|  |  |
| --- | --- |
| Semester | T.E. Semester V – Computer Engineering |
| Subject | Computer Network |
| Subject Professor In-charge | Prof. Amit K. Nerurkar |
| Assisting Teachers | Prof. Amit K. Nerurkar |
| Laboratory | M-313-A |

|  |  |
| --- | --- |
| Student Name | Deep Salunkhe |
| Roll Number | 21102A0014 |
| TE Division | A |

**Title : IP Addressing**

**Theory:**

IP (Internet Protocol) addresses are numerical labels assigned to devices on a computer network that uses the Internet Protocol for communication. These addresses serve two primary functions:

1. **Host Identification**: IP addresses uniquely identify devices (hosts) on a network. Each device connected to a network, whether it's a computer, smartphone, or server, is assigned a distinct IP address.
2. **Location Addressing**: IP addresses also provide information about the device's location on the network. This allows routers and switches to determine how to forward data packets from the source to the destination.

**IP Address Classes**

IPv4 addresses are divided into different classes, each with its own range and purpose. The classes are:

1. **Class A**: IP addresses in the range 1.0.0.0 to 126.0.0.0 are designated as Class A addresses. They are typically used by large organizations and corporations.
2. **Class B**: IP addresses in the range 128.0.0.0 to 191.255.0.0 belong to Class B. These addresses are often assigned to medium-sized networks.
3. **Class C**: IP addresses in the range 192.0.0.0 to 223.255.255.0 fall into Class C. These addresses are commonly used for smaller networks.
4. **Class D**: Class D addresses (224.0.0.0 to 239.255.255.255) are reserved for multicast groups, where data is sent to multiple recipients simultaneously.
5. **Class E**: Class E addresses (240.0.0.0 to 255.255.255.255) are reserved for experimental and research purposes.

**Validating IP Addresses**

In your lab, you are tasked with creating a program to validate IP addresses. Valid IP addresses must adhere to the following rules:

* The address must consist of four numerical segments separated by periods (e.g., "192.168.0.1").
* Each segment must be between 0 and 255.
* Leading zeros in each segment are not allowed.
* The first segment must not be 0.
* The last segment must not be 0.

**Determining IP Address Class**

In addition to validation, your program will determine the class of the IP address based on its first segment:

* Class A: 1 to 126
* Class B: 128 to 191
* Class C: 192 to 223
* Class D: 224 to 239
* Class E: 240 to 255

**Implementation**:

#include <iostream>

#include <bits/stdc++.h>

using namespace std;

void takeip(vector<int> &ip)

{

    cout << "enter ip without . :";

    int n;

    cout << "enter the number of group's':";

    cin >> n;

    cout << "start entering the input's......." << endl;

    for (int i = 0; i < n; i++)

    {

        int t;

        cin >> t;

        ip.push\_back(t);

    }

}

bool isvalid(vector<int> &ip)

{

*// size*

    int sz = ip.size();

    if (sz != 4)

    {

        return false;

    }

*// every group should [0,255]*

    for (int i = 0; i < 4; i++)

    {

        int value = ip[i];

        if (!(value >= 0 && value < 256))

        {

            return false;

        }

    }

*// hex and binary*

    return true;

}

int findgroup(vector<int> &ip)

{

    if (ip[0] >= 0 && ip[0] <= 127)

    {

        cout << "Belong to group A" << endl;

        return 0;

    }

    else if (ip[0] >= 128 && ip[0] <= 191)

    {

        cout << "Belong to group B" << endl;

        return 1;

    }

    else if (ip[0] >= 192 && ip[0] <= 223)

    {

        cout << "Belong to group C" << endl;

        return 2;

    }

    else if (ip[0] >= 224 && ip[0] <= 239)

    {

        cout << "Belong to group D" << endl;

        return 3;

    }

    else if (ip[0] >= 240 && ip[0] <= 255)

    {

        cout << "Belong to group E" << endl;

        return 4;

    }

    return 0;

}

void firstip(vector<int> &ip, int gid, vector<vector<int>> &cm)

{

    vector<int> fip;

    cout << "first ip is:" << endl;

    for (int i = 0; i < 4; i++)

    {

        int temp = ip[i] & cm[gid][i];

        fip.push\_back(temp);

        cout << fip[i] << ".";

    }

}

void lastip(vector<int> &ip, int gid, vector<vector<int>> &cm)

{

    vector<int> lip;

    cout << "last ip is:" << endl;

    for (int i = 0; i < 4; i++)

    {

        int ones;

        if (cm[gid][i] == 255)

            ones = 0;

        else

            ones = 255;

        int temp = ip[i] | ones;

        lip.push\_back(temp);

        cout << lip[i] << ".";

    }

}

int main()

{

    vector<int> ip;

    takeip(ip);

    vector<vector<int>> cm = {{255, 0, 0, 0}, {255, 255, 0, 0}, {255, 255, 255, 0}};

    if (isvalid(ip))

    {

        cout << "ip is valid......" << endl;

*// finding to which group it belong's*

        int gid = findgroup(ip);

        if ((gid == 3 || gid == 4))

        {

            cout << "this is reversed class......" << endl;

        }

        else

        {

            firstip(ip, gid, cm);

            lastip(ip, gid, cm);

        }

    }

    else

    {

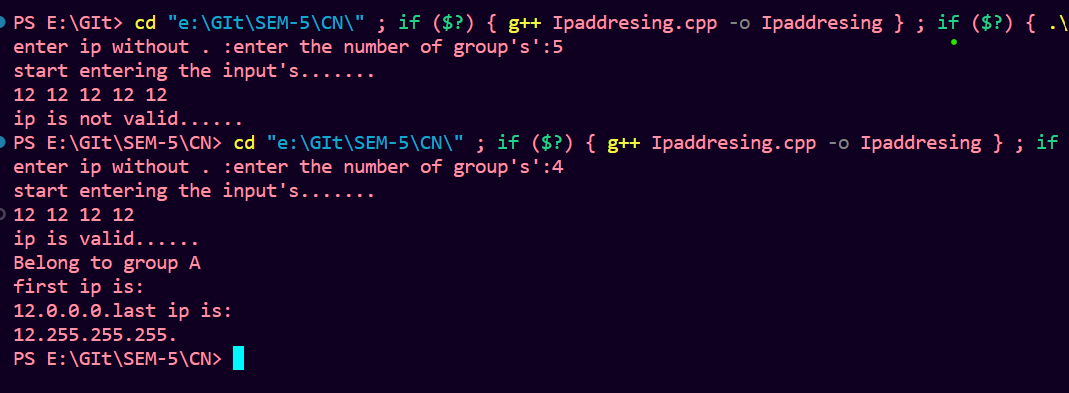
        cout << "ip is not valid......" << endl;

    }

    return 0;

}

**Output:**

****

Top of Form