

PROJECT 1 REPORT

Ravi Rajpurohit - 1002079916
Sarthak Wadhawan - 1002028186

I have neither given nor received unauthorized assistance on this work. I will not post the project description and the solution online.

Sign: Ravi, Sarthak

Part - 1

File Server

The file server utilizes XML-RPC, a remote procedure call mechanism that employs the HTTP transport protocol. Through this, clients have the ability to issue requests to the server and receive corresponding responses.

This file server accommodates four primary operations:

1. Uploading
2. Downloading
3. Renaming
4. Deleting

To discover the list of supported operations for the file server, execute the following command:

```
...
```

```
python client.py -h
```

```
...
```

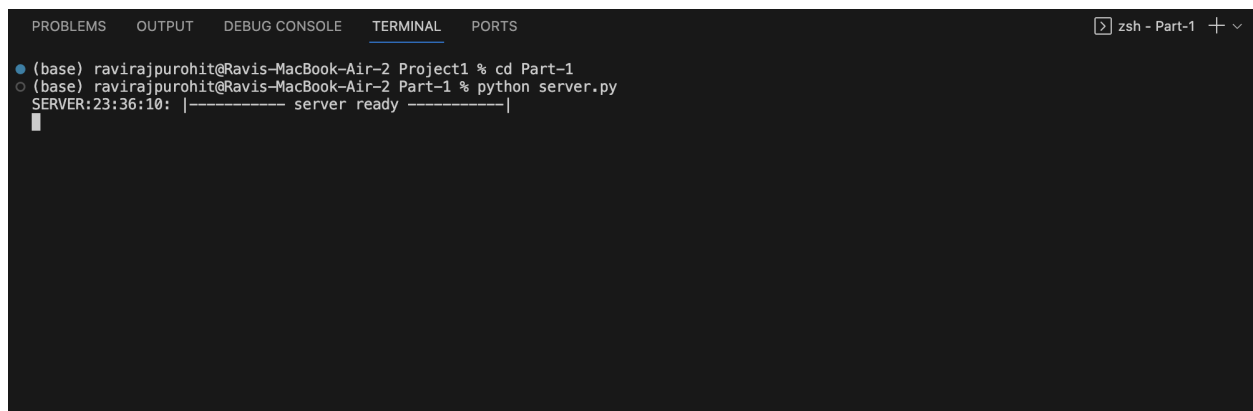
Run the Server

At the very beginning, we need to start the server

```
...
```

```
python server.py
```

```
...
```



The screenshot shows a terminal window with the following content:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS zsh - Part-1 + v
(base) ravirajpurohit@Ravis-MacBook-Air-2 Project1 % cd Part-1
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python server.py
SERVER:23:36:10: |----- server ready -----|
```

We can list the files present in the assigned client and server locations like following -

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

conda activate base
(base) ravirajpurohit@Ravis-MacBook-Air-2 Project1 % conda activate base
(base) ravirajpurohit@Ravis-MacBook-Air-2 Project1 % cd Part-1
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ls client
|----- Client:23:36:49: ['file2.txt', 'file3.txt', 'file1.txt'] -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ls server
|----- Client:23:36:56: [] -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %

```

Server Operations

The server can perform the following operations -

- Upload a file - a file can be uploaded using the following command. Note that the filenames and directory names are arbitrary.
 - `python client.py -up file1.txt server_files/file1.txt`
- Download a file - a file can be downloaded using the following command.
 - `python client.py -dow server_files/file1.txt file1.txt`
- Delete a file - a file can be deleted using -
 - `python client.py -del server_files/file1.txt`
- Rename a file - a file can be renamed using -
 - `python client.py -ren file1.txt file1renamed.txt`
- List files - we can always list the files on server side. After all these operations, we can list the files and see if the operation was successful in addition to the logs.

