# IP ADDRESS VALIDATION AND BLOCK ALLOCATION

A MINI PROJECT REPORT

*Submitted by-*

Syed Abrar Ahmed(190905386) and Ayush Gupta(190905372)

*In partial fulfillment for the award of the degree of*

Bachelors of Technology

IN

Computer Science and Engineering

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
*A Constituent Institution of Manipal University*

Department of Computer Science and Engineering

November 2022

# Department of Computer Science & Engineering

## BONAFIDE CERTIFICATE

Certified that this project report “IP Address Validation and Block Allocation” is the bonafide work of “SYED ABRAR AHMED and AYUSH GUPTA” who carried out the mini project work under my supervision.

| | |
|---|---|
| <<Signature of the HOD with date>> | <<Signature of the Supervisor with date>> |
| <<Name of the HOD>> | <<Name of the Supervisor>> |
| Professor and Head | <<Academic Designation of Supervisor>> |

Submitted to the Viva Voce Examination held on ______________________

| | |
|---|---|
| EXAMINER 1 | EXAMINER 2 |

# Abstract

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.

Internet Protocol works the same way as any other language, communicating using set guidelines to pass information. All devices find, send, and exchange information with other connected devices using this protocol. By speaking the same language, any computer in any location can talk to one another.

A valid IP address must be in the form of ‘xxx.xxx.xxx.xxx’, where xxx is a number from 0-255. There are a few reserved addresses (RFC 1918) that cannot be used. 10.x.x.x cannot be used along with 192.168.x.x, 172.16.0.0 to 172.31.255.255.

**Versions of IP Addresses**

There are two versions of IP in use today, IPv4 and IPv6. The original IPv4 protocol is still used today on both the internet and many corporate networks. However, the IPv4 protocol only allowed for $2^{32}$ addresses. This, coupled with how addresses were allocated, led to a situation where there would not be enough unique addresses for all devices connected to the internet.

IPv6 was developed by the Internet Engineering Task Force (IETF) and was formalized in 1998. This upgrade substantially increased the available address space and allowed for $2^{128}$ addresses. In addition, there were changes to improve the efficiency of IP packet headers and improvements to routing and security.

IPv4 addresses are 32-bit binary numbers, consisting of the two sub-addresses (identifiers) that identify the network and the host to the network, with an imaginary boundary separating the two. An IP address is, as such, generally shown as four octets of numbers from 0-255 represented in decimal form instead of binary form. For example, the address 168.212.226.204 represents the 32-bit binary number 10101000.11010100.11100010.11001100.

The binary number is essential because that will determine which class of network the IP address belongs to.

An IPv4 address is typically expressed in dotted-decimal notation, with every eight bits (octet) represented by a number from one to 255, each separated by a dot. An example IPv4 address would look like this:

192.168.17.43

IPv4 addresses are composed of two parts. The first numbers in the address specify the network, while the latter numbers specify the specific host. A subnet mask determines which part of an address is the network part and addresses the particular host.

A packet with a destination address that is not on the same network as the source address will be forwarded or routed to the appropriate network. Once on the correct network, the host part of the address determines which interface the packet gets delivered to.

This project is based on Classless Addressing, which is a concept of addressing the IPv4 addresses. It was adopted after the failure of classful addressing. The classful addressing leads to a wastage of addresses as it assigns a fixed-size block of addresses to the customer. The classless IPv4 addressing **does not divide** the address space into classes like classful addressing. It provides a **variable-length of blocks**, which have a range of addresses according to the needs of users, which **prevents the wastage of addresses**. One can request for a block having **one address, two addresses, four addresses, eight addresses**, and so on. It means that the number of addresses that a customer can request should be the power of **2**.

# Acknowledgments

It has been a great pleasure for us- Ayush Gupta & Syed Abrar Ahmed to undertake this project on ‘IP Address Validation & Block Allocation’.

We want to extend our special thanks to Professor Ms. Roopalakshmi, Associate Professor, CSE Department, MIT, Manipal, who gave us the golden opportunity to work on this unique project.

This project would not have been possible without her enormous help and worthy experience, and her expert guidance through every step of the project.

