MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Mini Project Report
of
Computer Networks LAB

TITLE
Tool to validate IP Address and generate random address blocks

SUBMITTED BY

NAME	REGISTRATION NUMBER	ROLL NUMBER	SECTION
RAJPREET LAL DAS	200905052	08	CSE ‘C’
KAUSTUBH PANDEY	200905142	24	CSE ‘C’

ACKNOWLEDGEMENT

We would like to thank the Department of Computer Science and Engineering for giving us the opportunity to work on a project to help in our understanding of the coursework.

We would like to thank our teacher Ms. Tanuja Shailesh for guiding us through this project and our coursework of Computer Networks. Her teachings have inspired us to take up this implementation. We would also like to express our sincere gratitude to Ms. Suchithra Shetty who also guided us during our labs.

A particular acknowledgement goes to our labmates who helped us by exchanging interesting ideas and information. Finally, we would like to thank our parents for their unwavering support and continuous encouragement in our educational journey.

ABSTRACT

IP addresses are the identifier that allows information to be sent between devices on a network. They contain location information and make devices accessible for communication. The internet needs a way to differentiate between different computers, routers, and websites. IP addresses provide a way of doing so and form an essential part of how the internet works. An IP address is a string of numbers separated by periods. IP addresses are expressed as a set of four numbers. Each number in the set can range from 0 to 255. So, the full IP addressing range goes from 0.0.0.0 to 255.255.255.255.

A Subnet Mask is a 32-bit combination used to describe which portion of an address refers to the subnet and which part refers to the host.

About the tool:

- The tool takes an IP address along with mask as separate inputs.
- The tool checks if the given IP is valid or not.
- It then displays the first address, last address and number of addresses in the block.
- It keeps the size of this block constant and gives three other blocks available of the same size with their network addresses.

CHAPTER 1 : INTRODUCTION

1.1 CONCEPTS IN COMPUTER NETWORK

1.1.1 IP Address

An IP address is a unique address that identifies a device on the internet or a local network. IP stands for "Internet Protocol," which is the set of rules governing the format of data sent via the internet or local network.

In essence, IP addresses are the identifier that allows information to be sent between devices on a network: they contain location information and make devices accessible for communication. The internet needs a way to differentiate between different computers, routers, and websites. IP addresses provide a way of doing so and form an essential part of how the internet works.

An IP address is a string of numbers separated by periods. IP addresses are expressed as a set of four numbers — an example address might be 192.158.1.38. Each number in the set can range from 0 to 255. So, the full IP addressing range goes from 0.0.0.0 to 255.255.255.255.

1.1.2 Subnet (Sub Network)

Every website needs a unique IP address, in order to uniquely identify the website, we are dividing the IP network into two or more networks called subnet, which is preferred to control network traffic.
It is a smaller network inside a large network. This technique makes the network routing an efficient one.

The main features of subnet are as follows :
- Reduce network congestion
- Control network growth
- Ease administration
- Boost network security

Generally, we divide the network into smaller networks or subnets which is called as subnetting.

1.1.3 Subnet Mask

A subnet mask is a number that distinguishes the network address and the host address within an IP address. A subnet is a smaller network within a network that requires a subnet mask.Subnet Mask is made by setting the network bits to all "1"s and setting host bits to all "0"s. Within a given network, two host addresses are reserved for a special purpose, and cannot be assigned to hosts.

The binary "0" in the subnet mask tell us about the host address. It tells us about the IP of the host which has done subnetting.
Subnetting is the process of dividing a network into two or more subnets.

1.2 Hardware and Software Requirements

1.2.1 Hardware Requirements

- Standard Pentium Series Processor. (Recommended Pentium III or Advanced).
- Minimum 256 MB RAM. (Recommended 512 MB)
- HDD Storage capacity of 4 GB with 5400 rpm or more.

1.2.2 Software Requirements

- Operating System - Ubuntu (Linux)
- Programming Language - C++
- Programming Tool - Visual Studio Code and Linux Terminal
- Documentation Tool - ,Google Docs,Microsoft Word, Adobe Acrobat PDF Viewer

CHAPTER 2 : PROBLEM DEFINITION

Design a tool which takes ip along with mask as input and, then, gives the validity of the ip, along with first address, last address and the number of addresses in the block. Keeping the size of this block as constant, it should give at least other three blocks avilable of the same size with their network address.

CHAPTER 3 : OBJECTIVES

The main objective of the project is to check the validity of the ip for the given input and to get its first address, last address and number of addresses in the block, and to get random network address of other three blocks keeping the block size constant for all.

CHAPTER 4 : METHODOLOGY

Algorithms for the required project is as follows:

Step 1: Taking user input of IP Address and Subnet Mask:

For this project,we are using C++ as implementation language.We take input of IP address in its dotted notation format and subnet mask in integer format.

Step 2 : Validating the user input

After taking user input,we validate the IP address by splitting the string (as we have taken the input in form of string) and then checking the respective elements of the array of strings under the bounded conditions i.e. each substring in between delimiters (here ".") are in respective range.

