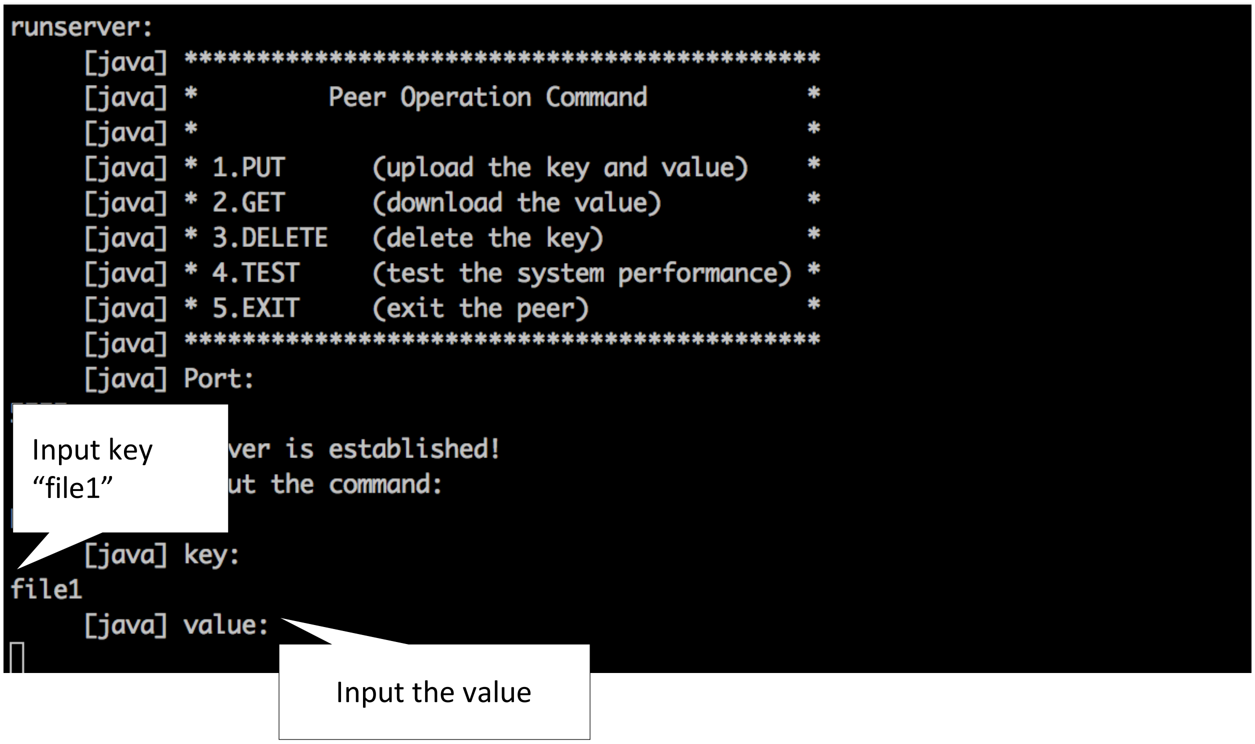
3.Input the PUT command, we can get the following results.



```
runserver:
     [java] ********************************************
     [java] *         Peer Operation Command           *
     [java] *                                          *
     [java] * 1.PUT      (upload the key and value)    *
     [java] * 2.GET      (download the value)          *
     [java] * 3.DELETE   (delete the key)              *
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
               *****************************************
               t:

               ver is established!
     [java] Input the command:
PUT
     [java] key:
```

