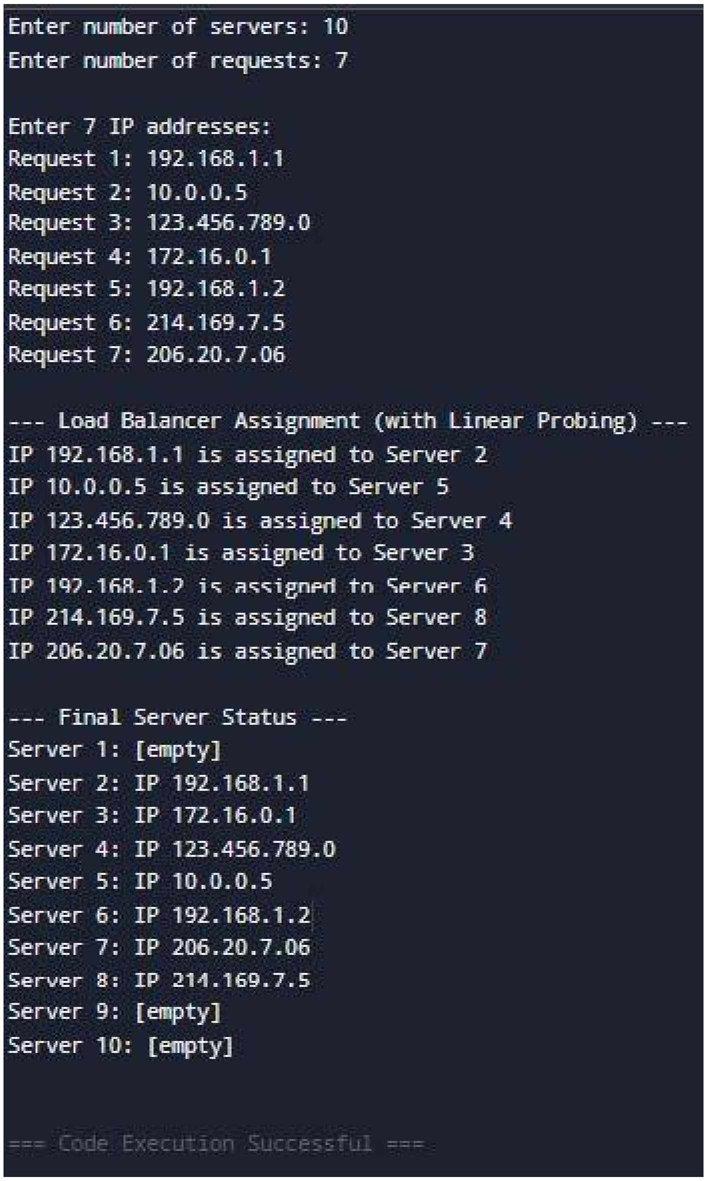
ASSIGNMENT 2

IMPLEMNATION OF HASHING (LOAD BALANCING)

PROGRAM :-

```
#include <iostream>
#include <string>
using namespace std;
int hashIP(const string& ip) {
int hash = 0;
for (char c : ip) {
hash += (int)c;
}
return hash;
}
int main() {
int s, r;
cout<<"Enter number of servers: ";
cin>>s;
cout<<"Enter number of requests: ";
cin>>r;
if (r > s) {
cout << "Error: Number of requests cannot be more than number of servers.\n";
return 0;
}
string* requests = new string[r];
string* servers = new string[s];
for (int i = 0; i < s; i++) {
servers[i] = "";
}
cout << "\nEnter " << r << " IP addresses:\n";
for (int i = 0; i < r; i++) {
cout << "Request " << i + 1 << ": ";
cin >> requests[i];
}
cout << "\n--- Load Balancer Assignment (with Linear Probing) ---\n";
for (int i = 0; i < r; i++) {
string ip = requests[i];
int hash_val = hashIP(ip);
int serverID = hash_val % s;
if (servers[serverID] == "") {
servers[serverID] = ip;
} else {
int new_pos = (serverID + 1) % s;
while (servers[new_pos] != "" && new_pos != serverID) {
new_pos = (new_pos + 1) % s;
}
servers[new_pos] = ip;
serverID = new_pos;
}
cout << "IP " << ip
<< " is assigned to Server " << serverID + 1 << "\n";
}
cout << "\n--- Final Server Status ---\n";
for (int i = 0; i < s; i++) {
if (servers[i] == "")
cout << "Server " << i + 1 << ": [empty]\n";
else
cout << "Server " << i + 1 << ": IP " << servers[i] << "\n";
}
delete[] requests;
delete[] servers;
return 0;
}
```

OUTPUT :-



```
Enter number of servers: 10
Enter number of requests: 7

Enter 7 IP addresses:
Request 1: 192.168.1.1
Request 2: 10.0.0.5
Request 3: 123.456.789.0
Request 4: 172.16.0.1
Request 5: 192.168.1.2
Request 6: 214.169.7.5
Request 7: 206.20.7.06

--- Load Balancer Assignment (with Linear Probing) ---
IP 192.168.1.1 is assigned to Server 2
IP 10.0.0.5 is assigned to Server 5
IP 123.456.789.0 is assigned to Server 4
IP 172.16.0.1 is assigned to Server 3
IP 192.168.1.2 is assigned to Server 6
IP 214.169.7.5 is assigned to Server 8
IP 206.20.7.06 is assigned to Server 7

--- Final Server Status ---
Server 1: [empty]
Server 2: IP 192.168.1.1
Server 3: IP 172.16.0.1
Server 4: IP 123.456.789.0
Server 5: IP 10.0.0.5
Server 6: IP 192.168.1.2
Server 7: IP 206.20.7.06
Server 8: IP 214.169.7.5
Server 9: [empty]
Server 10: [empty]


=== Code Execution Successful ===
```