Step 3 : Generating First and Last Address and range of Address in block

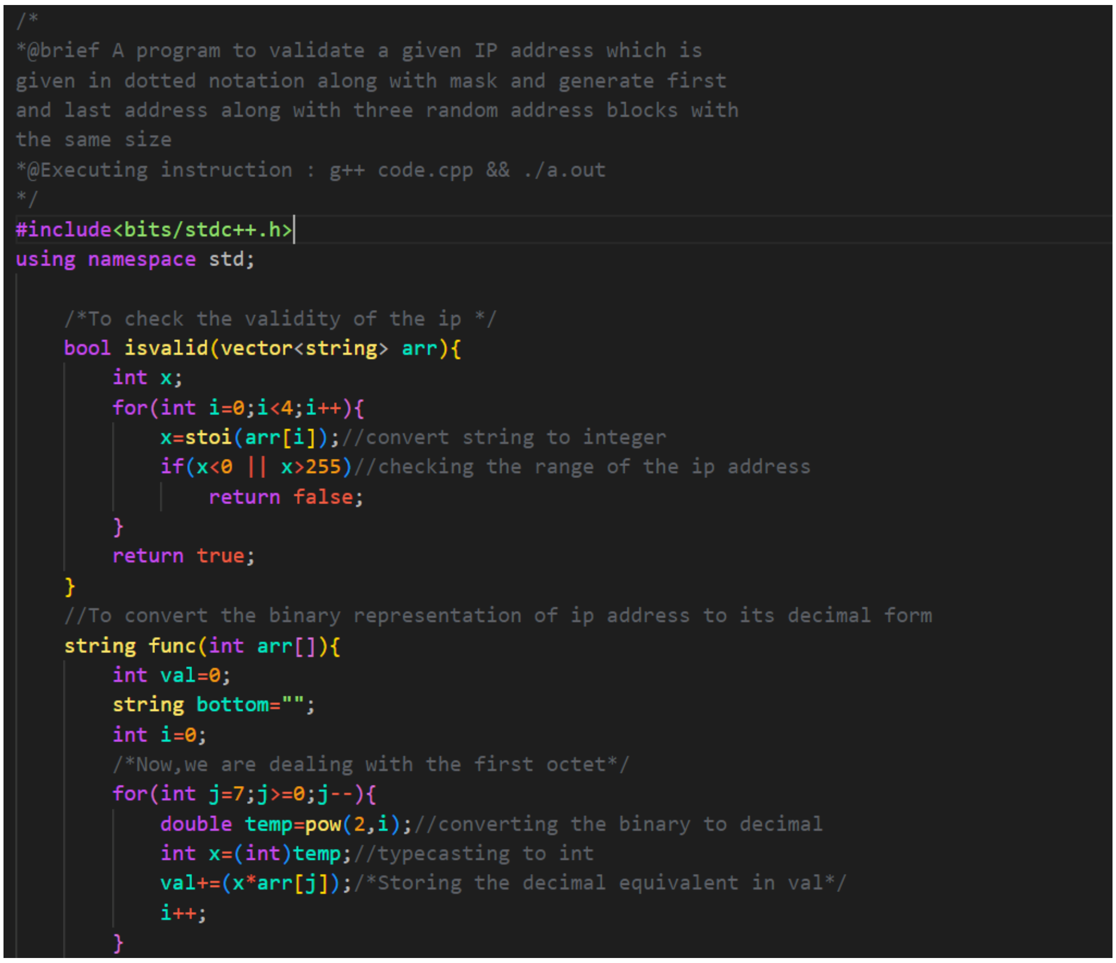
After validating the input, we convert the given string to binary and repeat the same bit sequence till the mask and generate first and last address respectively by zeroing or oneing the leftover bits after the mask.We then generate number of addresses in the corresponding block i.e. the range of addresses.

Step 4 : Generating Random address blocks

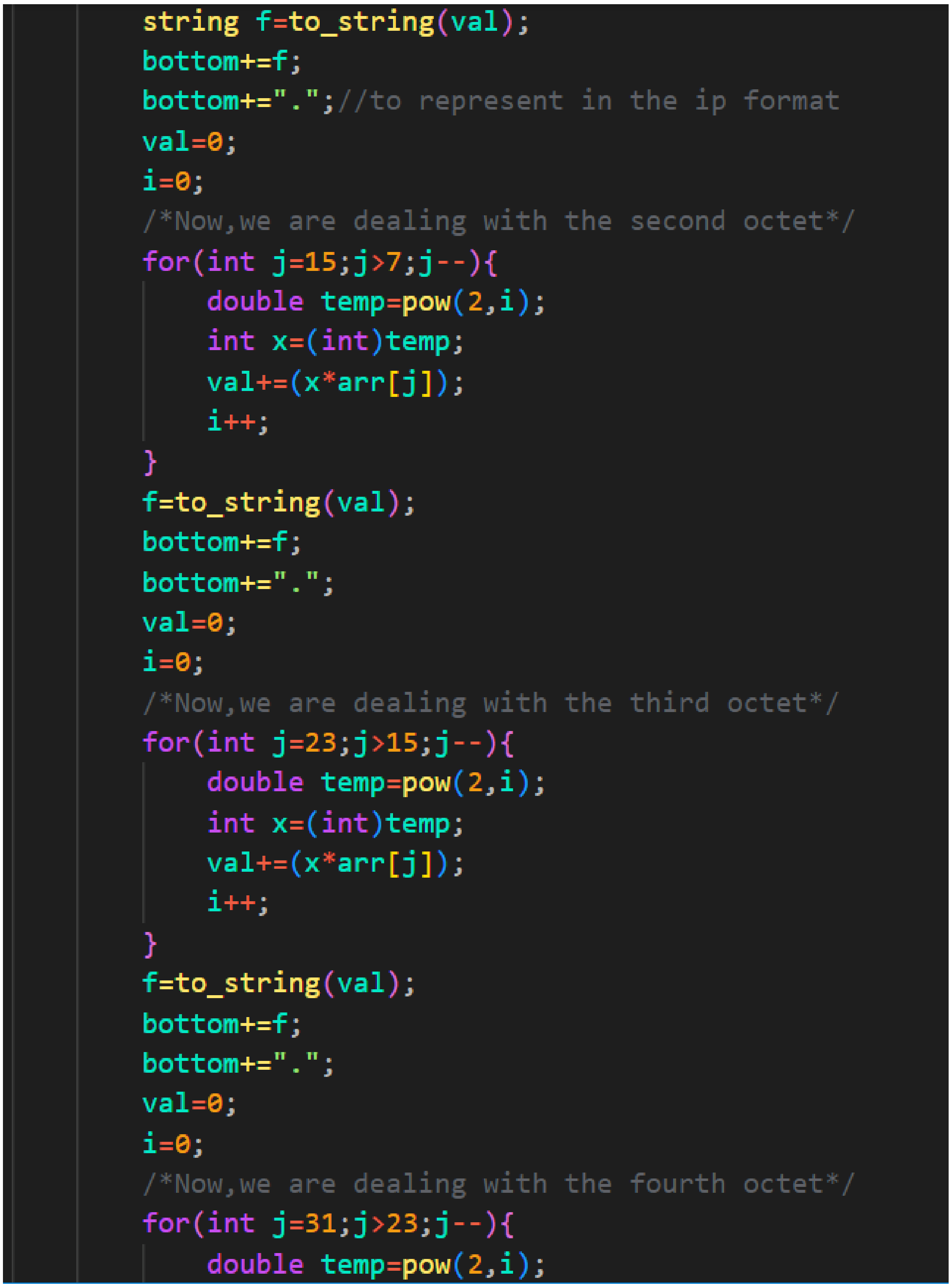
After obtaining the first and last address of the block which the given address belongs to,we generated random address blocks which have the same subnet mask and the same size as the user input's address block.