All the server operations are shown in the command prompt screenshot below along with the python code -

```

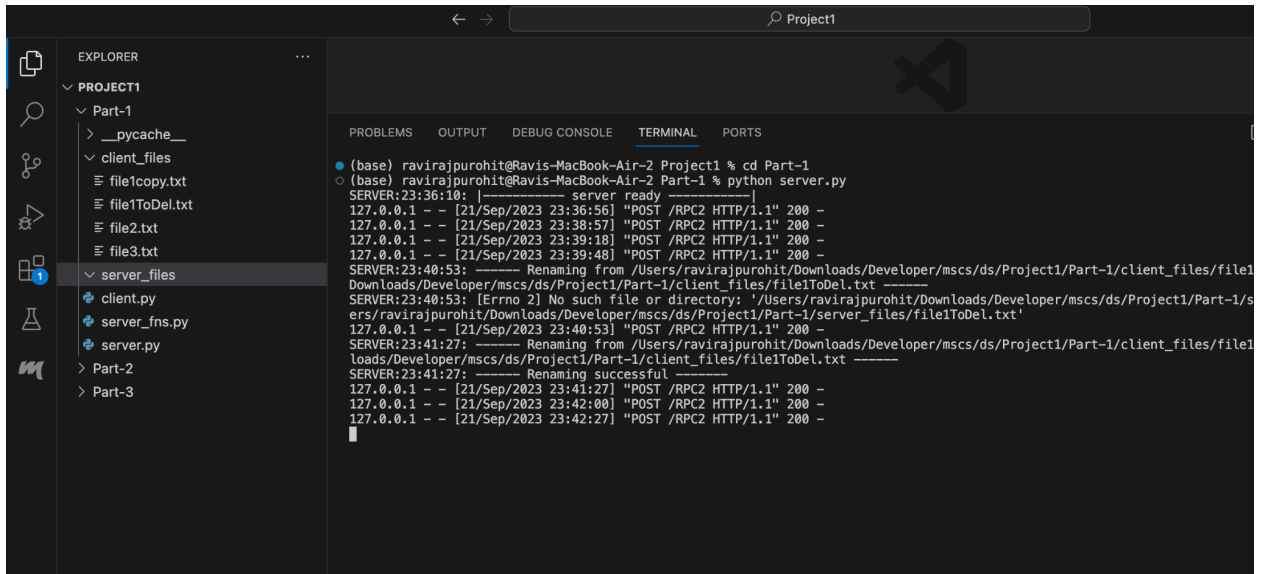
EXPLORER
PROJECT1
├── Part-1
│   ├── __pycache__
│   ├── client_files
│   │   ├── file1copy.txt
│   │   ├── file1ToDel.txt
│   │   ├── file2.txt
│   │   ├── file3.txt
│   └── server_files
│       ├── client.py
│       ├── server_fns.py
│       └── server.py
├── Part-2
└── Part-3

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -up file1.txt server_files/file1.txt
|----- Upload Successful -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ls server
|----- Client:23:39:18: ['file1.txt'] -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -dow server_files/file1.txt file1copy.txt
|----- Client:23:39:48: Download Successful -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ren file1copy.txt file1ToDel.txt
|----- Client:23:41:27: File Renaming Successful -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -del server_files/file1ToDel.txt
|----- Client:23:42:00: File Deletion Successful -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ls client
|----- Client:23:42:17: ['file2.txt', 'file3.txt', 'file1copy.txt', 'file1ToDel.txt'] -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python client.py -ls server
|----- Client:23:42:27: [] -----|
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 %

```

Following is the server side of the logs -



The screenshot shows a Visual Studio Code interface with a terminal window open. The Explorer panel on the left shows a project structure with folders 'Part-1' and 'Part-2', and files 'client.py', 'server_fns.py', and 'server.py'. The terminal window displays the following logs:

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Project1 % cd Part-1
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-1 % python server.py
SERVER:23:36:10: |----- server ready -----|
127.0.0.1 - - [21/Sep/2023 23:36:56] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2023 23:38:57] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2023 23:39:18] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2023 23:39:48] "POST /RPC2 HTTP/1.1" 200 -
SERVER:23:40:53: ----- Renaming from /Users/ravirajpurohit/Downloads/Developer/mscs/ds/Project1/Part-1/client_files/file1
Downloads/Developer/mscs/ds/Project1/Part-1/client_files/file1ToDel.txt -----
SERVER:23:40:53: [Errno 2] No such file or directory: '/Users/ravirajpurohit/Downloads/Developer/mscs/ds/Project1/Part-1/s
ers/ravirajpurohit/Downloads/Developer/mscs/ds/Project1/Part-1/server_files/file1ToDel.txt'
127.0.0.1 - - [21/Sep/2023 23:40:53] "POST /RPC2 HTTP/1.1" 200 -
SERVER:23:41:27: ----- Renaming from /Users/ravirajpurohit/Downloads/Developer/mscs/ds/Project1/Part-1/client_files/file1
loads/Developer/mscs/ds/Project1/Part-1/client_files/file1ToDel.txt -----
SERVER:23:41:27: ----- Renaming successful -----
127.0.0.1 - - [21/Sep/2023 23:41:27] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2023 23:42:00] "POST /RPC2 HTTP/1.1" 200 -
127.0.0.1 - - [21/Sep/2023 23:42:27] "POST /RPC2 HTTP/1.1" 200 -
```

Part 2

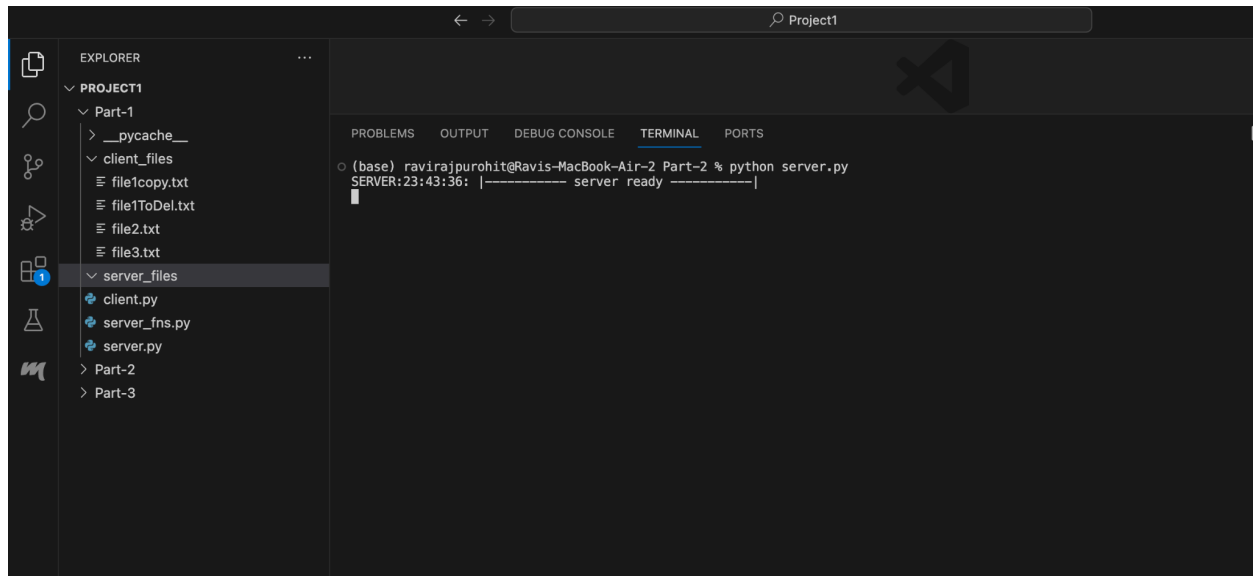
Synchronized Storage Service

Go to the Part-2 folder and run the server again as we did previously -

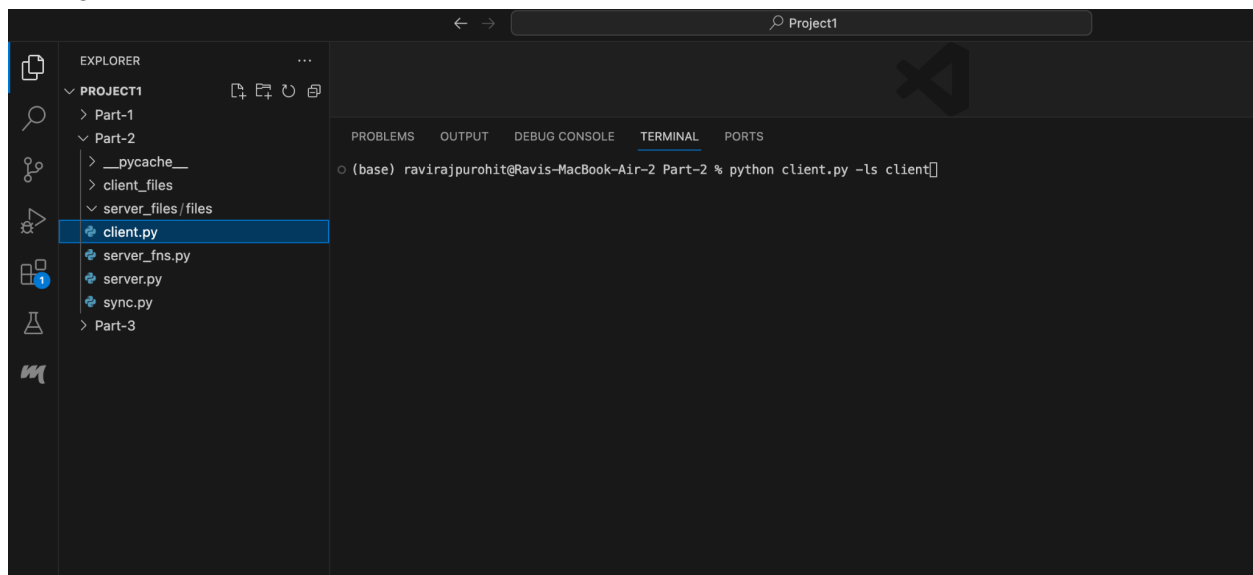
...

