IPV4 ADDRESSING

CSE1004(NETWORK AND COMMUNICATION)LAB:L53-L54

APRIL 9, 2022

ANIRUDH VADERA
20BCE2940

DESCRIPTION:

Class of an IP Address

Network Address and Mask:

Network address – It identifies a network on internet. Using this, we can find range of addresses in the network and total possible number of hosts in the network.

Mask – It is a 32-bit binary number that gives the network address in the address block when AND operation is bitwise applied on the mask and any IP address of the block.

The default mask in different classes are:

Class A – 255.0.0.0

Class B – 255.255.0.0

Class C – 255.255.255.0

Example: Given IP address 132.6.17.85 and default class B mask, find the beginning address (network address).

Solution: The default mask is 255.255.0.0, which means that the only the first 2 bytes are preserved and the other 2 bytes are set to 0. Therefore, the network address is 132.6.0.0.

Subnetting: Dividing a large block of addresses into several contiguous sub-blocks and assigning these sub-blocks to different smaller networks is called subnetting. It is a practice that is widely used when classless addressing is done.

Classless Addressing:

To reduce the wastage of IP addresses in a block, we use sub-netting. What we do is that we use host id bits as net id bits of a classful IP address. We give the IP address and define the number of bits for mask along with it (usually followed by a '/' symbol), like, 192.168.1.1/28. Here, subnet mask is found by putting the given number of bits out of 32 as 1, like, in the given address, we need to put 28 out of 32 bits as 1 and the rest as 0, and so, the subnet mask would be 255.255.255.240.

Some values calculated in subnetting:

1. **Number of subnets: Given bits for mask – No. of bits in default mask**
2. **Subnet address: AND result of subnet mask and the given IP address**
3. **Broadcast address: By putting the host bits as 1 and retaining the network bits as in the IP address**
4. **Number of hosts per subnet: 2(32 – Given bits for mask) – 2**
5. **First Host ID: Subnet address + 1 (adding one to the binary representation of the subnet address)**
6. **Last Host ID: Subnet address + Number of Hosts**

Example: Given IP Address – 172.16.0.0/25, find the number of subnets and the number of hosts per subnet. Also, for the first subnet block, find the subnet address, first host ID, last host ID and broadcast address.

Solution: This is a class B address. So, no. of subnets = 2(25-16) = 29 = 512.

No. of hosts per subnet = 2(32-25) – 2 = 27 – 2 = 128 – 2 = 126

For the first subnet block, we have subnet address = 0.0, first host id = 0.1, last host id =

0.126 and broadcast address = 0.127

AIM:

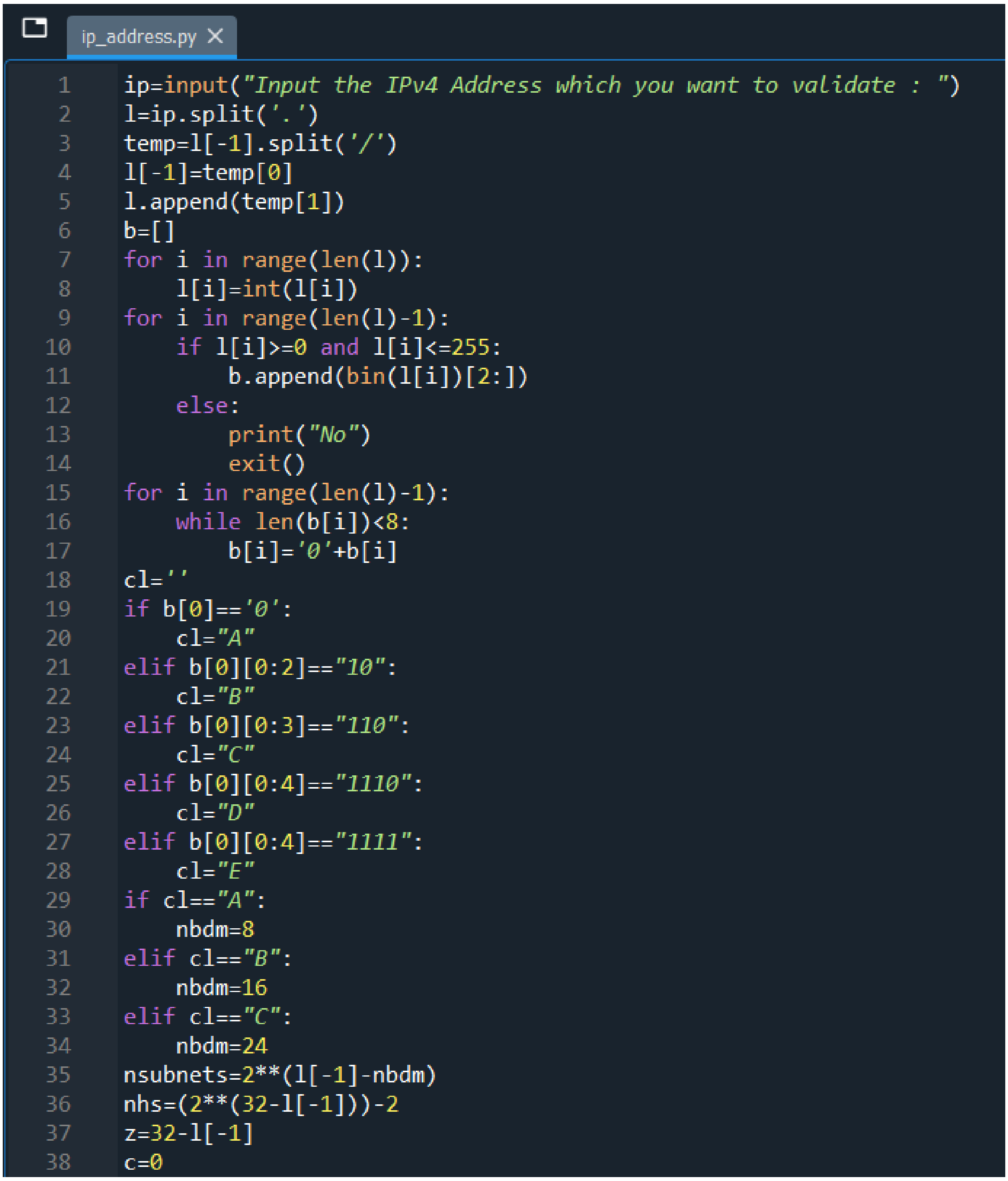
Validate the class of an IP Addressing Schemes using Python/JAVA Compiler.

SOURCE CODE:

```
ip=input("Input the IPv4 Address which you want to validate : ")
l=ip.split('.')
temp=l[-1].split('/')
l[-1]=temp[0]
l.append(temp[1])
b=[]
for i in range(len(l)):
   l[i]=int(l[i])
for i in range(len(l)-1):
   if l[i]>=0 and l[i]<=255:
      b.append(bin(l[i])[2:])
   else:
      print("No")
      exit()
for i in range(len(l)-1):
   while len(b[i])<8:
      b[i]='0'+b[i]
cl=''
if b[0]=='0':
   cl="A"
elif b[0][0:2]=="10":
   cl="B"
elif b[0][0:3]=="110":
   cl="C"
elif b[0][0:4]=="1110":
   cl="D"
elif b[0][0:4]=="1111":
   cl="E"
if cl=="A":
   nbdm=8
elif cl=="B":
   nbdm=16
elif cl=="C":
   nbdm=24
nsubnets=2**(l[-1]-nbdm)
nhs=(2**(32-l[-1]))-2
z=32-l[-1]
```

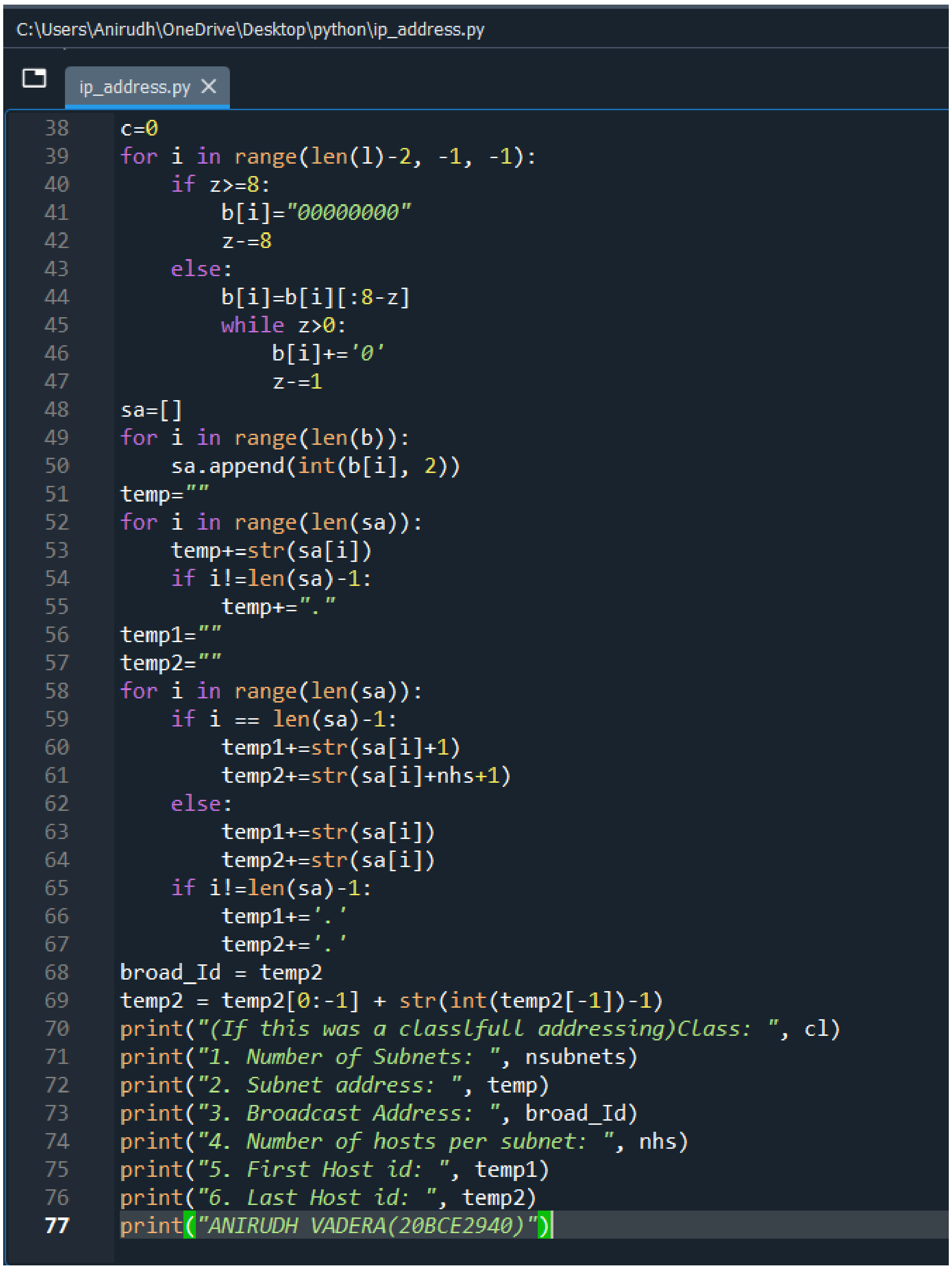
```
c=0
for i in range(len(l)-2, -1, -1):
    if z>=8:
        b[i]="00000000"
        z-=8
    else:
        b[i]=b[i][:8-z]
        while z>0:
            b[i]+='0'
            z-=1
sa=[]
for i in range(len(b)):
    sa.append(int(b[i], 2))
temp=""
for i in range(len(sa)):
    temp+=str(sa[i])
    if i!=len(sa)-1:
        temp+="."
temp1=""
temp2=""
for i in range(len(sa)):
    if i == len(sa)-1:
        temp1+=str(sa[i]+1)
        temp2+=str(sa[i]+nhs+1)
    else:
        temp1+=str(sa[i])
        temp2+=str(sa[i])
    if i!=len(sa)-1:
        temp1+='.'
        temp2+='.'
broad_Id = temp2
temp2 = temp2[0:-1] + str(int(temp2[-1])-1)
print("(If this was a classlfull addressing)Class: ", cl)
print("1. Number of Subnets: ", nsubnets)
print("2. Subnet address: ", temp)
print("3. Broadcast Address: ", broad_Id)
print("4. Number of hosts per subnet: ", nhs)
print("5. First Host id: ", temp1)
print("6. Last Host id: ", temp2)
```

CODE SNAPSHOTS:



ip_address.py

```python
ip=input("Input the IPv4 Address which you want to validate : ")
l=ip.split('.')
temp=l[-1].split('/')
l[-1]=temp[0]
l.append(temp[1])
b=[]
for i in range(len(l)):
    l[i]=int(l[i])
for i in range(len(l)-1):
    if l[i]>=0 and l[i]<=255:
        b.append(bin(l[i])[2:])
    else:
        print("No")
        exit()
for i in range(len(l)-1):
    while len(b[i])<8:
        b[i]='0'+b[i]
cl=''
if b[0]=='0':
    cl="A"
elif b[0][0:2]=="10":
    cl="B"
elif b[0][0:3]=="110":
    cl="C"
elif b[0][0:4]=="1110":
    cl="D"
elif b[0][0:4]=="1111":
    cl="E"
if cl=="A":
    nbdm=8
elif cl=="B":
    nbdm=16
elif cl=="C":
    nbdm=24
nsubnets=2**(l[-1]-nbdm)
nhs=(2**(32-l[-1]))-2
z=32-l[-1]
c=0
```