The completion of the project would also not have been possible without the support of our classmates and our friends, who helped us in the research for the project and motivated us at every stage.

# Table of Contents

# List of Figures

## List of Symbols, Abbreviations & Nomenclature

**IPv4-** Internet Protocol version 4 addresses are 32-bit binary numbers, consisting of the two sub-addresses (identifiers) that identify the network and the host to the network, with an imaginary boundary separating the two.

**IPv6-** Internet Protocol version 6 is the most recent version of the Internet Protocol, the communications protocol that provides an identification and location system for computers on networks and routes traffic across the Internet.

**GUI-** The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation, instead of text-based user interfaces, typed command labels or text navigation.

**Blocks-** IP addresses are assigned to networks in different sized 'blocks'. The size of the 'block' assigned is written after an oblique (/), which shows the number of IP addresses contained in that block.

**Subnet Mask-** Every device has an IP address with two pieces: the client or host address and the server or network address. IP addresses are either configured by a DHCP server or manually configured (static IP addresses). The subnet mask splits the IP address into the host and network addresses, thereby defining which part of the IP address belongs to the device and which part belongs to the network.

**First Address**- The network IP address is the first address of the subnet.

**Last Address**- A host can use the broadcast address to send data to all the other hosts on the subnet. It's the last address on the subnet.

**Classless Addressing**- Classless addressing is a concept of addressing IPv4 addresses. It was adopted after the failure of classful addressing. The classful addressing leads to a wastage of addresses as it assigns a fixed-size block of addresses to the customer.

**CIDR- Classless Inter-Domain Routing** - also known as super netting, is a method of assigning Internet Protocol (IP) addresses that improves the efficiency of address distribution and replaces the previous system based on Class A, Class B and Class C networks.

# Introduction

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.

Internet Protocol works the same way as any other language, communicating using set guidelines to pass information. All devices find, send, and exchange information with other connected devices using this protocol. By speaking the same language, any computer in any location can talk to one another.

**Types of IP Addresses:**

There are different categories of IP addresses, and within each category, different types.

**Consumer IP Addresses**

Every individual or business with an internet service plan will have two types of IP addresses: their private IP addresses and their public IP address. The terms public and private relate to the network location — that is, a private IP address is used inside a network, while a public one is used outside a network.

- **Private IP Addresses**

  Every device that connects to an internet network has a private IP address. This includes computers, smartphones, tablets, and any Bluetooth-enabled devices like speakers, printers, or smart TVs. Routers need a way to identify these items separately, and many items need a way to recognize each other. Therefore, routers generate private IP addresses with unique identifiers for each device that differentiate them on the network.

- **Public IP Addresses**

  A public IP address is the primary address associated with the whole network. While each connected device has its own IP address, they are also included within the primary IP address for the network. The public IP address is provided to the router by the ISP. Typically, ISPs have a large pool of IP addresses that they distribute to their customers.

  Public IP addresses come in two forms – dynamic and static.

  **->Dynamic IP Addresses**

  Dynamic IP addresses change automatically and regularly. ISPs buy a large pool of IP addresses and assign them automatically to their customers. Periodically, they re-assign them and put the older IP addresses back into the pool to be used for other customers. The rationale for this approach is to generate cost savings for the ISP. Automating the regular movement of IP addresses means they don’t have to carry out specific actions to re-establish a customer's IP address if they move home, for example. There are security benefits, too, because a changing IP address makes it harder for criminals to hack into the network interface.

->**Static IP Addresses**

In contrast to dynamic IP addresses, static addresses remain consistent. Once the network assigns an IP address, it remains the same. Most individuals and businesses do not need a static IP address, but for companies that plan to host their own server, it is crucial to have one. This is because a static IP address ensures that websites and email addresses tied to it will have a consistent IP address — vital if other devices need to find them consistently on the web.

There are two types of website IP addresses. For website owners who don’t host their own server and instead rely on a web hosting package – which is the case for most websites – there are two types of website IP addresses. These are shared and dedicated.

· **Shared IP Addresses**

Websites that rely on shared hosting plans from web hosting providers will typically be one of many websites hosted on the same server. This tends to be the case for individual websites or SME websites, where traffic volumes are manageable, and the sites themselves are limited in terms of the number of pages, etc. Websites hosted in this way will have shared IP addresses.