```
python server.py
```

...



To begin with, we can print the files present at the client and server sides.



To synchronize the files on both client and server sides, run the following command -

```
...
```

```
python sync.py
```

```
...
```

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-2 % python client.py -ls client
----- Client:23:44:41: ['files/file4.txt', 'files/file5.txt'] -----|
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-2 %
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-2 % python sync.py
-----
new file synced -----
modified file synced -----
modified file synced -----
```

While Sync.py is continuously running, it carries out three key operations:

- Files that exist in the client_directory but are absent in the server_directory are added to the server_directory.
- If a file in the client_directory is modified, the entire file is replaced with the updated version in the server_directory.
- When a file is deleted from the client_directory, it is likewise removed from the server_directory.

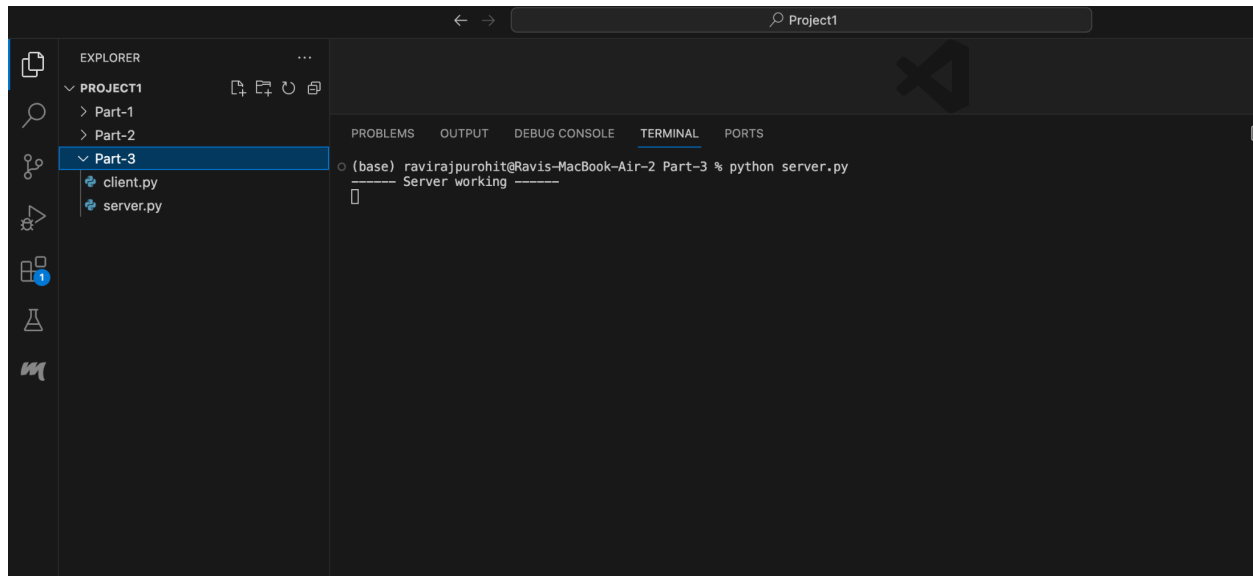
In case a file is deleted later, it is also synchronized -