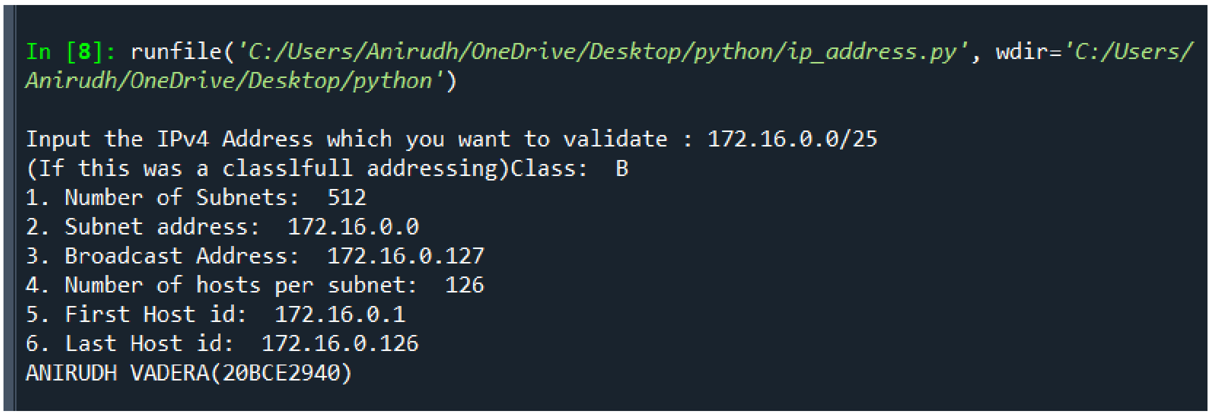


C:\Users\Anirudh\OneDrive\Desktop\python\ip_address.py

ip_address.py

```
c=0
for i in range(len(l)-2, -1, -1):
    if z>=8:
        b[i]="00000000"
        z-=8
    else:
        b[i]=b[i][:8-z]
        while z>0:
            b[i]+='0'
            z-=1
sa=[]
for i in range(len(b)):
    sa.append(int(b[i], 2))
temp=""
for i in range(len(sa)):
    temp+=str(sa[i])
    if i!=len(sa)-1:
        temp+="."
temp1=""
temp2=""
for i in range(len(sa)):
    if i == len(sa)-1:
        temp1+=str(sa[i]+1)
        temp2+=str(sa[i]+nhs+1)
    else:
        temp1+=str(sa[i])
        temp2+=str(sa[i])
    if i!=len(sa)-1:
        temp1+='.'
        temp2+='.'
broad_Id = temp2
temp2 = temp2[0:-1] + str(int(temp2[-1])-1)
print("(If this was a classlfull addressing)Class: ", cl)
print("1. Number of Subnets: ", nsubnets)
print("2. Subnet address: ", temp)
print("3. Broadcast Address: ", broad_Id)
print("4. Number of hosts per subnet: ", nhs)
print("5. First Host id: ", temp1)
print("6. Last Host id: ", temp2)
print("ANIRUDH VADERA(20BCE2940)")
```

OUTPUT:



```
In [8]: runfile('C:/Users/Anirudh/OneDrive/Desktop/python/ip_address.py', wdir='C:/Users/
Anirudh/OneDrive/Desktop/python')

Input the IPv4 Address which you want to validate : 172.16.0.0/25
(If this was a classlfull addressing)Class:  B
1. Number of Subnets:  512
2. Subnet address:  172.16.0.0
3. Broadcast Address:  172.16.0.127
4. Number of hosts per subnet:  126
5. First Host id:  172.16.0.1
6. Last Host id:  172.16.0.126
ANIRUDH VADERA(20BCE2940)
```

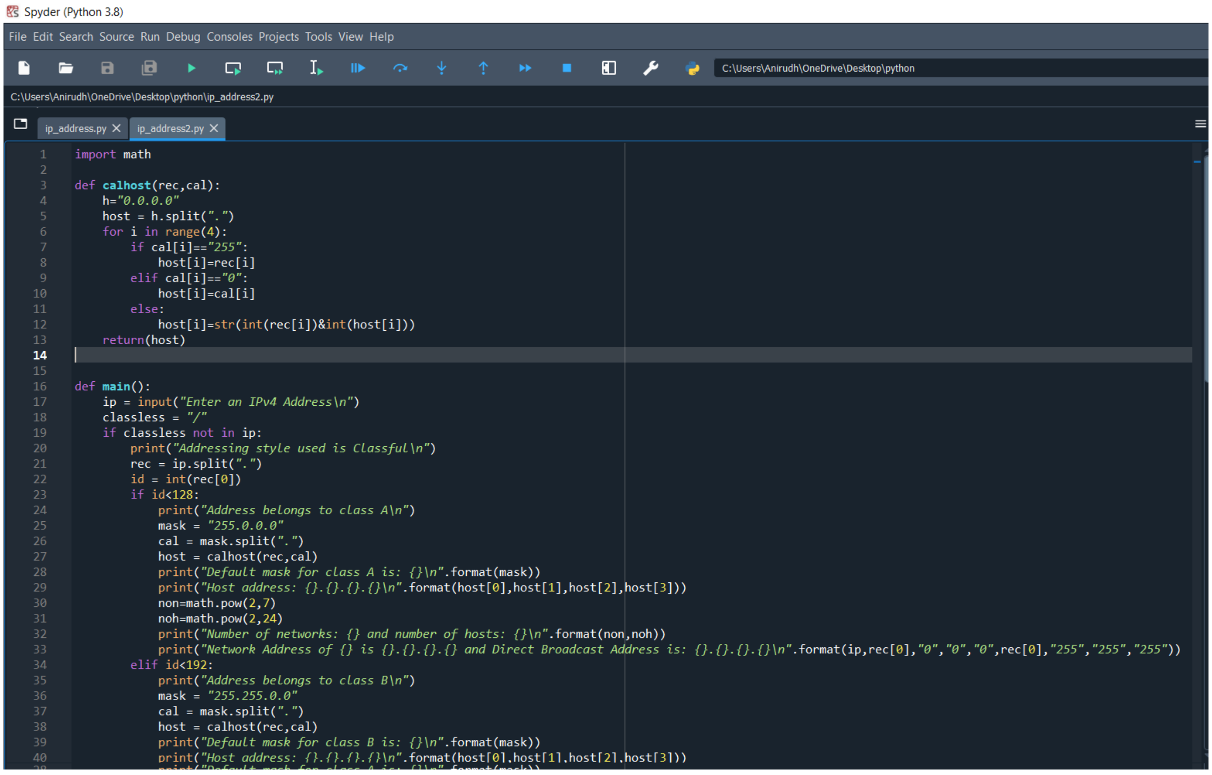
CLASSFULL ADDRESSING:

CODE:

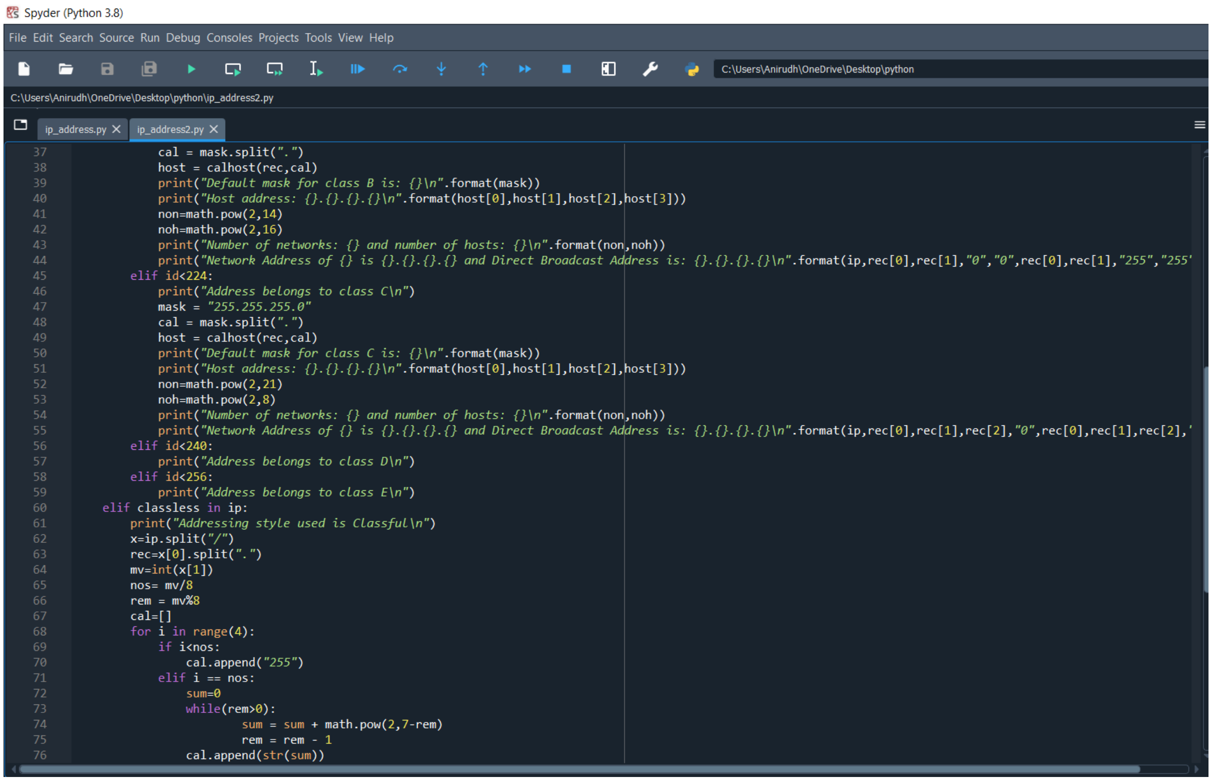
```
import math

def calhost(rec,cal):
    h="0.0.0.0"
    host = h.split(".")
    for i in range(4):
        if cal[i]=="255":
            host[i]=rec[i]
        elif cal[i]=="0":
            host[i]=cal[i]
        else:
            host[i]=str(int(rec[i])&int(host[i]))
    return(host)


def main():
    ip = input("Enter an IPv4 Address\n")
    classless = "/"
    if classless not in ip:
        print("Addressing style used is Classful\n")
        rec = ip.split(".")
        id = int(rec[0])
        if id<128:
            print("Address belongs to class A\n")
            mask = "255.0.0.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class A is: {}\n".format(mask))
            print("Host address:
{}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,7)
            noh=math.pow(2,24)
            print("Number of networks: {} and number of hosts:
{}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast
Address is:
{}.{}.{}.{}\n".format(ip,rec[0],"0","0","0",rec[0],"255","255","255"))
        elif id<192:
            print("Address belongs to class B\n")
            mask = "255.255.0.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class B is: {}\n".format(mask))
            print("Host address:
{}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,14)
            noh=math.pow(2,16)
            print("Number of networks: {} and number of hosts:
{}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast
Address is:
{}.{}.{}.{}\n".format(ip,rec[0],rec[1],"0","0",rec[0],rec[1],"255","255"))
        elif id<224:
            print("Address belongs to class C\n")
            mask = "255.255.255.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class C is: {}\n".format(mask))
            print("Host address:
{}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,21)
            noh=math.pow(2,8)
            print("Number of networks: {} and number of hosts:
{}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast
Address is:
{}.{}.{}.{}\n".format(ip,rec[0],rec[1],rec[2],"0",rec[0],rec[1],rec[2],"255"))
        elif id<240:
            print("Address belongs to class D\n")
        elif id<256:
            print("Address belongs to class E\n")
    elif classless in ip:
        print("Addressing style used is Classful\n")
        x=ip.split("/")
        rec=x[0].split(".")
        mv=int(x[1])
        nos= mv/8
        rem = mv%8
        cal=[]
        for i in range(4):
            if i<nos:
                cal.append("255")
            elif i == nos:
```

```
            sum=0
            while(rem>0):
                sum = sum + math.pow(2,7-rem)
                rem = rem - 1
            cal.append(str(sum))
          else:
            cal.append("0")
        host = calhost(rec,cal)
        print("Mask is: {}.{}.{}.{}\n".format(cal[0],cal[1],cal[2],cal[3]))
        print("Host address:
{}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
        id = int(rec[0])
        if id<128:
          print("Address belongs to class A\n")
          nor=8
        elif id<192:
          print("Address belongs to class B\n")
          nor=16;
        elif id<224:
          print("Address belongs to class C\n")
          nor=24;
        elif id<240:
          print("Address belongs to class D\n")
        elif id<256:
          print("Address belongs to class E\n")
        nos=math.pow(2,(mv-nor))
        noh=math.pow(2,(32-mv))-2
        print("Number of subnets: {} and number of hosts per subnet:
{}\n".format(nos,noh))

if __name__ == "__main__":
    print("ANIRUDH VADERA(20BCE2940)")
    main()
```