· **Dedicated IP Addresses**

Some web hosting plans have the option to purchase a dedicated IP address (or addresses). This can make obtaining an SSL certificate easier and allows you to run your own File Transfer Protocol (FTP) server. This makes it easier to share and transfer files with multiple people within an organization and allows anonymous FTP sharing options. A dedicated IP address also allows you to access your website using the IP address alone rather than the domain name — applicable if you want to build and test it before registering your domain.

## Versions of IP Addresses

There are two versions of IP in use today, IPv4 and IPv6. The original IPv4 protocol is still used today on both the internet and many corporate networks. However, the IPv4 protocol only allowed for $2^{32}$ addresses. This, coupled with how addresses were allocated, led to a situation where there would not be enough unique addresses for all devices connected to the internet.

IPv6 was developed by the Internet Engineering Task Force (IETF) and was formalized in 1998. This upgrade substantially increased the available address space and allowed for $2^{128}$ addresses. In addition, there were changes to improve the efficiency of IP packet headers and improvements to routing and security.

- **IPv4**

IPv4 addresses are 32-bit binary numbers, consisting of the two sub-addresses (identifiers) that identify the network and the host to the network, with an imaginary boundary separating the two. An IP address is, as such, generally shown as four octets of numbers from

0-255 represented in decimal form instead of binary form. For example, the address 168.212.226.204 represents the 32-bit binary number 10101000.11010100.11100010.11001100.

The binary number is essential because that will determine which class of network the IP address belongs to.

An IPv4 address is typically expressed in dotted-decimal notation, with every eight bits (octet) represented by a number from one to 255, each separated by a dot. An example IPv4 address would look like this:

192.168.17.43

IPv4 addresses are composed of two parts. The first numbers in the address specify the network, while the latter numbers specify the specific host. A subnet mask identifies which part of an address is the network part and addresses the particular host.

A packet with a destination address that is not on the same network as the source address will be forwarded or routed to the appropriate network. Once on the correct network, the host part of the address determines which interface the packet gets delivered to.

**Subnet Masks**

A single IP address identifies both a network and a unique interface on that network. A subnet mask can also be written in dotted-decimal notation and determines where the network part of an IP address ends and where the host portion of the address begins.

When expressed in binary, any bit set to one means the corresponding bit in the IP address is part of the network address. All the bits set to zero mark the corresponding bits in the IP address as part of the host address.

The bits marking the subnet mask must be consecutive ones. Most subnet masks start with 255. and continue until the network mask ends. A Class C subnet mask would be 255.255.255.0.

### IPv4 Address Exhaustion

The original IPv4 specification was designed for the DARPA network that would eventually become the internet. Initially, a test network, no one contemplated how many addresses might be needed in the future. At the time, the $2^{32}$ addresses (4.3 billion) were certainly considered sufficient. However, over time, it became apparent that as currently implemented, the IPv4 address space would not be big enough for a worldwide internet with numerous connected devices per person. The last top-level address blocks were allocated in 2011.

## Classless Advertising

**Classless addressing** is a concept of addressing IPv4 addresses. It was adopted after the failure of classful addressing. The classful addressing leads to a wastage of addresses as it assigns a fixed-size block of addresses to the customer. The classless IPv4 addressing **does not divide** the address space into classes like classful addressing. It provides a **variable length of blocks**, which have a range of addresses according to the needs of users, which **prevents the wastage of addresses**. One can request for a block having **one address, two addresses, four addresses, eight addresses**, and so on. It means that the number of addresses that a customer can request should be the power of **2**.

## CIDR Notation

If an address is provided in classful addressing, we can detect its class as its first few bits help us recognize the class of address. Knowing the class of address, we can find the length of the net-id as it would be 8, 16, or 24.

Like in **classful** addressing, the address was divided into two parts **net-id** and **host-id**. Where net-id would define the address of the network and host-id would define the host address in the corresponding network. And the net-id part and host-id part would vary with the classes.

In the same way, the **classless** addressing also divides the IPv4 address into two parts referred to as **'prefix'** and **'suffix'. Prefix** defines the **network id** , whereas **suffix** defines the **host address** in the corresponding network.

**Addresses** belonging to the **same block** persist the **same prefix**, whereas **each host in a block** has a **different suffix**. As in classful addressing, the length of net-id depends on the class to which the address belongs, and it can only be 8, 16, and 24. On the other hand, the **length of a prefix (n)** can be **0, 1, 2, 3, . . . , 32**. So, the value of the suffix would automatically be (32- length of the prefix).

In classless addressing, prefix length could not be calculated for a given address as it can belong to a block of any prefix length. So, here the length of the prefix is included with each address to ease the extraction of block information.

The **length of the prefix (n)** is added to the last of the address, separated by a **slash**. This is called **slash notation**, and more formally, it is known as **Classless Inter-Domain Routing (CIDR) notation.**

For example:

IPv4 address 167.199.170.82/27 has an added value '27', which is separated by a slash, is a CIDR notation of classless IPv4 address. The value '**27**' denotes the length of the **prefix**. So, the length of the **suffix** would be '3**2-27= 5'**.


Figure 1: 32-bit IPv4 Address

**CIDR Block**

For the correct working of CIDR blocks, **three restrictions** are implemented on the blocks of classless addressing.

1. A block of addresses allocated to an organization must have **contiguous unallocated addresses**.
2. The **number of addresses in a block** allocated to an organization must be the **power of 2**.
3. The **first address** of every block must be **divisible** by the **length of the block**.

Extraction of block information:

1. The number of host addresses in a block

$$N=2^{32-n}$$

*N is the number of host addresses in a block, and n is the length of the prefix.*

2. The first address of a block

*First Address = (any address) AND (network mask)*

3. Last address of a block

*Last Address = (any address) OR [NOT (network mask)]*

Classless Addressing Example

Let us take an IPv4 classless address in CIDR notation, i.e., **167.199.170.82/27**. From the given address, we will calculate the number of addresses in the corresponding network, the first address of the network, and the last address of the network.

Our given address is **167.199.170.82/27**. As we know that the value after the slash in IP address is prefix (n) value=27.

The number of addresses in the network can be calculated with the formula below:

$$N=2^{32-n}$$

$$=2^{32-27}$$

$$=2^{5}$$

$$= 32$$

So, there are 32 addresses in the corresponding network block and it can be assigned to 32 hosts in the network.

The first address of the network can be calculated with the formula below:

First address = (given address) AND (network mask)

The given address is 167.199.170.82

| | |
|---|---|
| Given address in Binary notation | 10100111 11000111 10101010 01010010 |
| | AND |
| Network Mask of given address | 11111111 11111111 11111111 11100000 |
| First address | 10100111 11000111 10101010 01000000 |

Figure 2: First Address Logic

The decimal notation of the first address in the block is 167.199.170.64.

The last address can be calculated with the formula below:

Last address = (given address) OR [NOT (network mask)]

| | |
|---|---|
| Given address in Binary notation | 10100111 11000111 10101010 01010010 |
| | OR |
| Not[Network Mask of given address] | 00000000 00000000 00000000 00011111 |
| Last address | 10100111 11000111 10101010 01011111 |

Figure 3: Last Address Logic

The decimal notation of the last address in the block would be 167.199.170.95.

So, we conclude that the given address 167.199.170.82 is from the block whose first address is 167.199.170.64 and last address is 167.199.170.95 and the number of the hosts in the network is 32.

Now, let us verify the rules for CIDR blocks:

- The number of addresses in a block must be in the power of 2, i.e., in the above case, the number of addresses in the block is 32 which is equal to $2^5$.
- The first address of the block must be divisible by the number size of the block. So, the first address of the block 167.199.170.64 is divisible by $2^5$ as the last five bits of 10100111 11000111 10101010 01000000 address (binary notation of first address 167.199.170.64) is ‘0’.
- The block (first address is 167.199.170.64 and last address is 167.199.170.95) contain contiguous 32 unallocated addresses.

# <u>Methods Used:</u>

We divide our project into 2 halves for better elucidation of our approach

1.Backend: The abstracted logic for calculation of various ip address ranges and its first and last address.