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-2 % python sync.py
-----
new file synced -----
new file synced -----
modified file synced -----
modified file synced -----
modified file synced -----
modified file synced -----
new file synced -----
modified file synced -----
modified file synced -----
```

Part 3

Computation Server

Lets go to Part-3 folder and, you may have guessed, run server.py file like we did before.



The computation server provides support for two functions, available both synchronously and asynchronously:

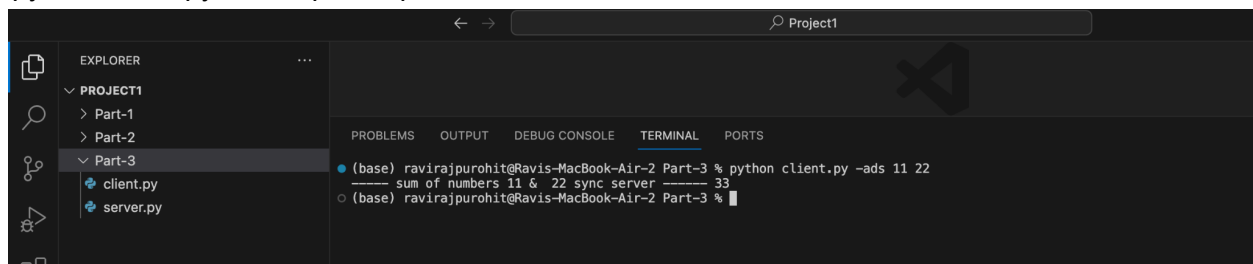
- Addition (Add)
- Sorting (Sort)

For synchronous "add" and "sort" operations, the server immediately sends the response to the client without any delay after the client's request.

In the case of asynchronous "add" and "sort" operations, the server does not immediately send the response to the client. Instead, it sends an acknowledgment along with an acknowledgment ID to the client. Subsequently, when the computation is completed, the client can retrieve the result using the provided acknowledgment ID.

Addition

Run this by the following command -
python client.py -ads input1 input2



For asynchronous addition (Add) operations, you can execute the following command to perform the operation synchronously:

```
...
python client -adas n1 n2
...
```

The screenshot shows a VS Code editor with a project named 'Project1'. The Explorer pane on the left shows the file structure: PROJECT1 > Part-1 > Part-2 > Part-3 > client.py and server.py. The client.py file is open in the editor, showing the following code:

```
1 import argparse
2 from xmlrpc.client import SimpleProxy
```

The TERMINAL pane shows the following output:

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -adas 11 22
----- request sent with id number ----- 91
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -ack1 91
----- sum of numbers from async server ----- 33
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 %
```

To retrieve the result using the acknowledgment number sent from the server, run the following command:

```
...
python client -ack1 acknowledgmentID
...
```

The screenshot shows a VS Code editor with a project named 'Project1'. The Explorer pane on the left shows the file structure: PROJECT1 > Part-1 > Part-2 > Part-3 > client.py and server.py. The client.py file is open in the editor, showing the following code:

```
1 import argparse
2 from xmlrpc.client import SimpleProxy
```

The TERMINAL pane shows the following output:

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -adas 11 22
----- request sent with id number ----- 91
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -ack1 91
----- sum of numbers from async server ----- 33
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 %
```

This allows you to obtain the calculated sum using the provided acknowledgment number.

Sort

Synchronous Sort can be performed like -

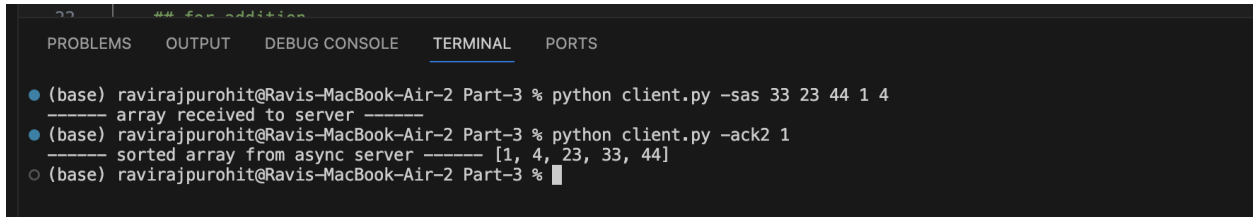
The screenshot shows a VS Code editor with a project named 'Project1'. The Explorer pane on the left shows the file structure: PROJECT1 > Part-1 > Part-2 > Part-3 > client.py and server.py. The client.py file is open in the editor, showing the following code:

```
1 import argparse
2 from xmlrpc.client import SimpleProxy
```

The TERMINAL pane shows the following output:

```
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -ss 33 23 44 1 4
----- sorted array from sync server ----- [1, 4, 23, 33, 44]
(base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 %
```


For Asynchronous sort, the server gives out an acknowledgement, and that can be used to retrieve a sorted output like shown below.



```
22 | ## for addition
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -sas 33 23 44 1 4
----- array received to server -----
● (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 % python client.py -ack2 1
----- sorted array from async server ----- [1, 4, 23, 33, 44]
○ (base) ravirajpurohit@Ravis-MacBook-Air-2 Part-3 %
```