Input PUT command

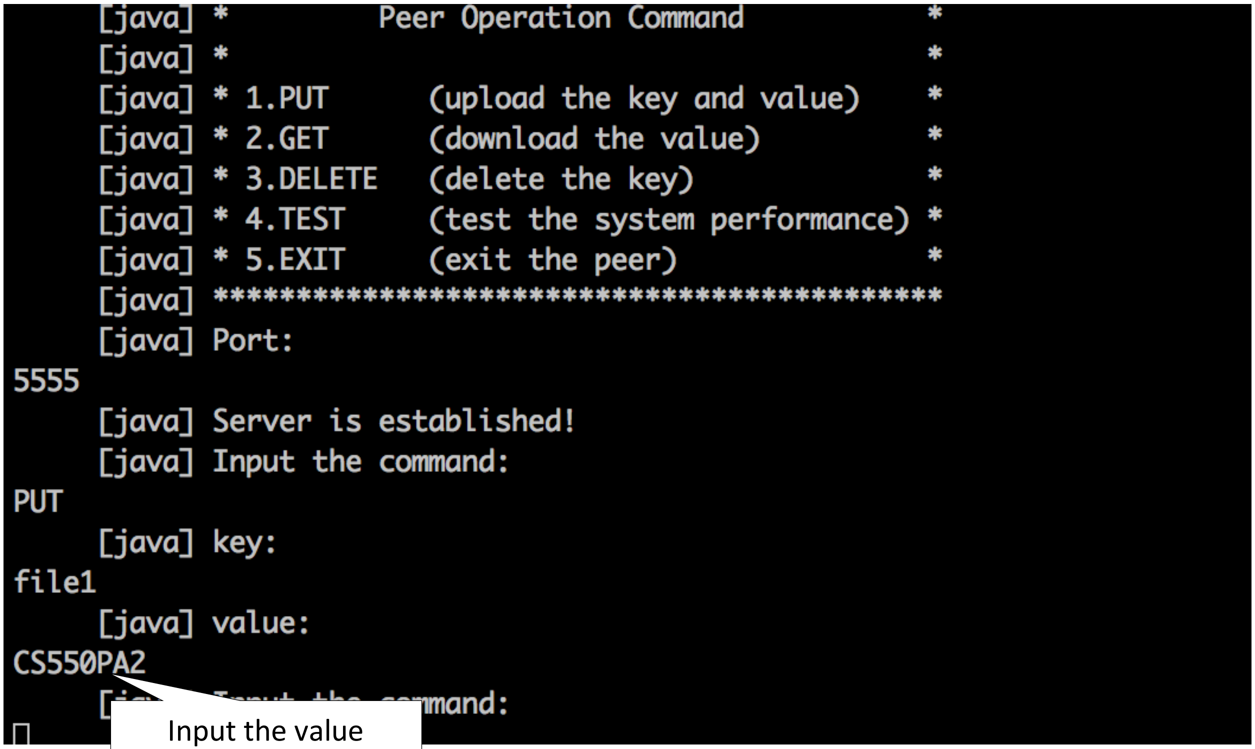
Input the key



```
runserver:
     [java] ********************************************
     [java] *         Peer Operation Command           *
     [java] *                                          *
     [java] * 1.PUT      (upload the key and value)    *
     [java] * 2.GET      (download the value)          *
     [java] * 3.DELETE   (delete the key)              *
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
     [java] ********************************************
     [java] Port:

               ver is established!
               ut the command:

     [java] key:
file1
     [java] value:
```

Input key “file1”