2.Frontend: The Graphical User Interface (GUI)

The project is made using python programming language. We make use of the **tkinter library,** for the development of the graphical user interface. The following section describes our method in further detail.

**Backend:** Abstract logic

Used various logical methods to get the first and last address of the IP address, validity of the mask and IP address along with displaying the available blocks.

**Frontend:** Graphical User Interface

The GUI is used to provide a means to obtain the IP Validation and Block Allocation. The library used for the development of the GUI is tkinter. The standard method for GUI development is followed wherein a window is created by defining various layout elements such as text boxes and elements. Events are then defined for the different elements.

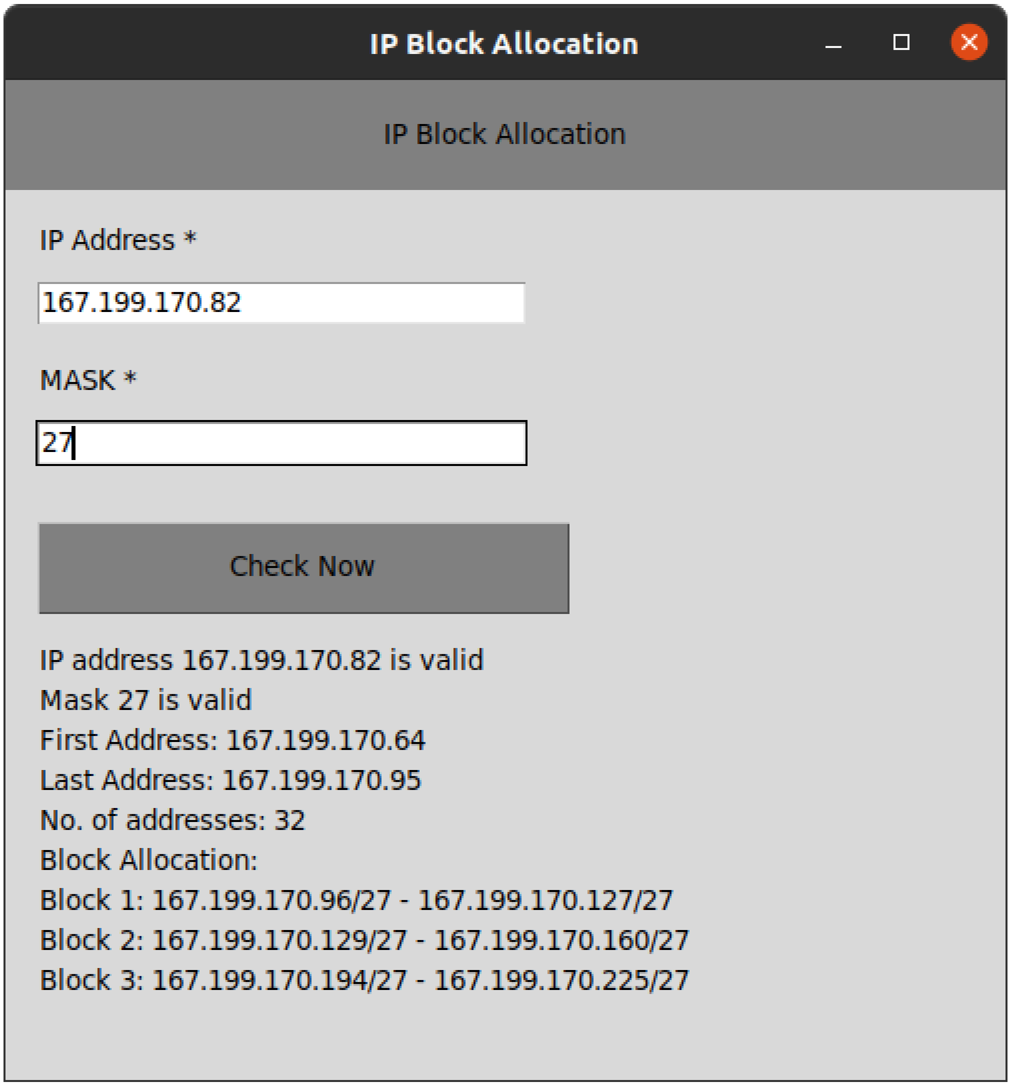


Figure 4: The developed GUI for verifying the IP Address

# Appendix:

## Code:

```
from tkinter import *

import tkinter as tk



"""GUI using tkinter"""



screen = tk.Tk()

screen.geometry("500x500")

screen.title("IP Block Allocation")

heading = Label(text = "IP Block Allocation", bg = "grey", fg = "black",
width = "500", height = "3")

heading.pack()



##command

def save_info():

   ip_info = ip.get()

   prefix_info = prefix.get()

   ipval = validIPv4(ip_info)

   maskval = validmask(prefix_info)

   if(ipval and maskval):

       parts = ip_info.split(".")
```

```python
map_parts = map(int,parts)

#integer list of ip

ip_list=list(map_parts)


mask = 1

for i in range (1,32):

    if(i<prefix_info):

        mask = mask*10 + 1

    else:

        mask = mask*10 + 0


mask = str(mask)


#converting the mask to numeric split int-array

res = []

for idx in range(0, len(mask), 8):

    res.append(mask[idx : idx + 8])


fa=[]

firadd=""

for i in range (0,4):

    fa.append(binaryToDecimal(res[i]) & ip_list[i])

    if(i<3):
```

```
            firadd += str(fa[i])+"."
        else:
            firadd+= str(fa[i])


    ##Last Address
    #Not of mask
    def rev(str):
        rev=''
        for i in str:
            if i == '1':
                rev += '0'
            elif i == '0':
                rev += '1'
        return rev


    la=[]
    lar=""
    for i in range(0,4):
        la.append(binaryToDecimal(rev((res[i]))) | ip_list[i])
        if(i<3):
            lar += str(la[i])+"."
        else:
            lar+= str(la[i])
```

```
        ##Block size and Display
        noofaddr = int(pow(2,32-prefix_info))


        label = Label(screen, text= "First Address: {}".format(firadd))
        label.pack()
        label.place(x = 15, y = 320)
        label = Label(screen, text= "Last Address: {}".format(lar))
        label.pack()
        label.place(x = 15, y = 340)



        label = Label(screen, text= "No. of addresses:
{}".format(noofaddr))
        label.pack()
        label.place(x = 15, y = 360)


        label = Label(screen, text= "Block Allocation:")
