SnakeServer.py

```python
import socket
import threading
import time

# List to store connected client sockets
client_sockets = []

IP, PORT = '192.168.20.69', 9999

def handle_client(client_socket):
    global client_sockets

    try:
        while True:
            data = client_socket.recv(1024)
            if not data:
                break

            # Process the received data
            processed_data = f'P{client_sockets.index(client_socket)}|{data.decode("utf-8")}'
            sendable_data = processed_data.encode('utf-8')
            # Send the processed data back to the client(s)
            client_socket.send(sendable_data)

            # Send data to all connected clients
            print(f'Player {client_sockets.index(client_socket)}: {processed_data}')
            send_to_all(sendable_data, client_socket)

    except Exception as e:
        print(f'Error: {e}')

    finally:
        # Remove the client socket from the list when done
        client_sockets.remove(client_socket)
        client_socket.close()

def send_to_all(message, sender_socket):
    global client_sockets

    # Iterate through the list of connected clients and send data to each one
    for client_socket in client_sockets:
        # Avoid sending the data back to the sender
        if client_socket != sender_socket:
            try:
                client_socket.send(message)
            except Exception as e:
                print(f'Error sending data to a client: {e}')
                # Remove the client socket from the list if an error occurs
                client_sockets.remove(client_socket)
                client_socket.close()

def start_server():
    global client_sockets

    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((IP, PORT))
```

```
    server_socket.listen(5)
    print(f'[*] Listening on {IP}:{PORT}')

    try:
        while True:
            client_socket, addr = server_socket.accept()
            print(f'[*] Accepted connection from {addr[0]}:{addr[1]}')

            # Add the new client socket to the list
            client_sockets.append(client_socket)
            client_socket.send(f'you|P{client_sockets.index(client_socket)}'.encode('utf-8'
))
            if len(client_sockets) == 2:
                time.sleep(0.5)
                client_socket.send('start|info'.encode('utf-8'))
                send_to_all('start|info'.encode('utf-8'), client_socket)
                update_handler = threading.Thread(target=update_loop, args=())
                update_handler.start()
            client_handler = threading.Thread(target=handle_client, args=(client_socket,))
            client_handler.start()

    except KeyboardInterrupt:
        print('[*] Server shutting down.')

        # Close all client sockets before shutting down the server
        for client_socket in client_sockets:
            client_socket.close()

        server_socket.close()

def update_loop():
    while True:
        send_to_all('update|info'.encode('utf-8'),None)
        time.sleep(0.1)

if __name__ == '__main__':
    start_server()

```