Input the value

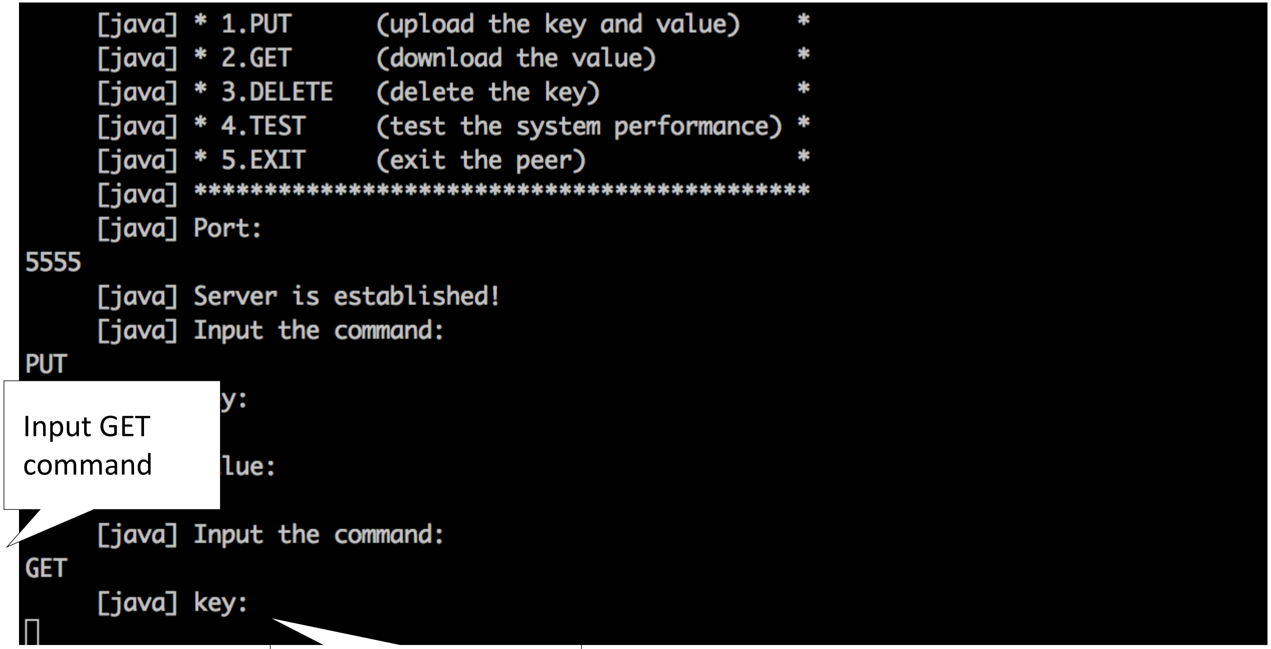


```
     [java] *          Peer Operation Command         *
     [java] *                                         *
     [java] * 1.PUT      (upload the key and value)    *
     [java] * 2.GET      (download the value)          *
     [java] * 3.DELETE   (delete the key)              *
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
     [java] ******************************************
     [java] Port:
5555
     [java] Server is established!
     [java] Input the command:
PUT
     [java] key:
file1
     [java] value:
CS550PA2
     [java] Input the command:
```

Input the value "CS550PA2"

PUT command is running successfully, there are no output to prove that the PUT is running successfully because we must test the cost time of these operations accurately. In the GET operation, we can prove that PUT operation is running successfully.

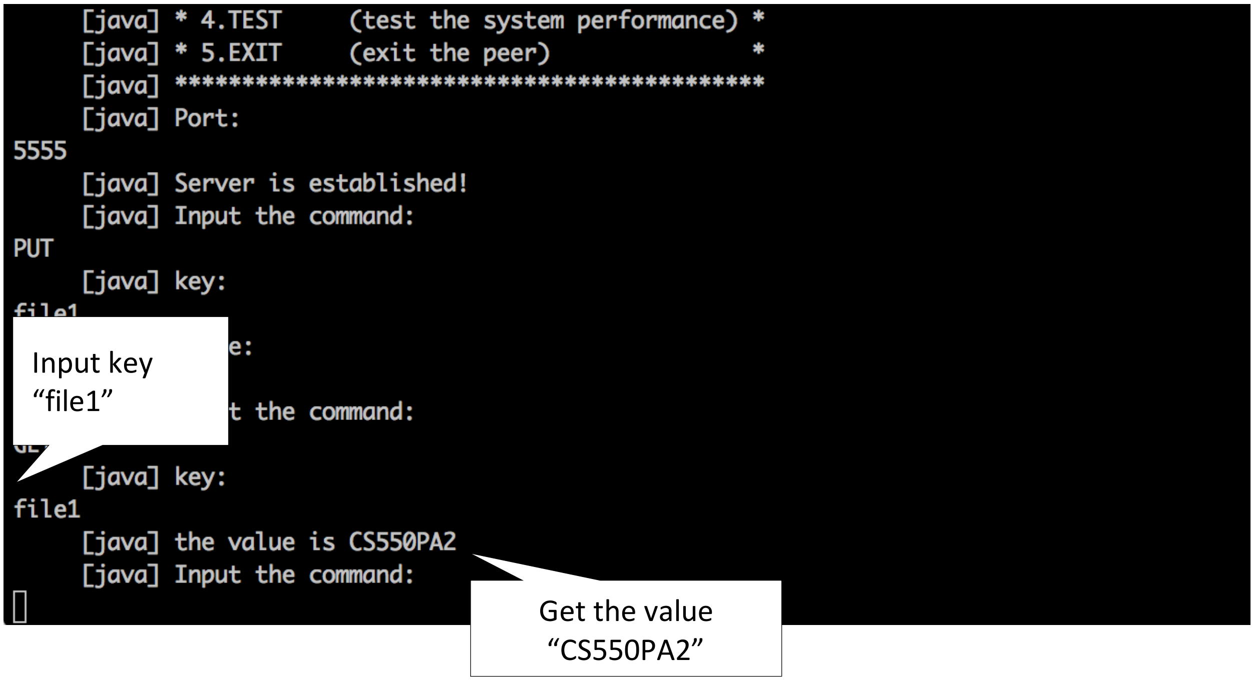
4.Input the GET command, we can get the following results.



```
     [java] * 1.PUT      (upload the key and value)    *
     [java] * 2.GET      (download the value)          *
     [java] * 3.DELETE   (delete the key)              *
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
     [java] ******************************************
     [java] Port:
5555
     [java] Server is established!
     [java] Input the command:
PUT
              y:

              lue:

     [java] Input the command:
GET
     [java] key:
```

Input GET command

Input the key



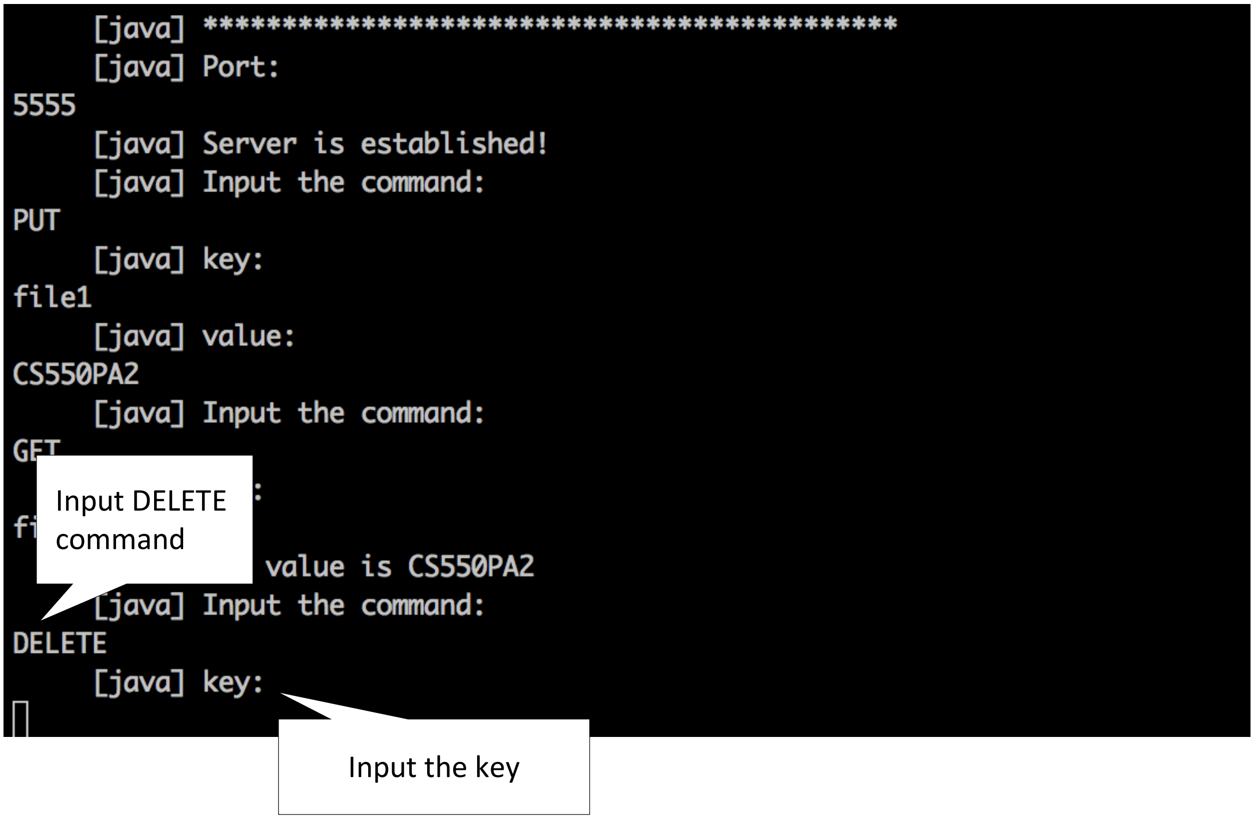
```
     [java] * 4.TEST     (test the system performance) *
     [java] * 5.EXIT     (exit the peer)               *
     [java] ********************************************
     [java] Port:
5555
     [java] Server is established!
     [java] Input the command:
PUT
     [java] key:
file1
                 e:

                 t the command:
GE
     [java] key:
file1
     [java] the value is CS550PA2
     [java] Input the command:
```

Input key “file1”

Get the value “CS550PA2”

we can get the value of the key “file1” which is the same value as we put into the distributed hash table. It can prove that the PUT and GET are successfully running.

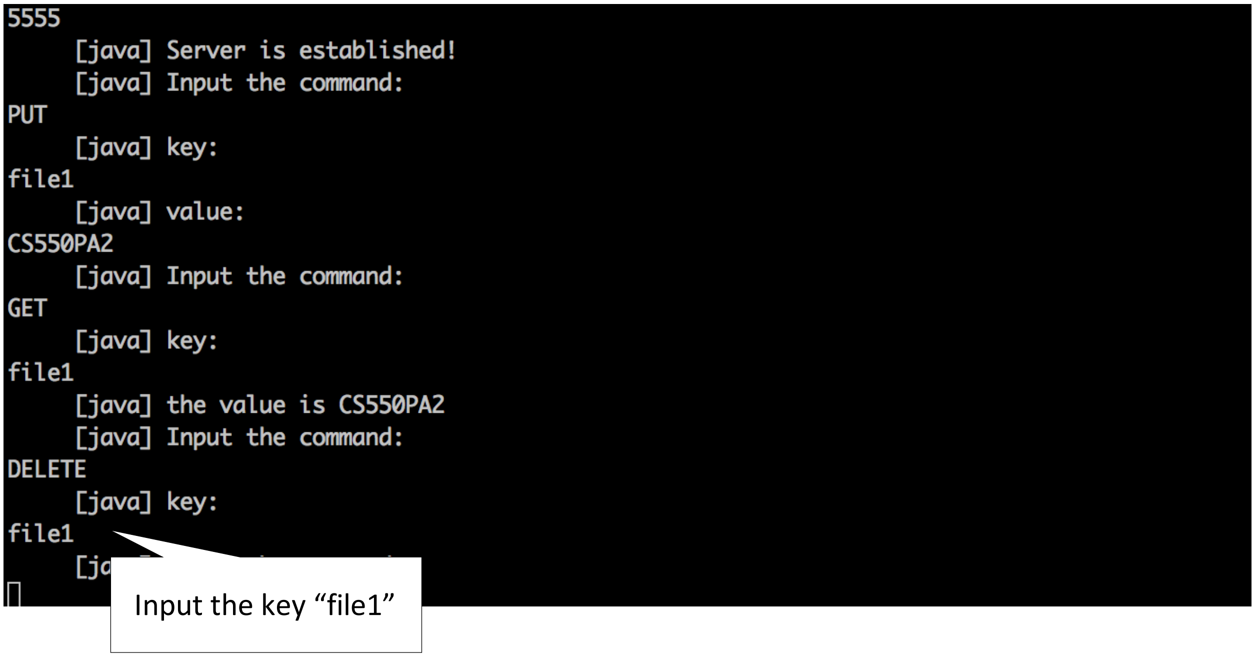
5.Input the DELETE command, we can get the following results.



```
     [java] ********************************************
     [java] Port:
5555
     [java] Server is established!
     [java] Input the command:
PUT
     [java] key:
file1
     [java] value:
CS550PA2
     [java] Input the command:
GET
                 :
fi
                value is CS550PA2
     [java] Input the command:
DELETE
     [java] key:
```

