

# Developing a Python Tool to Validate IP Addresses, Determine Class, and Calculate Subnet Ranges

---

COMPUTER NETWORKS(Team C11)

NALLA MANASA (23891A05H1)

VADLAPALLI SUKRUTHA(23891A05I9)

KOMIRELLY SNEHA REDDY(23891A05F6)

DANDU NIKHIL VARMA(23891A05E1)

# CONTENTS

1. Project Overview

2. Objectives

3. Concepts Used

4. Understanding  
IP Addresses

5. IP Address  
Classes & Subnet  
Masks

6. Python Tool  
Design & Workflow

7. Outputs

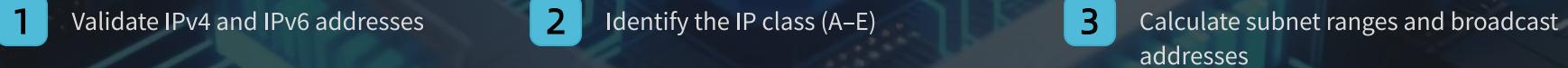
8. Difference  
Between IPv4 and  
IPv6

9. Conclusion &  
Future Scope

# 1. Project Overview

## Aim

To create a Python-based tool that can:

- 

1 Validate IPv4 and IPv6 addresses      2 Identify the IP class (A-E)      3 Calculate subnet ranges and broadcast addresses

## Motivation

Simplify manual network configuration tasks.

## 2. Objectives

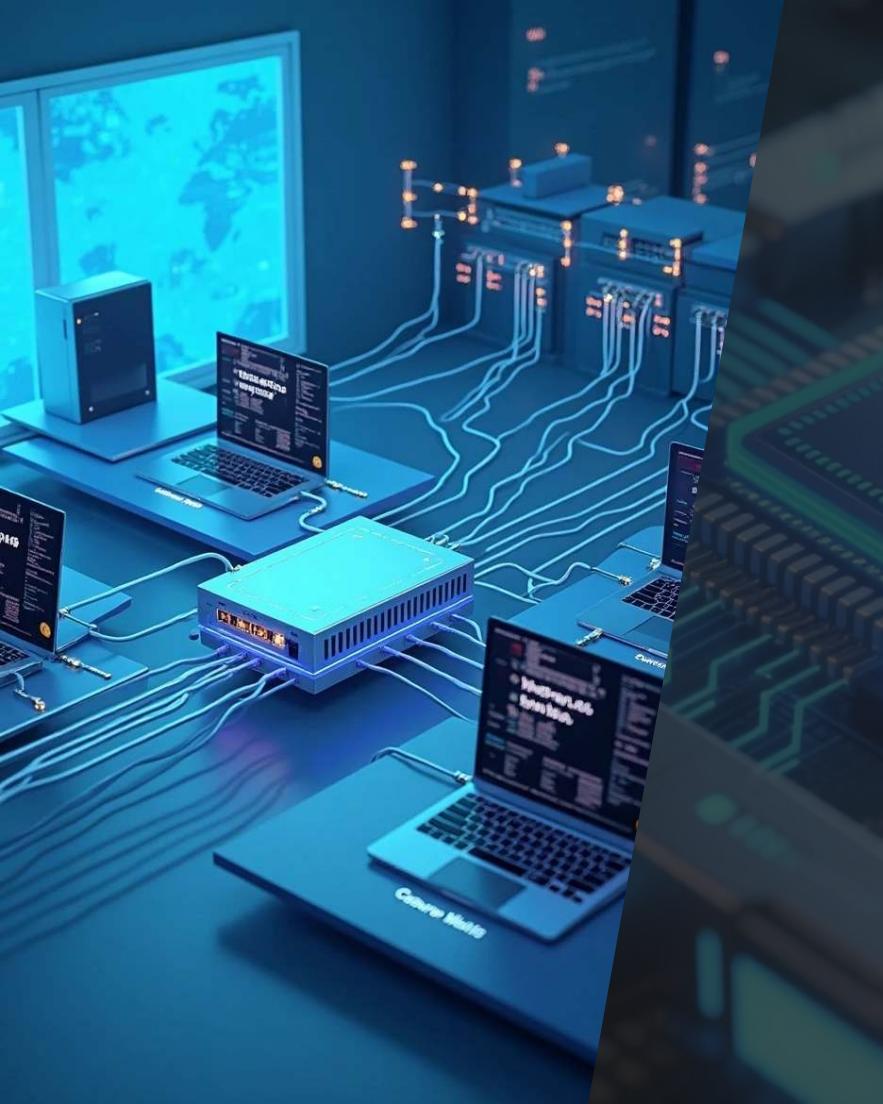
### Project Objectives

Develop an automated IP validation tool

Understand differences between IPv4 and IPv6

Implement subnetting logic programmatically

Provide user-friendly CLI or GUI output



## 3. Concepts Used

### Core Networking & Python Concepts

1

#### Networking Concepts:

- IP address structure
- Subnet masks and network classes
- CIDR notation

2

#### Python Concepts:

- String handling and regex for validation
- ipaddress module (for subnet and class detection)
- Functions and exception handling

# 4. Understanding IP Addresses

## What is an IP Address?

IPv4: 32-bit, written as 192.168.1.1

IPv6: 128-bit, written as  
2001:0db8:85a3::7334

Uniquely identifies a device in a network.