CHAPTER 5 : IMPLEMENTATION DETAILS

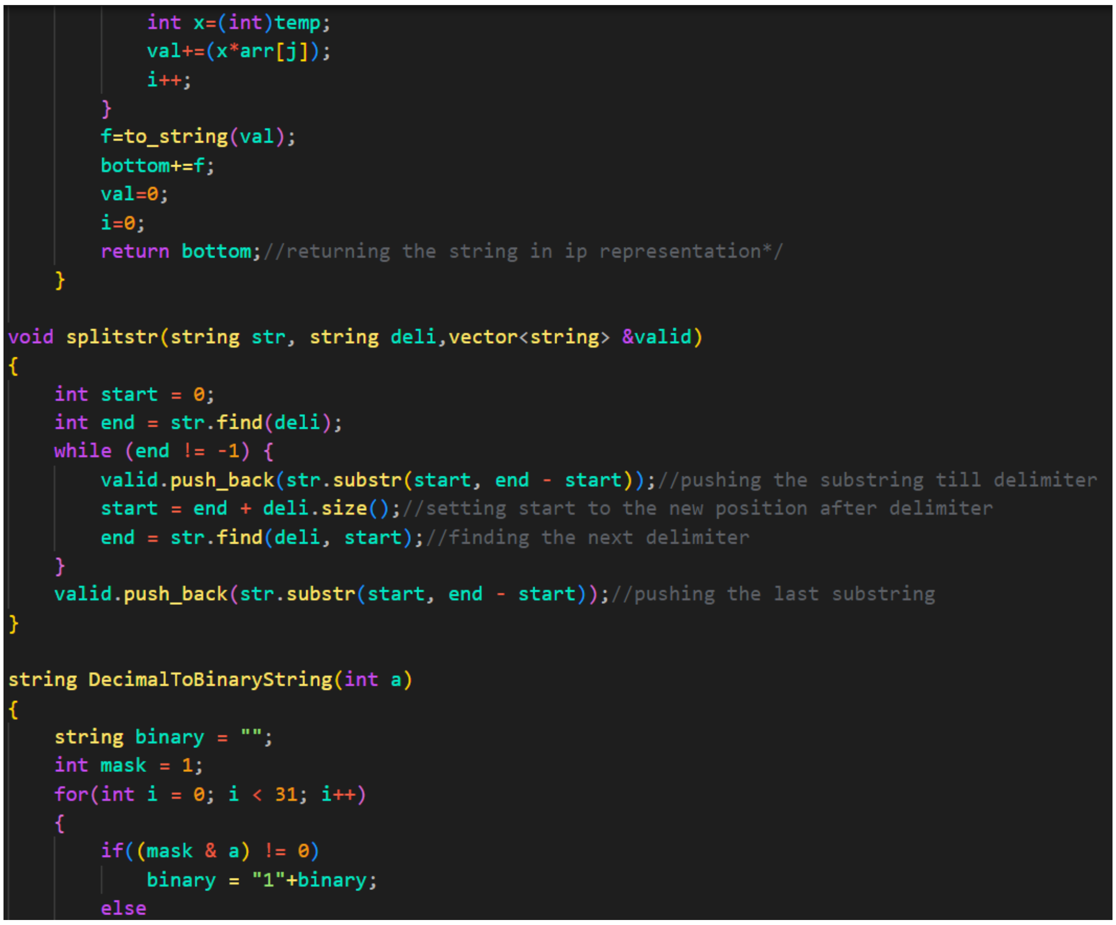
5.1 CODE



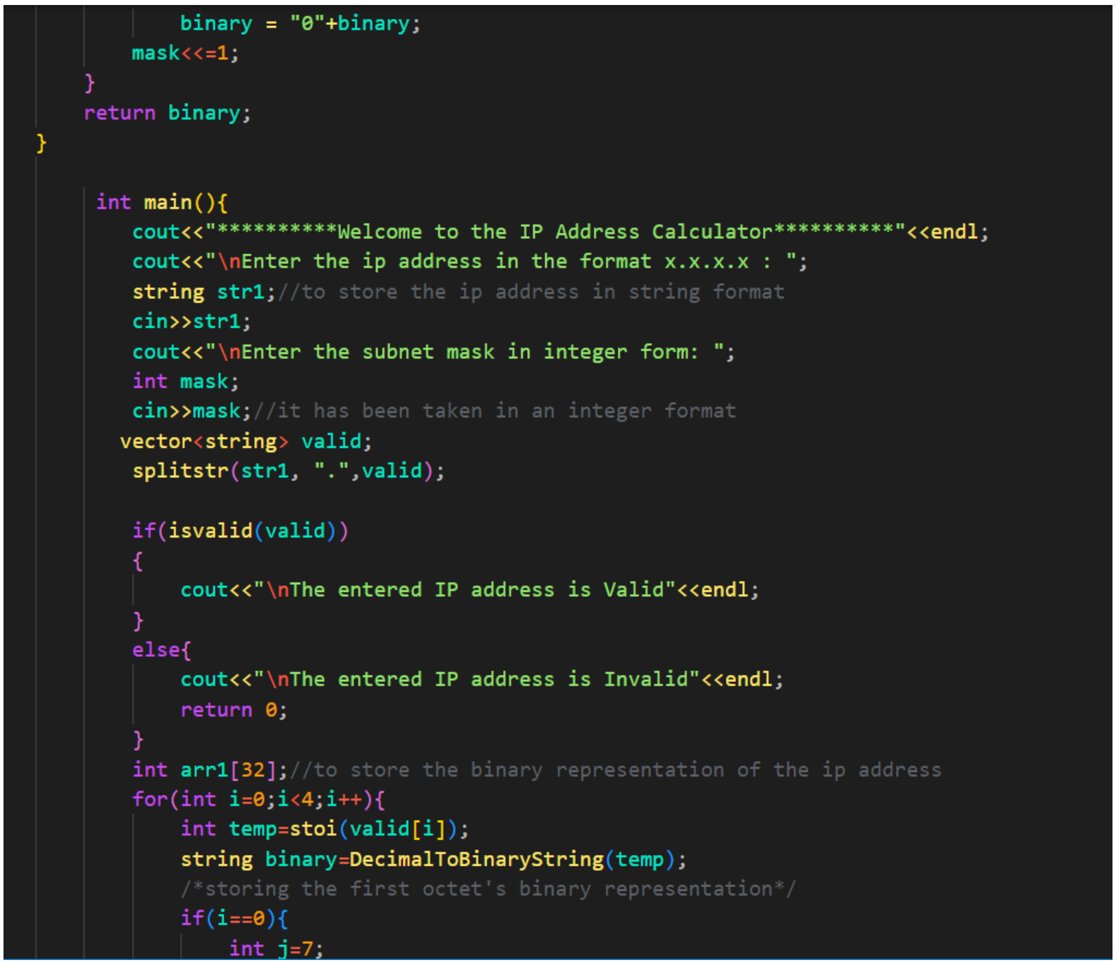
```
/*
*@brief A program to validate a given IP address which is
given in dotted notation along with mask and generate first
and last address along with three random address blocks with
the same size
*@Executing instruction : g++ code.cpp && ./a.out
*/
#include<bits/stdc++.h>
using namespace std;

    /*To check the validity of the ip */
    bool isvalid(vector<string> arr){
        int x;
        for(int i=0;i<4;i++){
            x=stoi(arr[i]);//convert string to integer
            if(x<0 || x>255)//checking the range of the ip address
                return false;
        }
        return true;
    }
    //To convert the binary representation of ip address to its decimal form
    string func(int arr[]){
        int val=0;
        string bottom="";
        int i=0;
        /*Now,we are dealing with the first octet*/
        for(int j=7;j>=0;j--){
            double temp=pow(2,i);//converting the binary to decimal
            int x=(int)temp;//typecasting to int
            val+=(x*arr[j]);/*Storing the decimal equivalent in val*/
            i++;
        }
```



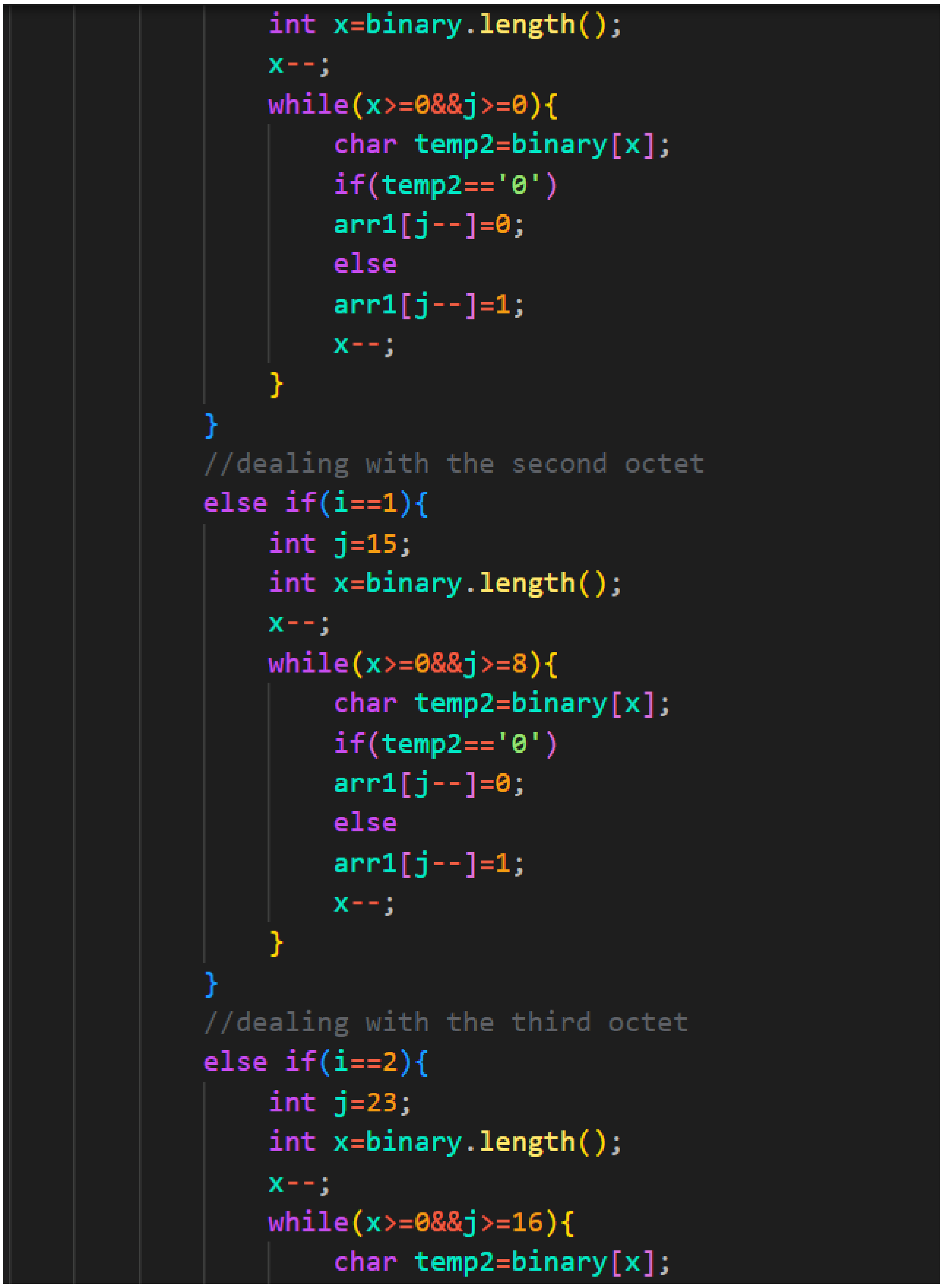
```
        string f=to_string(val);
        bottom+=f;
        bottom+=".";//to represent in the ip format
        val=0;
        i=0;
        /*Now,we are dealing with the second octet*/
        for(int j=15;j>7;j--){
            double temp=pow(2,i);
            int x=(int)temp;
            val+=(x*arr[j]);
            i++;
        }
        f=to_string(val);
        bottom+=f;
        bottom+=".";
        val=0;
        i=0;
        /*Now,we are dealing with the third octet*/
        for(int j=23;j>15;j--){
            double temp=pow(2,i);
            int x=(int)temp;
            val+=(x*arr[j]);
            i++;
        }
        f=to_string(val);
        bottom+=f;
        bottom+=".";
        val=0;
        i=0;
        /*Now,we are dealing with the fourth octet*/
        for(int j=31;j>23;j--){
            double temp=pow(2,i);
```



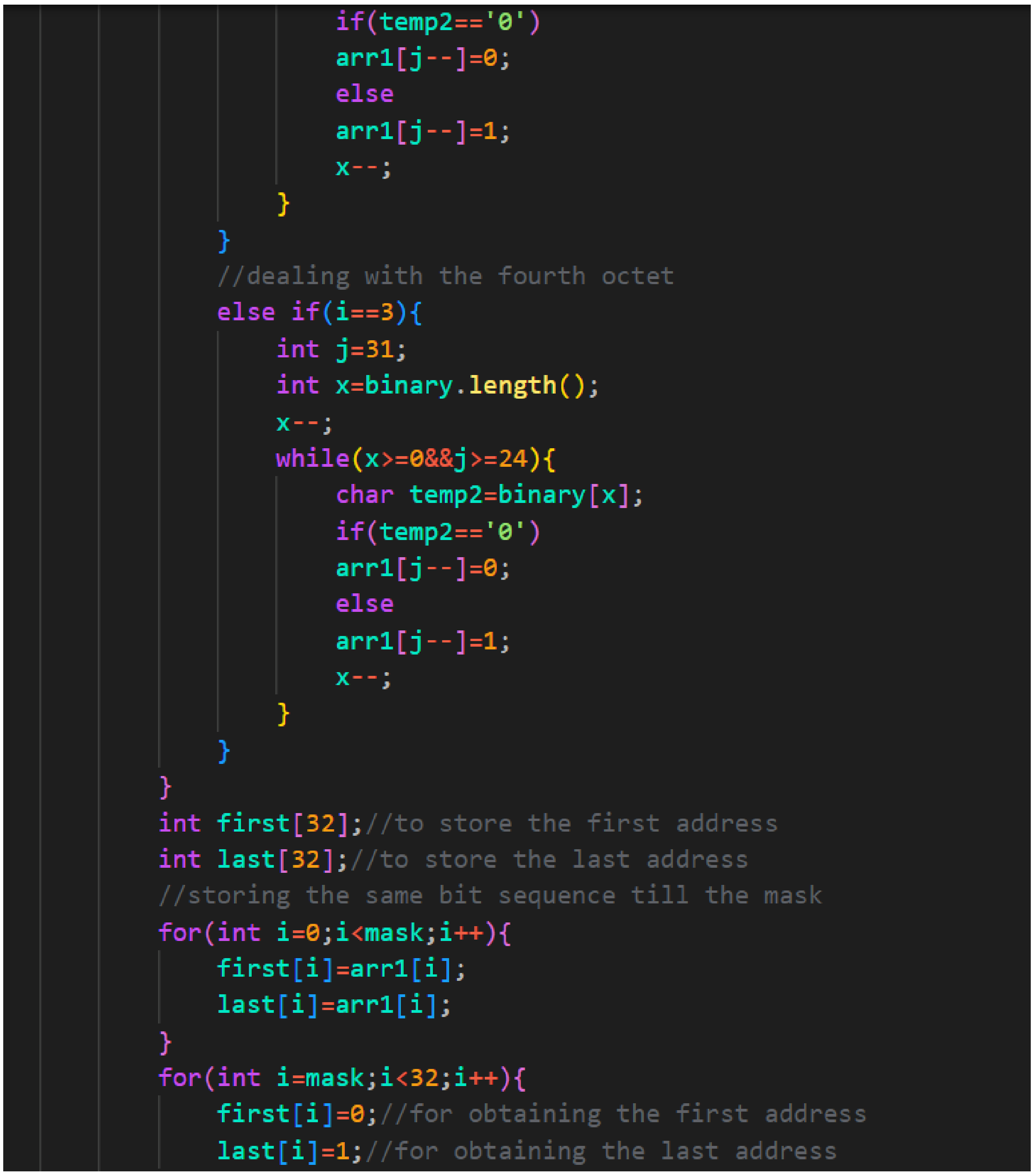
```
            int x=(int)temp;
            val+=(x*arr[j]);
            i++;
        }
        f=to_string(val);
        bottom+=f;
        val=0;
        i=0;
        return bottom;//returning the string in ip representation*/
    }

void splitstr(string str, string deli,vector<string> &valid)
{
    int start = 0;
    int end = str.find(deli);
    while (end != -1) {
        valid.push_back(str.substr(start, end - start));//pushing the substring till delimiter
        start = end + deli.size();//setting start to the new position after delimiter
        end = str.find(deli, start);//finding the next delimiter
    }
    valid.push_back(str.substr(start, end - start));//pushing the last substring
}

string DecimalToBinaryString(int a)
{
    string binary = "";
    int mask = 1;
    for(int i = 0; i < 31; i++)
    {
        if((mask & a) != 0)
            binary = "1"+binary;
        else
```



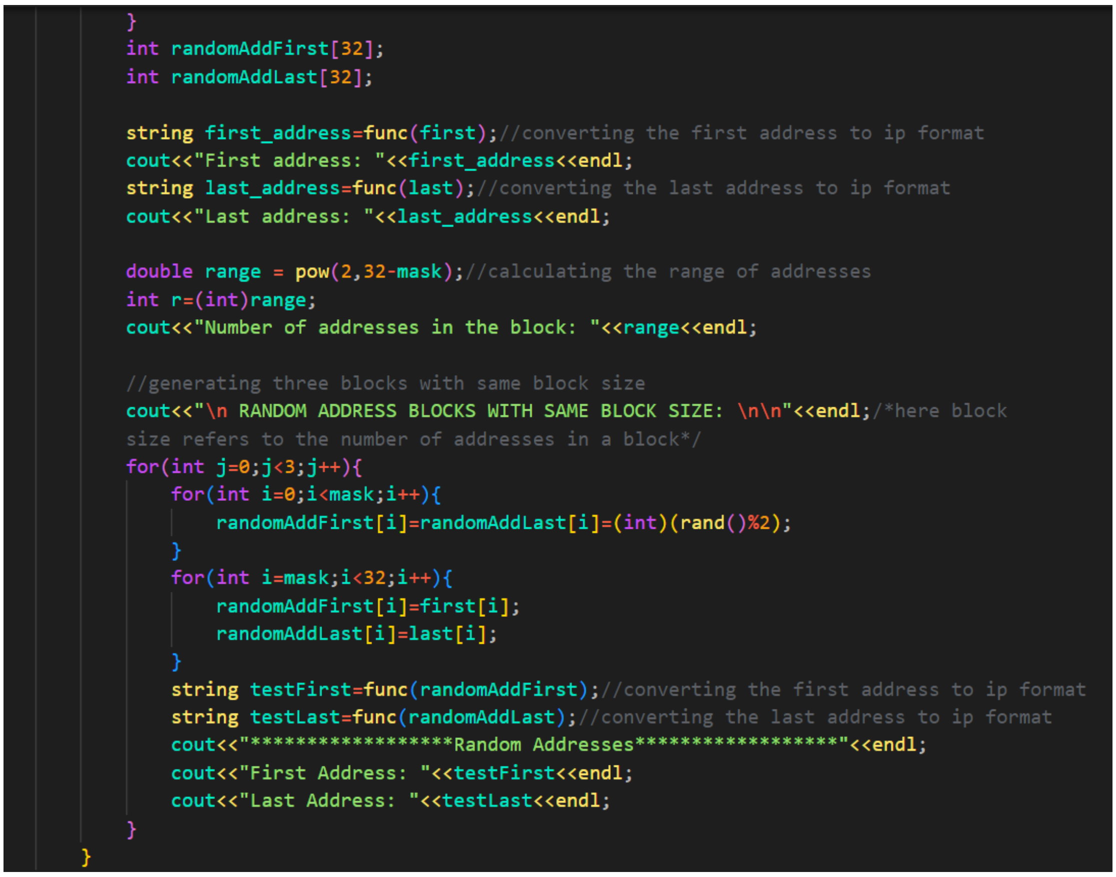
```
            binary = "0"+binary;
        mask<<=1;
    }
    return binary;
}

    int main(){
        cout<<"**********Welcome to the IP Address Calculator**********"<<endl;
        cout<<"\nEnter the ip address in the format x.x.x.x : ";
        string str1;//to store the ip address in string format
        cin>>str1;
        cout<<"\nEnter the subnet mask in integer form: ";
        int mask;
        cin>>mask;//it has been taken in an integer format
       vector<string> valid;
        splitstr(str1, ".",valid);

        if(isvalid(valid))
        {
            cout<<"\nThe entered IP address is Valid"<<endl;
        }
        else{
            cout<<"\nThe entered IP address is Invalid"<<endl;
            return 0;
        }
        int arr1[32];//to store the binary representation of the ip address
        for(int i=0;i<4;i++){
            int temp=stoi(valid[i]);
            string binary=DecimalToBinaryString(temp);
            /*storing the first octet's binary representation*/
            if(i==0){
                int j=7;
```



```
                int x=binary.length();
                x--;
                while(x>=0&&j>=0){
                    char temp2=binary[x];
                    if(temp2=='0')
                    arr1[j--]=0;
                    else
                    arr1[j--]=1;
                    x--;
                }
            }
            //dealing with the second octet
            else if(i==1){
                int j=15;
                int x=binary.length();
                x--;
                while(x>=0&&j>=8){
                    char temp2=binary[x];
                    if(temp2=='0')
                    arr1[j--]=0;
                    else
                    arr1[j--]=1;
                    x--;
                }
            }
            //dealing with the third octet
            else if(i==2){
                int j=23;
                int x=binary.length();
                x--;
                while(x>=0&&j>=16){
                    char temp2=binary[x];
```



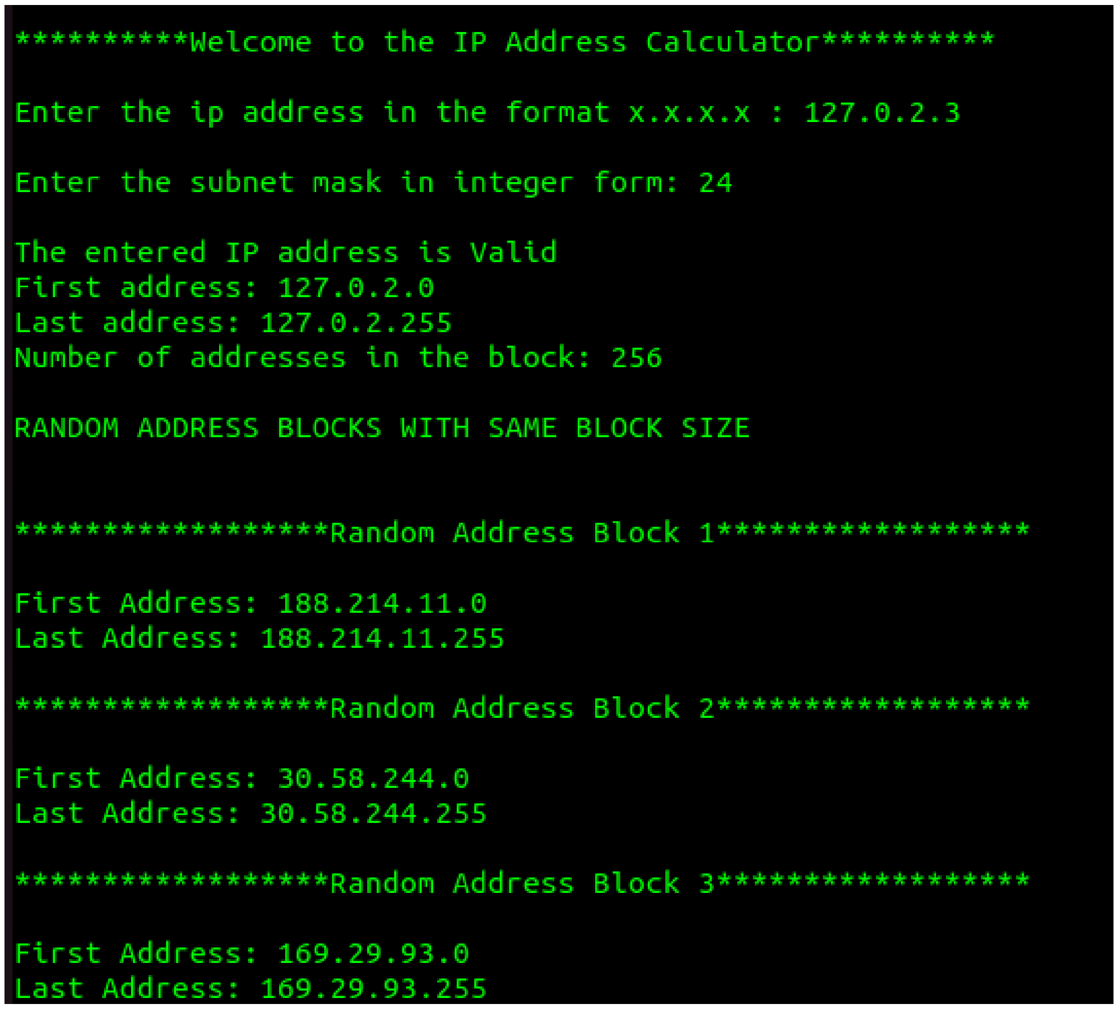
```
                    if(temp2=='0')
                    arr1[j--]=0;
                    else
                    arr1[j--]=1;
                    x--;
                }
            }
            //dealing with the fourth octet
            else if(i==3){
                int j=31;
                int x=binary.length();
                x--;
                while(x>=0&&j>=24){
                    char temp2=binary[x];
                    if(temp2=='0')
                    arr1[j--]=0;
                    else
                    arr1[j--]=1;
                    x--;
                }
            }
        }
        int first[32];//to store the first address
        int last[32];//to store the last address
        //storing the same bit sequence till the mask
        for(int i=0;i<mask;i++){
            first[i]=arr1[i];
            last[i]=arr1[i];
        }
        for(int i=mask;i<32;i++){
            first[i]=0;//for obtaining the first address
            last[i]=1;//for obtaining the last address
```



```
        }
        int randomAddFirst[32];
        int randomAddLast[32];

        string first_address=func(first);//converting the first address to ip format
        cout<<"First address: "<<first_address<<endl;
        string last_address=func(last);//converting the last address to ip format
        cout<<"Last address: "<<last_address<<endl;

        double range = pow(2,32-mask);//calculating the range of addresses
        int r=(int)range;
        cout<<"Number of addresses in the block: "<<range<<endl;

        //generating three blocks with same block size
        cout<<"\n RANDOM ADDRESS BLOCKS WITH SAME BLOCK SIZE: \n\n"<<endl;/*here block
        size refers to the number of addresses in a block*/
        for(int j=0;j<3;j++){
            for(int i=0;i<mask;i++){
                randomAddFirst[i]=randomAddLast[i]=(int)(rand()%2);
            }
            for(int i=mask;i<32;i++){
                randomAddFirst[i]=first[i];
                randomAddLast[i]=last[i];
            }
            string testFirst=func(randomAddFirst);//converting the first address to ip format
            string testLast=func(randomAddLast);//converting the last address to ip format
            cout<<"*****************Random Addresses*****************"<<endl;
            cout<<"First Address: "<<testFirst<<endl;
            cout<<"Last Address: "<<testLast<<endl;
        }
    }
```

5.2 OUTPUT

Case 1:



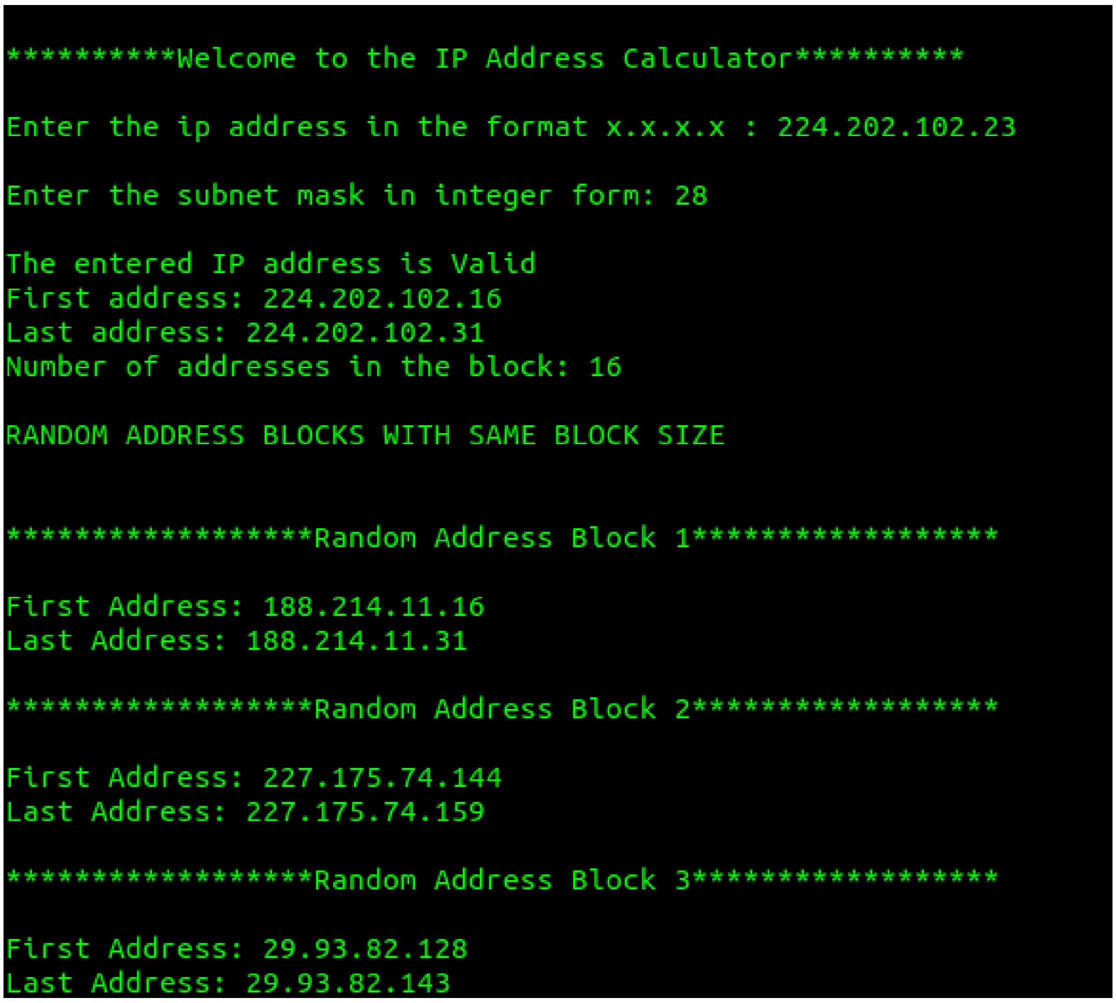
```
**********Welcome to the IP Address Calculator**********

Enter the ip address in the format x.x.x.x : 127.0.2.3

Enter the subnet mask in integer form: 24

The entered IP address is Valid
First address: 127.0.2.0
Last address: 127.0.2.255
Number of addresses in the block: 256

RANDOM ADDRESS BLOCKS WITH SAME BLOCK SIZE


******************Random Address Block 1******************

First Address: 188.214.11.0
Last Address: 188.214.11.255

******************Random Address Block 2******************

First Address: 30.58.244.0
Last Address: 30.58.244.255

******************Random Address Block 3******************

First Address: 169.29.93.0
Last Address: 169.29.93.255
```