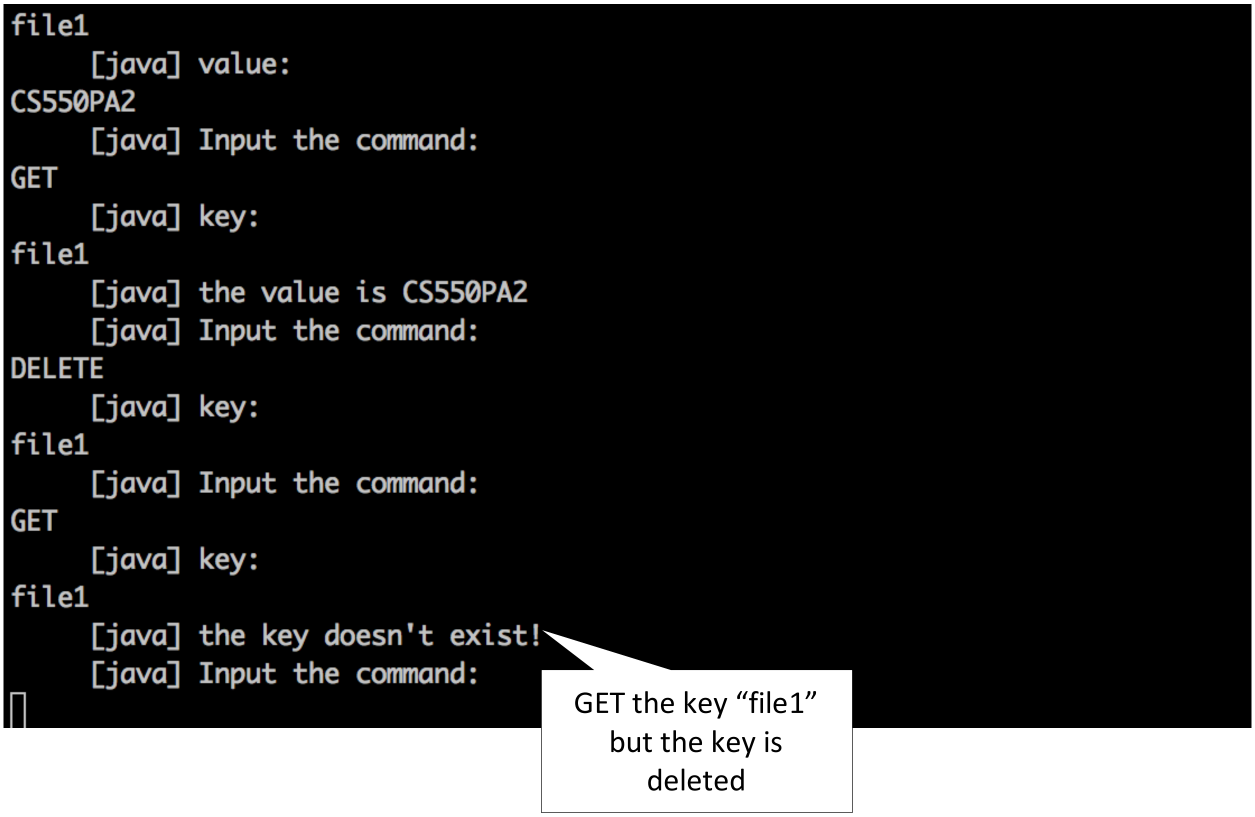
Input DELETE command

Input the key



```
5555
     [java] Server is established!
     [java] Input the command:
PUT
     [java] key:
file1
     [java] value:
CS550PA2
     [java] Input the command:
GET
     [java] key:
file1
     [java] the value is CS550PA2
     [java] Input the command:
DELETE
     [java] key:
file1
     [ja
```

Input the key “file1”