Feature	IPv4	IPv6
Length	32-bit	128-bit
Format	Decimal (4 octets)	Hexadecimal (8 groups)
Example	192.168.0.1	2001:db8::1

# 5. IP Address Classes & Subnet Masks

## IP Classes and Subnet Masks

Class	Range	Default Subnet Mask	Usage
A	1.0.0.0 – 126.255.255.255	255.0.0.0	Large networks
B	128.0.0.0 – 191.255.255.255	255.255.0.0	Medium networks
C	192.0.0.0 – 223.255.255.255	255.255.255.0	Small networks
D	224.0.0.0 – 239.255.255.255	N/A	Multicasting
E	240.0.0.0 – 255.255.255.255	N/A	Experimental

# 6. Python Tool Design & Workflow

## Tool Design

Input: IP address  
(IPv4/IPv6)

Validation: Regex  
or  
ipaddress.ip\_address()

Determine Class  
(if IPv4)

Calculate Subnet and  
Broadcast (using ip\_network())

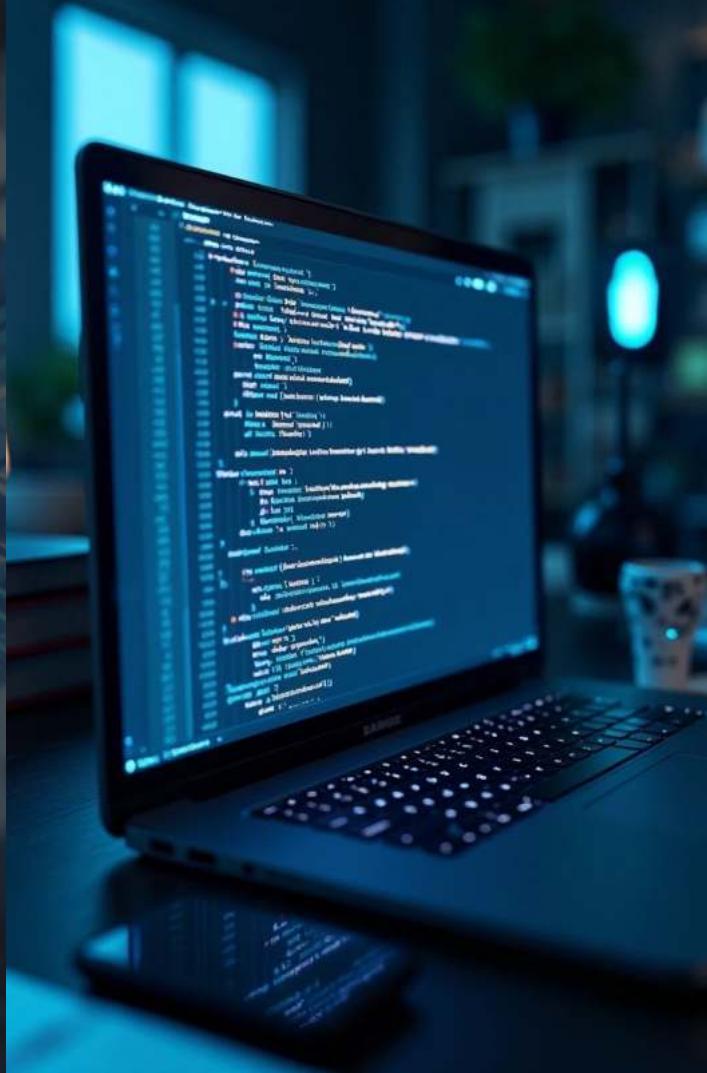
Output results



# Tool Design

## Sample Code Snippet:

```
import ipaddress
ip = input("Enter IP: ")
try:
    addr = ipaddress.ip_address(ip)
    print("Valid IP:", addr)
    if addr.version == 4:
        print("Class:", "A" if addr.is_private else "Public")
    except ValueError:
        print("Invalid IP")
```



## 7.0.OUTPUT

IP Address (IPv4 or IPv6): 192.168.1.10

Subnet Mask / CIDR Prefix (optional): 255.255.255.0

*Input looks valid* 

**Validate**

**Summary**

192.168.1.10 is a private Class C IPv4 address. Subnet 192.168.1.0/24 has 254 usable hosts (from 192.168.1.1 to 192.168.1.254).

Property	Value
Valid IP?	Yes
IP Version	IPv4
IP Address	192.168.1.10
Network Address	192.168.1.0
Prefix Length	/24
Total Addresses in Subnet	256
Private Address?	Yes
Globally Routable?	No
IP Class	Class C
Subnet Mask	255.255.255.0

IP Address (IPv4 or IPv6): 192.168.1.10

Subnet Mask / CIDR Prefix (optional): 255.255.255.0

*Input looks valid* ⓘ

Validate

## Summary

192.168.1.10 is a private Class C IPv4 address. Subnet 192.168.1.0/24 has 254 usable hosts (from 192.168.1.1 to 192.168.1.254).

Property	Value
Prefix Length	/24
Total Addresses in Subnet	256
Private Address?	Yes
Globally Routable?	No
IP Class	Class C
Subnet Mask	255.255.255.0
Wildcard Mask	0.0.0.255
Broadcast Address	192.168.1.255
First Usable Host	192.168.1.1

## 8.Difference Between IPv4 and IPv6

### IPV4

Address length is 32-bit

Format is Decimal

Example :192.168.1.1

Total Address is ~4.3 Billion

Type is Dotted-decimal

Security :Optimal IPSec

Broadcast is Supported

NAT Required : Yes

Auto Config :DHCP/manual

### IPV6

Address length is 128-Bit

Format is Hexadecimal

Example :2001:db8:85a3:8a2e:370:7334

Total Address is ~340 Undecillion

Type is Colon-Seperated

Security : IPSec Mandatory

Broadcast is Not Supported

NAT Required : No

Auto Config : SLAAC

# 9. Conclusion & Future Scope

## Conclusion

- The tool simplifies IP validation and subnetting.
- Demonstrates real-world Python networking application.

## Future Enhancements

- GUI-based interface
- Integration with database or network scanning tools



# THANK YOU