        label.pack()
        label.place(x = 15, y = 380)


        ##Block allocation(any 3)
        blocf1 = cloning(la)
        for i in range(0,3):
```

```
            # blocf1 = cloning(la)

            blocf1[3] += noofaddr*i+1

            if(blocf1[3]>255):

                blocf1[2] += 1

                blocf1[3] = 0

            blocl1 = cloning(blocf1)

            blocl1[3] += noofaddr-1

            if(blocl1[3]>255):

                blocl1[2] += 1

                blocl1[3] = 0

            label = Label(screen, text= "Block {}: {}.{}.{}.{}/{} -
{}.{}.{}.{}/{}".format(i+1,blocf1[0],blocf1[1],blocf1[2],blocf1[3],prefix_
info,blocl1[0],blocl1[1],blocl1[2],blocl1[3],prefix_info))

            label.pack()

            label.place(x = 15, y = 400+i*20)




##Labelling GUI

ip_text = Label(text = "IP Address * ",)

prefix_text = Label(text = "MASK * ",)

ip_text.place(x = 15, y = 70)
```

```
prefix_text.place(x = 15, y = 140)




ip = StringVar()

prefix = IntVar()


ip_entry = Entry(textvariable = ip, width = "30")

prefix_entry = Entry(textvariable = prefix, width = "30")


ip_entry.place(x = 15, y = 100)

prefix_entry.place(x = 15, y = 170)


submit = Button(screen,text = "Check Now", width = "30", height = "2",
command = save_info, bg = "grey")

submit.place(x = 15, y = 220)


#binary to decimal

def binaryToDecimal(n):

   num = n

   dec_value = 0


   # Initializing base

   # value to 1, i.e 2 ^ 0
```

```
    base1 = 1


    len1 = len(num)

    for i in range(len1 - 1, -1, -1):

        if (num[i] == '1'):

            dec_value += base1

        base1 = base1 * 2


    return dec_value


#copying ip

def cloning(li1):

    li_copy = li1[:]

    return li_copy


## IP Validation

#validation functions


def validIPv4(ip):

    parts = ip.split(".")


    if len(parts) != 4:

        label = Label(screen, text= "IP address {} is not
valid".format(ip))
```

```
        label.pack()

        label.place(x = 15, y = 280)

        return False


    for part in parts:

        if not isinstance(int(part), int):

            label = Label(screen, text= "IP address {} is not
valid".format(ip))

            label.pack()

            label.place(x = 15, y = 280)

            return False


        if int(part) < 0 or int(part) > 255:

            label = Label(screen, text= "IP address {} is not
valid".format(ip))

            label.pack()

            label.place(x = 15, y = 280)

            return False

    label = Label(screen, text= "IP address {} is valid".format(ip))

    label.pack()

    label.place(x = 15, y = 280)

    return True


def validmask(msk):
```

```
    if(msk<1 or msk>32):

        label = Label(screen, text= "Mask {} is not valid".format(msk))

        label.pack()

        label.place(x = 15, y = 300)

        return False

    else:

        label = Label(screen, text= "Mask {} is valid".format(msk))

        label.pack()

        label.place(x = 15, y = 300)

        return True


screen.mainloop()
```