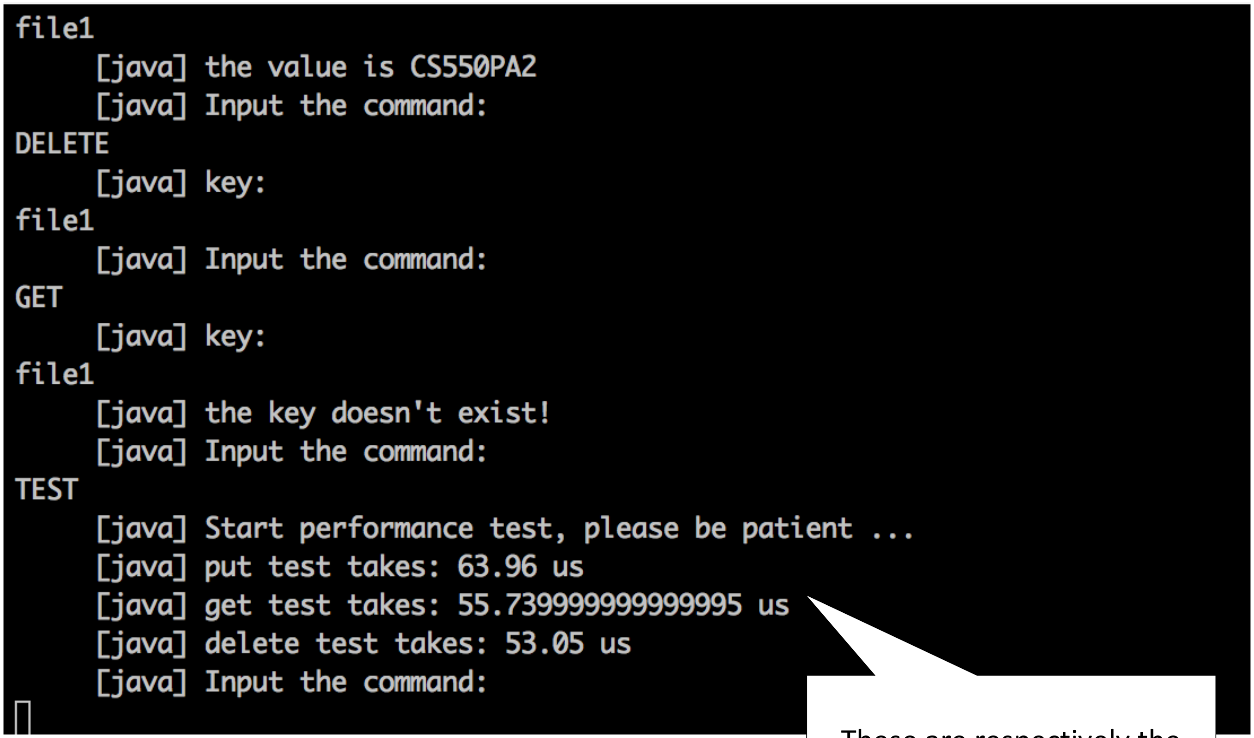


```
file1
     [java] value:
CS550PA2
     [java] Input the command:
GET
     [java] key:
file1
     [java] the value is CS550PA2
     [java] Input the command:
DELETE
     [java] key:
file1
     [java] Input the command:
GET
     [java] key:
file1
     [java] the key doesn't exist!
     [java] Input the command:
```

GET the key “file1” but the key is deleted

Through the result of GET file1, we can know the DELETE operation is running successfully.

6. Input the TEST command, we can get the following results.



```
file1
     [java] the value is CS550PA2
     [java] Input the command:
DELETE
     [java] key:
file1
     [java] Input the command:
GET
     [java] key:
file1
     [java] the key doesn't exist!
     [java] Input the command:
TEST
     [java] Start performance test, please be patient ...
     [java] put test takes: 63.96 us
     [java] get test takes: 55.739999999999995 us
     [java] delete test takes: 53.05 us
     [java] Input the command:
```

These are respectively the average cost time of 100k PUT commands, 100k GET commands and 100k DELETE commands