Ex.No: 4	Simulation of DNS Using UDP Sockets.	Date: 23/07/2025

Aim:

To simulate the Domain Name System (DNS) service using UDP sockets in Java, where a client sends a domain name to the server and the server responds with the corresponding IP address.

Algorithm:

1. **Understanding DNS and UDP:**
    - DNS translates domain names into IP addresses.
    - UDP (User Datagram Protocol) is a connectionless protocol used for sending short messages (datagrams).
2. **Setup:**
    - Create two Java programs: a DNS Server and a DNS Client.
    - The DNS Server maintains a list (mapping) of domain names and their IP addresses.
    - The DNS Client sends a domain name request to the server using UDP.
    - The server looks up the IP address and sends it back to the client.
3. **Server-side steps:**
    - Create a UDP socket and bind it to a port.
    - Wait for incoming datagrams from clients.
    - On receiving a domain name, search the IP address from the predefined dictionary.
    - Send the IP address back to the client as a UDP datagram.
4. **Client-side steps:**
    - Create a UDP socket.
    - Send a datagram containing the domain name to the server.
    - Wait to receive the server's response containing the IP address.
    - Display the IP address to the user.
5. **Execution:**
    - Run the DNS server program.
    - Run the DNS client program.
    - Input domain names on the client and observe the responses.

Program:

DNS Server (UDP)

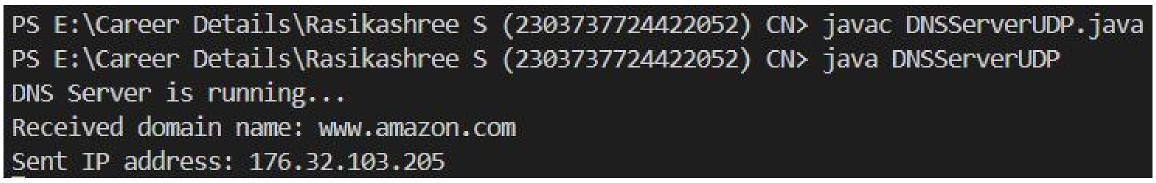
```
import java.net.*;
import java.util.HashMap;

public class DNSServerUDP {
    public static void main(String[] args) {
        try {
            DatagramSocket serverSocket = new DatagramSocket(9876);
            byte[] receiveData = new byte[1024];
            byte[] sendData;

            // DNS records: domain name to IP address mapping
            HashMap<String, String> dnsRecords = new HashMap<>();
            dnsRecords.put("www.google.com", "172.217.16.196");
            dnsRecords.put("www.facebook.com", "157.240.22.35");
            dnsRecords.put("www.yahoo.com", "98.137.11.163");
            dnsRecords.put("www.amazon.com", "176.32.103.205");

            System.out.println("DNS Server is running...");

            while (true) {
                // Receive packet from client
                DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
                serverSocket.receive(receivePacket);

                String domainName = new String(receivePacket.getData(), 0, receivePacket.getLength());
                System.out.println("Received domain name: " + domainName);

                // Lookup IP address
                String ipAddress = dnsRecords.getOrDefault(domainName, "Domain not found");

                // Send IP address back to client
                sendData = ipAddress.getBytes();
                InetAddress clientAddress = receivePacket.getAddress();
                int clientPort = receivePacket.getPort();

                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
                clientAddress, clientPort);
                serverSocket.send(sendPacket);

                System.out.println("Sent IP address: " + ipAddress);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

DNS Client (UDP)

```
import java.net.*;
import java.util.Scanner;

public class DNSClientUDP {
   public static void main(String[] args) {
      try {
         DatagramSocket clientSocket = new DatagramSocket();
         InetAddress IPAddress = InetAddress.getByName("localhost"); // Server is running locally

         Scanner scanner = new Scanner(System.in);
         byte[] sendData;
         byte[] receiveData = new byte[1024];

         System.out.print("Enter domain name to resolve: ");
         String domainName = scanner.nextLine();

         sendData = domainName.getBytes();

         // Send domain name to DNS server
         DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress,
         9876);
         clientSocket.send(sendPacket);

         // Receive IP address from server
         DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
         clientSocket.receive(receivePacket);

         String ipAddress = new String(receivePacket.getData(), 0, receivePacket.getLength());
         System.out.println("IP Address for " + domainName + ": " + ipAddress);

         clientSocket.close();
         scanner.close();
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

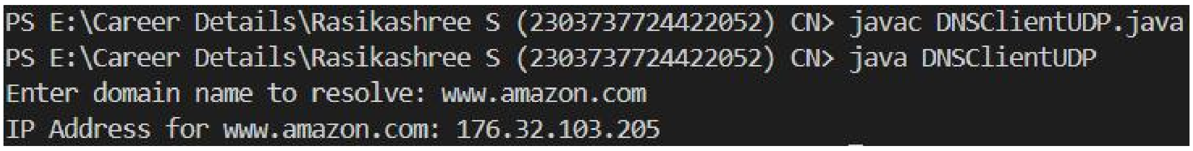
<u>Output:</u>

DNS Server



```
PS E:\Career Details\Rasikashree S (2303737724422052) CN> javac DNSServerUDP.java
PS E:\Career Details\Rasikashree S (2303737724422052) CN> java DNSServerUDP
DNS Server is running...
Received domain name: www.amazon.com
Sent IP address: 176.32.103.205
```

DNS Client



```
PS E:\Career Details\Rasikashree S (2303737724422052) CN> javac DNSClientUDP.java
PS E:\Career Details\Rasikashree S (2303737724422052) CN> java DNSClientUDP
Enter domain name to resolve: www.amazon.com
IP Address for www.amazon.com: 176.32.103.205
```

Particulars	Marks Allotted	Marks Awarded
Program / Simulation	40	
Program Execution	30	
Result	20	
Viva Voce	10	
Total	100	

Result:

The DNS simulation using UDP sockets is successfully implemented. The client sends a domain name to the server, and the server returns the corresponding IP address using UDP communication. This mimics the DNS query process over a network.