
LABORATORY-2
SOCKET PROGRAMMING

Q1) Infix Evaluation

#server.py

```
import socket as sk
server = sk.socket(sk.AF_INET, sk.SOCK_STREAM)
host = "localhost"
port = 1262
server.bind((host,port))
server.listen(3)
conn, addr = server.accept()
infix = conn.recv(1024).decode()
infix = eval(infix)
conn.send(str(infix).encode())
conn.close()
server.close()
```

#client.py

```
import socket as sk
conn = sk.socket(sk.AF_INET, sk.SOCK_STREAM)
host = "localhost"
port = 1262
conn.connect((host,port))
infix = input("Enter infix expression : ")
conn.send(infix.encode())
res = conn.recv(1024).decode()
print(f"answer = {res}")
conn.close()
```

I/O:

```
Enter infix expression : 123 + 89*5
answer = 568
```

Q2) Implement a whatsapp like personal chat system

#server.py

```
import socket as sk
import threading
port = 1234
host = ""
clients = []
def broadcast(c1,c2,addr):
    while 1:
        ms = c1.recv(1024).decode()
        msg = f"{addr} : {ms}"
        c2.send(msg.encode())
def continue_conn(c1,c2,addr1,addr2) :
    t1  = threading.Thread(target=broadcast, args=(c1,c2,addr1))
    t2  = threading.Thread(target=broadcast, args=(c2,c1,addr2))
    t1.start()
    t2.start()
server = sk.socket(sk.AF_INET, sk.SOCK_STREAM)
server.bind((host,port))
server.listen(5)
c1, addr1 = server.accept()
c2, addr2 = server.accept()
continue_conn(c1,c2,addr1,addr2)
```

#client.py

```
import socket
import threading
host = "localhost"
port = 1234
def write(c,p):
    while 1:
        msg = input()
        c.send(msg.encode())
def read(c,p):
    while 1:
        print(c.recv(1024).decode())
conn = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
conn.connect((host,port))
t1  = threading.Thread(target=write, args=(conn,1))
t2  = threading.Thread(target=read, args=(conn,1))
t1.start()
t2.start()
```

I/O:

```
heyy
('127.0.0.1', 57137) : hello there
wyd??
('127.0.0.1', 57137) : coding...
```

```
('127.0.0.1', 57136) : heyy
hello there
('127.0.0.1', 57136) : wyd??
coding...
```

Q3) Tic-Tac-Toe game

#server.py

```
import socket as sk
import random
server = sk.socket(sk.AF_INET,sk.SOCK_STREAM)
port = 8080
host = "localhost"
server.bind((host,port))
server.listen(4)
def getcoord():
    xr = random.randint(0,2)
    yr = random.randint(0,2)
    return xr,yr
def check(arr):
    for i in range(3):
        if(arr[i][0] == arr[i][1] and arr[i][1] == arr[i][2] and arr[i][0] !=
0):
            return arr[i][0]
    for i in range(3):
        if(arr[0][i] == arr[1][i] and arr[1][i] == arr[2][i] and arr[0][i] !=
0):
            return arr[0][i]
    if(arr[0][0] == arr[1][1] and arr[1][1] == arr[2][2] and arr[1][1] != 0):
        return arr[1][1]
    if(arr[0][2] == arr[1][1] and arr[1][1] == arr[2][0] and arr[1][1] != 0):
        return arr[1][1]
    return 0
while 1:
    arr = [[0,0,0],[0,0,0],[0,0,0]]
    conn, addr = server.accept()
    while 1:
        coord = int(conn.recv(1024).decode())
        i = int(coord/10)-1
        j = int(coord%10)-1
        if(arr[i][j] != 0):
            conn.send(str(400).encode())
        else:
            arr[i][j] = 1
            while 1:
                ti,tj = getcoord()
                if(arr[ti][tj] == 0):
                    arr[ti][tj] = 2
                    break
            res = check(arr)
            if(res == 1):
```

```
                conn.send(str(100).encode())
                break
            if(res == 2):
                conn.send(str(200).encode())
                break
            if(res == 0):
                conn.send(str(500).encode())
            print(arr)
    print(arr)
    print("match over")
    conn.close()
server.close()
```

#client.py

```
import socket as sk
conn = sk.socket(sk.AF_INET,sk.SOCK_STREAM)
port = 8080
host = "localhost"
conn.connect((host,port))
while 1:
    print("Enter coordinate to mark X : ")
    n = int(input())
    conn.send(str(n).encode())
    n = int(conn.recv(1024).decode())
    if(n == 400):
        print("Given input is wrong try again")
    elif(n == 100):
        print("you win")
        break
    elif(n == 200):
        print("computer win")
        break
conn.close()
```

I/O: