// ACN_PRACTICAL_2 =>

```
// Given an IPv4 address string implement a program to check & print for the following:
// 1. Identify the class of IPv4 address in binary notation.
// 2. Generate the network address, net Id and Host id for classful IPv4 address scheme using default address value.
// 3. Generate the 1st address, last address and total no. of addresses for classless IPv4 addressing scheme.
// 4. Calculation of custom masking value for classless IPv4 addresses.

#include<bits/stdc++.h>
using namespace std;

void customMasking(){
    int n;
    cout << "Enter the value for custom masking: ";
    cin >> n;

    if( n> 0 && n<=32){
        if(n>=24){
            int res = 0;
            int i = n - 24;
            for(int j = 0; j<=i; j++){
                res = res + pow(2, 7 - i);
            }
            cout<< "Custom Masking: 255.255.255."<<res;
        }
        else if(n>=16){
            int res = 0;
            int i = n - 16;
            for(int j = 0; j<=i; j++){
                res = res + pow(2, 7 - j);
            }
            cout<< "Custom Masking: 255.255."<<res<<".0";
        }
        else if(n>=8){
            int res = 0;
            int i = n - 16;
            for(int j = 0; j<=i; j++){
                res = res + pow(2, 7 - j);
            }
            cout<< "Custom Masking: 255."<<res<<".0.0";
        }
        else if(n>0){
            int res = 0;
            int i = n - 16;
            for(int j = 0; j<=i; j++){
                res = res + pow(2, 7 - j);
            }
            cout<<"Custom Masking: "<<res<<"0.0.0";
        } else{
            cout <<"Invalid Value";
        }
    }
    cout<<endl;
}

int main(){
    string ip;
    cout << "Enter the IP address: ";
    cin >> ip;
    int n = ip.length();
    string arr[6];

    int j = 0;
```

```
for(int i = 0; i<n; i++)
    if(ip[i]!='.')
        arr[j] = arr[j] + ip[i];
    else
        j++;

bitset<8> arrBits[4];
for(int i = 0; i<4; i++){
    bitset<8> binaryRepresentation(stoi(arr[i]));
    arrBits[i] = binaryRepresentation;
}

cout << "Binary representation of the IP address: ";
for(int i = 0; i<4; i++){
    cout << arrBits[i] << ".";
} cout << endl;
try{
    if(stoi(arr[0]) > 0 && stoi(arr[0])<127)
        cout<<"Class A"<<endl;
    if(stoi(arr[0]) > 127  && stoi(arr[0])<191)
        cout<<"Class B"<<endl;
    if(stoi(arr[0]) > 191  && stoi(arr[0])<223)
        cout<<"Class C"<<endl;
    if(stoi(arr[0]) > 223  && stoi(arr[0])<239)
        cout<<"Class D"<<endl;
    if(stoi(arr[0]) > 240  && stoi(arr[0])<255)
        cout<<"Class E"<<endl;
    } catch(exception e){
        cout<<"Undefined Class"<<endl;
}

if (stoi(arr[0]) >= 1 && stoi(arr[0]) <= 126) {
    cout << "Network address :" << stoi(arr[0]) << ".0.0.0" << endl;
    cout << "Net id: " << stoi(arr[0]) << endl;
    cout << "Host address : 0." << arr[1] << "." << arr[2] << "." << arr[3] << endl;
    cout << "Host ID : " << arr[1] << "." << arr[2] << "." << arr[3] << endl;
    cout << "First address : " << arr[0] << ".0.0.0" << endl;
    cout << "Last address : " << arr[0] << "." << "255.255.255" << endl;
    cout << "Total number of address : " << 255 * 255 * 255 << endl;
}

if (stoi(arr[0]) >= 128 && stoi(arr[0]) <= 191) {
    cout << "Network address : " << arr[0] << "." << arr[1] << ".0.0" << endl;
    cout << "Net id : " << arr[0] << "." << arr[1] << endl;
    cout << "Host address : 0.0" << "." << arr[2] << "." << arr[3] << endl;
    cout << "Host ID : " << arr[2] << "." << arr[3] << endl;
    cout << "First address : " << arr[0] << "." << arr[1] << ".0.0" << endl;
    cout << "Last address : " << arr[0] << "." << arr[1] << "." << "255.255" << endl;
    cout << "Total number of address : " << 255 * 255 << endl;
}

 if (stoi(arr[0]) >= 192 && stoi(arr[0]) <= 223) {
    cout << "Network address : " << arr[0] << "." << arr[1] << "." << arr[2] << "." << "0" << endl;
    cout << "Net id : " << arr[0] << "." << arr[1] << "." << arr[2] << endl;
    cout << "Host address : 0.0.0" << "." << arr[3] << endl;
    cout << "Host ID : " << arr[3] << endl;
    cout << "First address : " << arr[0] << "." << arr[1] << "." << arr[2] << "." << "0" << endl;
    cout << "Last address : " << arr[0] << "." << arr[1] << "." << arr[2] << "." << "255" << endl;
    cout << "Total number of address : " << 255 << endl;
}

if (stoi(arr[0]) >= 224 && stoi(arr[0]) <= 239) {
```

```
        cout << "No network address available for class D" << endl;
        cout << "No net ID available" << endl;
        cout << "No host address available for class D" << endl;
        cout << "No host ID available" << endl;
        cout << "No first address and last address available" << endl;
    }

    if (stoi(arr[0]) >= 240 && stoi(arr[0]) <= 255) {
        cout << "No network address available for class E" << endl;
        cout << "No net ID available" << endl;
        cout << "No host address available for class E" << endl;
        cout << "No host ID available" << endl;
        cout << "No first address and last address available" << endl;
    }
    customMasking();

    return 0;
}
```

```
Enter the IP address: 192.167.4.1
Binary representation of the IP address: 11000000.10100111.00000100.00000001
Class C
Network address : 192.167.4.0
Net id : 192.167.4
Host address : 0.0.0.1
Host ID : 1
First address : 192.167.4.0
Last address : 192.167.4.255
Total number of address : 255
Enter the value for custom masking: 24
Custom Masking: 255.255.255.128


Enter the IP address: 300.254.2.35
Invaid Ipv4 address

Enter the value for custom masking: 40
Out of Range error

```