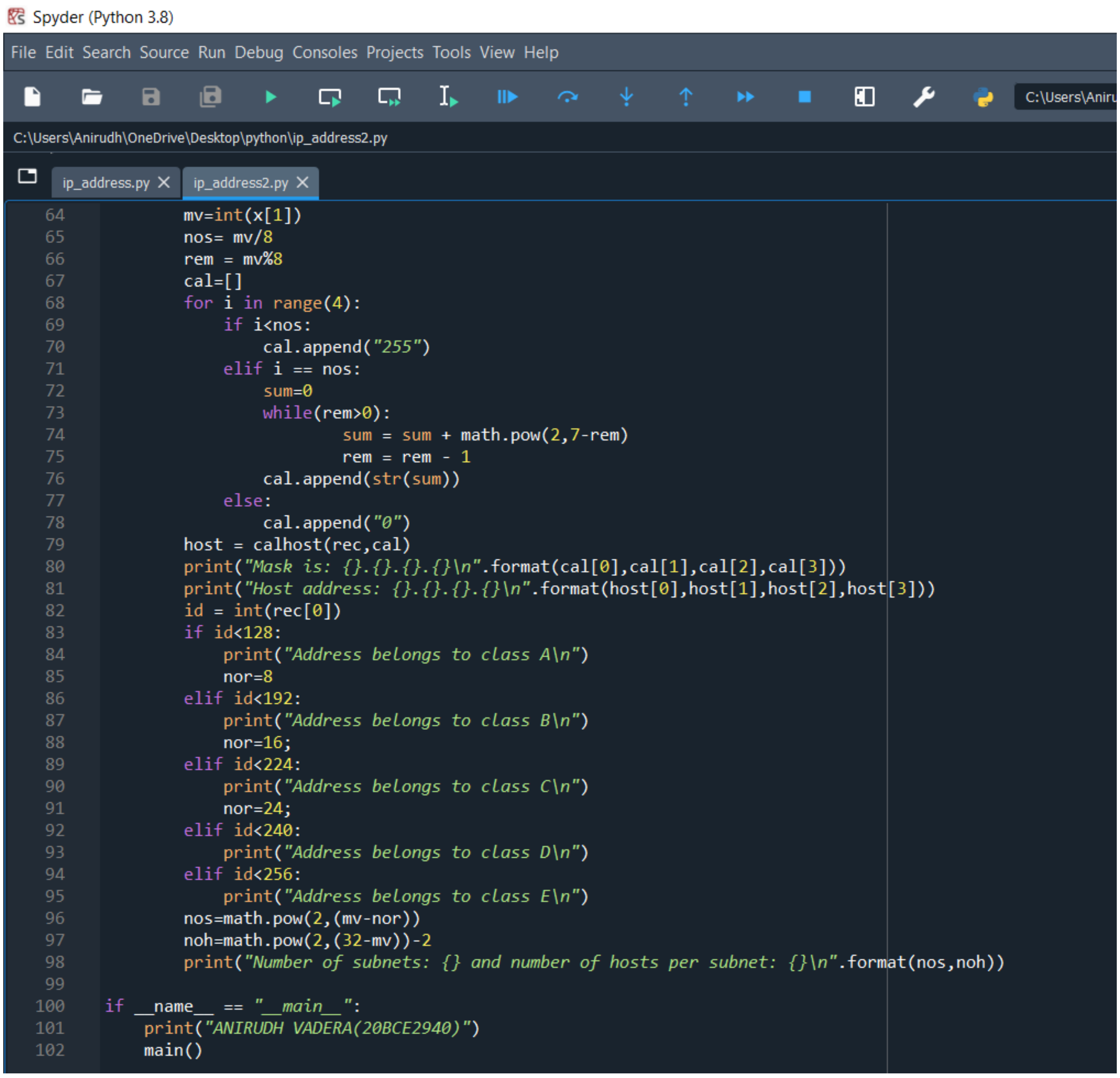
CODE SNAPSHOTS:


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Anirudh\OneDrive\Desktop\python
C:\Users\Anirudh\OneDrive\Desktop\python\ip_address2.py
ip_address.py
ip_address2.py

```python
import math

def calhost(rec,cal):
    h="0.0.0.0"
    host = h.split(".")
    for i in range(4):
        if cal[i]=="255":
            host[i]=rec[i]
        elif cal[i]=="0":
            host[i]=cal[i]
        else:
            host[i]=str(int(rec[i])&int(host[i]))
    return(host)


def main():
    ip = input("Enter an IPv4 Address\n")
    classless = "/"
    if classless not in ip:
        print("Addressing style used is Classful\n")
        rec = ip.split(".")
        id = int(rec[0])
        if id<128:
            print("Address belongs to class A\n")
            mask = "255.0.0.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class A is: {}\n".format(mask))
            print("Host address: {}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,7)
            noh=math.pow(2,24)
            print("Number of networks: {} and number of hosts: {}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast Address is: {}.{}.{}.{}\n".format(ip,rec[0],"0","0","0",rec[0],"255","255","255"))
        elif id<192:
            print("Address belongs to class B\n")
            mask = "255.255.0.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class B is: {}\n".format(mask))
            print("Host address: {}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
```