## Output:

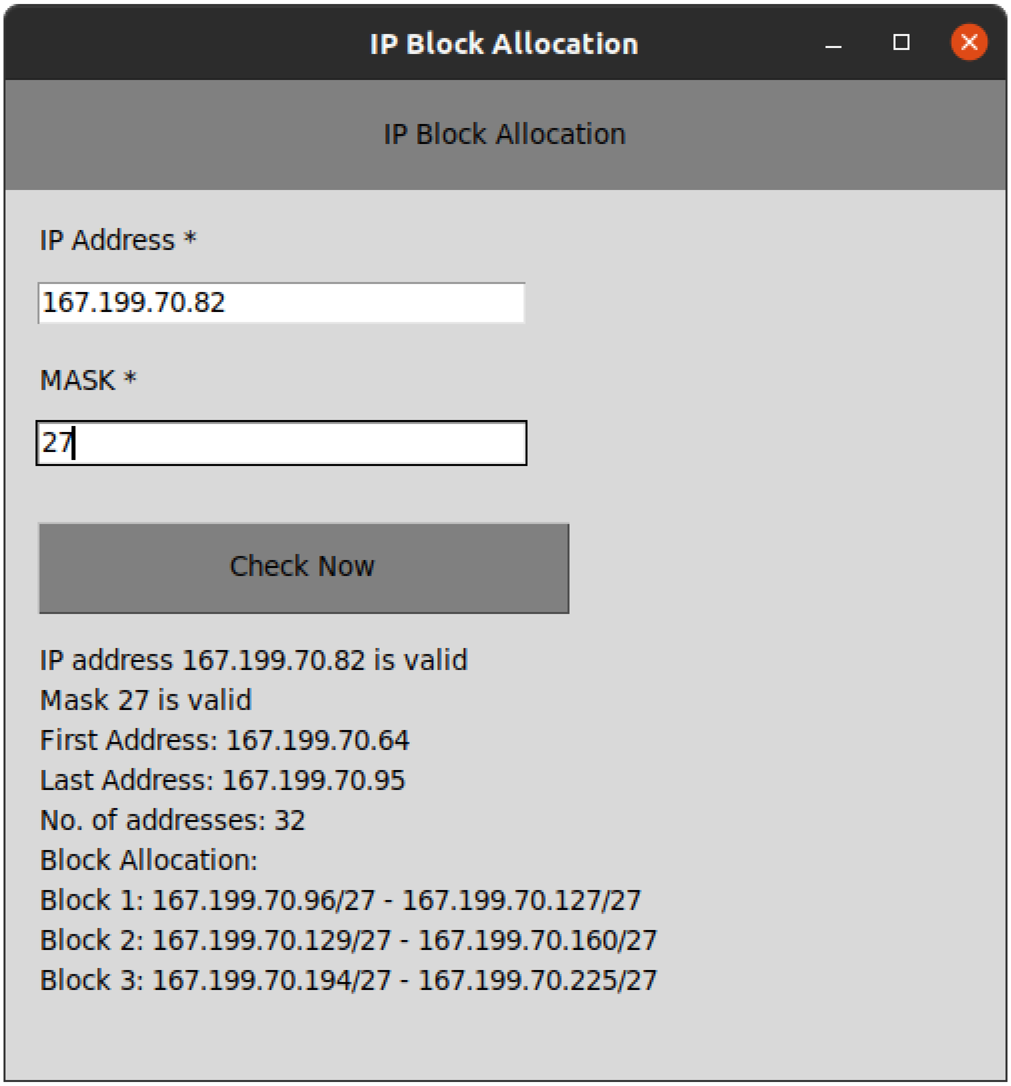


Figure 5: Valid Mask and IP

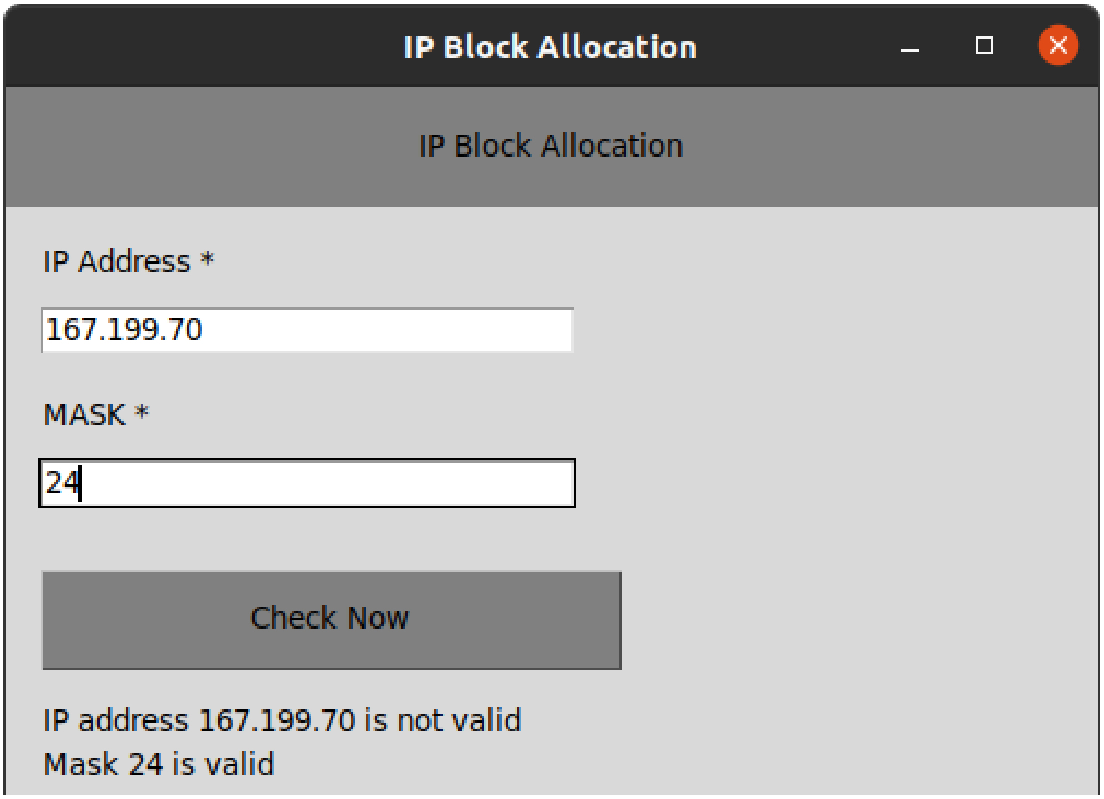


Figure 6: Invalid IP and Valid Mask

# References

1. James F. Kurose & Keith W. Ross, Computer Networking A Top-Down Approach 2.
2.https://www.gatevidyalay.com/classless-addressing-cidr/