(fig 1)

The given subnet mask has 2^(32-24)=256 addresses in the blocks.

As we can see,the tool generated 3 Random Address Blocks with the same subnet mask (24 or 255.255.255.0) of the same size (256).

Case 2:



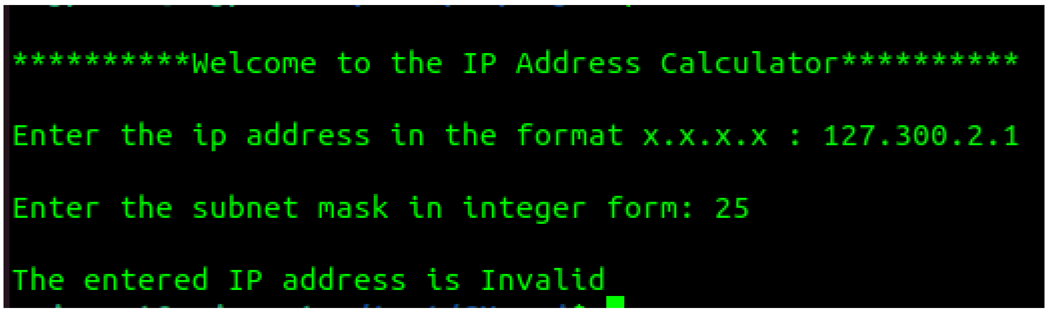
```
**********Welcome to the IP Address Calculator**********

Enter the ip address in the format x.x.x.x : 224.202.102.23

Enter the subnet mask in integer form: 28

The entered IP address is Valid
First address: 224.202.102.16
Last address: 224.202.102.31
Number of addresses in the block: 16

RANDOM ADDRESS BLOCKS WITH SAME BLOCK SIZE


******************Random Address Block 1******************

First Address: 188.214.11.16
Last Address: 188.214.11.31

******************Random Address Block 2******************

First Address: 227.175.74.144
Last Address: 227.175.74.159

******************Random Address Block 3******************

First Address: 29.93.82.128
Last Address: 29.93.82.143
```

(fig 2)

The given subnet mask has 2^(32-28)=16 addresses in the blocks.

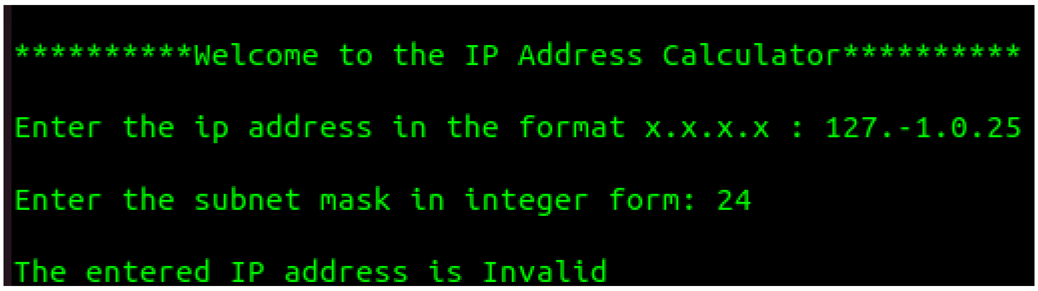
As we can see,the tool generated 3 Random Address Blocks with the same subnet mask (28 or 255.255.255.240) of the same size (16).

Case 3:



```
**********Welcome to the IP Address Calculator**********

Enter the ip address in the format x.x.x.x : 127.300.2.1

Enter the subnet mask in integer form: 25

The entered IP address is Invalid
```

(fig 3)



```
**********Welcome to the IP Address Calculator**********

Enter the ip address in the format x.x.x.x : 127.-1.0.25

Enter the subnet mask in integer form: 24

The entered IP address is Invalid
```

(fig 4)

The Octet values should lie between 0 and 255 (inclusive), hence error is thrown.

CHAPTER 6 : CONTRIBUTION SUMMARY

- We were successfully able to implement IP validation and random address block generator tool.
- Kaustubh helped with the IP validation code segment as well as first and last address block generation and Rajpreet contributed with decimal to binary function and random address block generation code segment.We both contributed towards the documentation and formatting of the project report.
- On executing this program we were successfully able to validate ip addresses and generate random address blocks.
- The overall objective of this project was successfully achieved.

CHAPTER 7 : REFERENCES

7.1 BOOK

- James F. Kurose, Keith W. Ross, “Transport Layer”, in Computer Networking: A Top-down Approach, 6th ed. New Jersey, (only U.S. State), The United States of America, Pearson Education, Inc., 2013, ch. 4, sec. 4.4.

7.2 WEBSITES

- https://www.geeksforgeeks.org/supernetting-in-network-layer/
- https://upscfever.com/upsc-fever/en/gatecse/en-gatecse-chp102.html
- https://www.techiedelight.com/validate-ip-address-c++/
- https://www.omnisecu.com/tcpip/how-to-find-network-address-and-broadcast-address-of-a-subnetted-ipv4-address.php
- https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13788-3.html