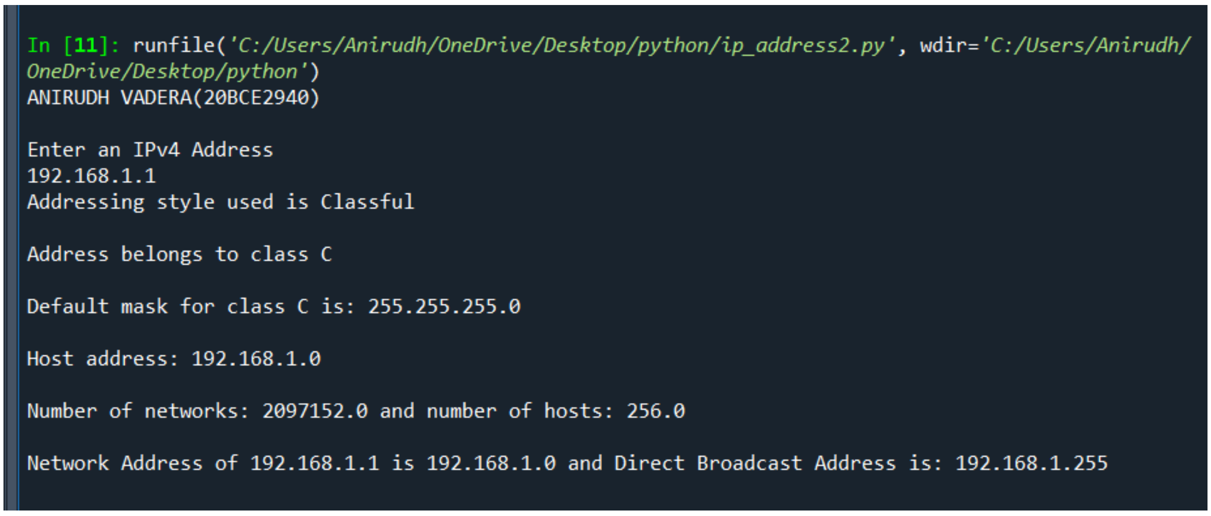

Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Anirudh\OneDrive\Desktop\python
C:\Users\Anirudh\OneDrive\Desktop\python\ip_address2.py
ip_address.py
ip_address2.py

```python
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class B is: {}\n".format(mask))
            print("Host address: {}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,14)
            noh=math.pow(2,16)
            print("Number of networks: {} and number of hosts: {}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast Address is: {}.{}.{}.{}\n".format(ip,rec[0],rec[1],"0","0",rec[0],rec[1],"255","255'
        elif id<224:
            print("Address belongs to class C\n")
            mask = "255.255.255.0"
            cal = mask.split(".")
            host = calhost(rec,cal)
            print("Default mask for class C is: {}\n".format(mask))
            print("Host address: {}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
            non=math.pow(2,21)
            noh=math.pow(2,8)
            print("Number of networks: {} and number of hosts: {}\n".format(non,noh))
            print("Network Address of {} is {}.{}.{}.{} and Direct Broadcast Address is: {}.{}.{}.{}\n".format(ip,rec[0],rec[1],rec[2],"0",rec[0],rec[1],rec[2],'
        elif id<240:
            print("Address belongs to class D\n")
        elif id<256:
            print("Address belongs to class E\n")
    elif classless in ip:
        print("Addressing style used is Classful\n")
        x=ip.split("/")
        rec=x[0].split(".")
        mv=int(x[1])
        nos= mv/8
        rem = mv%8
        cal=[]
        for i in range(4):
            if i<nos:
                cal.append("255")
            elif i == nos:
                sum=0
                while(rem>0):
                        sum = sum + math.pow(2,7-rem)
                        rem = rem - 1
                cal.append(str(sum))
```


Spyder (Python 3.8)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Aniru
C:\Users\Anirudh\OneDrive\Desktop\python\ip_address2.py
ip_address.py
ip_address2.py

```
        mv=int(x[1])
        nos= mv/8
        rem = mv%8
        cal=[]
        for i in range(4):
            if i<nos:
                cal.append("255")
            elif i == nos:
                sum=0
                while(rem>0):
                        sum = sum + math.pow(2,7-rem)
                        rem = rem - 1
                cal.append(str(sum))
            else:
                cal.append("0")
        host = calhost(rec,cal)
        print("Mask is: {}.{}.{}.{}\n".format(cal[0],cal[1],cal[2],cal[3]))
        print("Host address: {}.{}.{}.{}\n".format(host[0],host[1],host[2],host[3]))
        id = int(rec[0])
        if id<128:
            print("Address belongs to class A\n")
            nor=8
        elif id<192:
            print("Address belongs to class B\n")
            nor=16;
        elif id<224:
            print("Address belongs to class C\n")
            nor=24;
        elif id<240:
            print("Address belongs to class D\n")
        elif id<256:
            print("Address belongs to class E\n")
        nos=math.pow(2,(mv-nor))
        noh=math.pow(2,(32-mv))-2
        print("Number of subnets: {} and number of hosts per subnet: {}\n".format(nos,noh))

if __name__ == "__main__":
    print("ANIRUDH VADERA(20BCE2940)")
    main()
```

OUTPUT:



```
In [11]: runfile('C:/Users/Anirudh/OneDrive/Desktop/python/ip_address2.py', wdir='C:/Users/Anirudh/
OneDrive/Desktop/python')
ANIRUDH VADERA(20BCE2940)

Enter an IPv4 Address
192.168.1.1
Addressing style used is Classful

Address belongs to class C

Default mask for class C is: 255.255.255.0

Host address: 192.168.1.0

Number of networks: 2097152.0 and number of hosts: 256.0

Network Address of 192.168.1.1 is 192.168.1.0 and Direct Broadcast Address is: 192.168.1.255
```