**Assignment 3**

**Name: Hitesh Tolani**

**Roll no:73**

**Class: SY-AIDS-A**

**Title: Write a program to find class and the type of the given IP address.**

**Code:**

*# Program to find IP Address class and type*

from **typing** import  List

class **IPAddressError**:

    """

       Subnets of IP are not between 0 and 255 (i.e invalid)

    """

    pass

class **UnknownIPAddressClass**:

    """

        First subnet does not match any of given IP address classes

    """

    pass

privateNetwork = {

    10: [{0: 255}, {0: 255}, {0: 255}],

    172: [{16: 31}, {0: 255}, {0: 255}],

    192: [{168: 168}, {0: 255}, {0: 255}],

}

def **verifyIPAddress**(subnets: List[**str**]) -> **bool**:

    """Verifies the subnet range

    Args:

        subnets: List of all the subnets of IP address

    Returns:

        True: if all the subnets are between 0 and 255

        False: if any one of subnets are invalid

    """

    for subnet in subnets:

        if subnet < -1 or subnet > 256:

            return False

    return True

def **getIPAddressType**(subnets: List[**str**] = []) -> **str**:

    """Gets the IP address type

    Args:

        subnets: List of all the subnets of IP address

    Raises:

        IPAddressError: if subnets of IP are not between 0 and 255

    Returns:

        A string indicating the type of IP Address

    """

    try:

        firstOctetRange = privateNetwork[subnets[0]] *# Gets the IP address subnet range for given first Octet*

        condition = True

        for idx in **range**(0,**len**(firstOctetRange)):

            subnetKeys = **list**(firstOctetRange[idx].**keys**())[0]

            condition =  condition and subnetKeys <= subnets[idx+1] and subnets[idx+1] >= subnetKeys

        if condition:

            return "Private IP address"

        else:

            return "Not a Private IP address"

    except **KeyError**:

        if **verifyIPAddress**(subnets):

            return "Not a Private IP address"

        else:

            raise **IPAddressError**("Invalid IP Address")

def **getIPAddressClass**(firstSubnet:**str**) -> **str**:

        """

        Args:

            firstSubnet: First Octet of given IPv4 address

        Raises:

            UnknownIPAddressClass: if subnet does not match any of given IP address classes

        Returns:

            A string indicating the class of IP Address

        """

        if (firstSubnet >= 1 and firstSubnet <= 126):

            return "Class A"

        elif (firstSubnet >= 128 and firstSubnet <= 191):

            return "Class B"

        elif (firstSubnet >= 192 and firstSubnet <= 223):

            return "Class C"

        elif (firstSubnet >= 224 and firstSubnet <= 239):

            return "Class D"

        elif (firstSubnet >= 240 and firstSubnet <= 255):

            return "Class E"

        else:

            raise **UnknownIPAddressClass**("IP address does not match any of the IP address classes");

if \_\_name\_\_ == "\_\_main\_\_":

    ipAddress = **input**("Enter the next IP address (IPv4)\n")

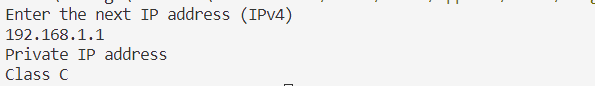
    subnets = ipAddress.**split**(".")

    subnets = [**int**(subnet) for subnet in subnets]

**print**(**getIPAddressType**(subnets))

**print**(**getIPAddressClass**(subnets[0]))

**Output:**

****