CNL ASSIGNMENT 3 (Subnetting) (31139-Durvesh)

//Write-up

Assignment No. 03

- Problem Statement : Write a program to demonstrate subnetting and find subnet mask.
- Objective : To understand the concept of subnetting and find subnet mask.
- Outcome : The output will be the subnet mask of given IP address of host.
- S/W & H/W Requirements :
  Fedora 20/Windows 10, Eclipse IDE 1GB RAM
- Theory :

  Subnetting :

  A subnet is a logical sub division of an IP network. The practice of dividing a network into 2 or more networks is called subnetting.

  Computers that belong to subnet are addressed with a common identical, most significant bit group in their IP address into 2 fields, a network or routing prefix a network and field or host-identifier.

  Subnet mask - It is a mask used to determine what subnet IP address belong to. Eg. IP address has 2 components, the network address and host address.

It is called a subnet mask because it is used to identify network address of IP by performing bitwise and operation on net mask.

Subnet mask is a 32 bit number that makes an IP address and divides the IP into network and host address.

Packet:

A network packet is a formatted unit of data carried by a packet switched network. When data is formatted into packets and packet switching is employed, the bandwidth of the communication medium can be better shared among users than with circuit switching.

A network is a 32-bit mask used to divide an IP address into subnets and specify the network's available hosts.

Netmask: It is a 32 bit mask used to divide an IP address into subnets and specify the network available host.

Class	Network length	Netmask
A	8	255.0.0.0
B	16	255.255.0.0
C	24	255.255.255.0

Q. No. | Q. No.

Management Packets: These are used by peer WLAN controller to maintain WLAN network.

They are divided by peer WLAN controller to maintain a WLAN network and as such in seldom importance above OSI layer 2

Receive the data

The UDP open node opens a local UDP socket on the port or service name you write to UDP. It also opens on the receiver side, the matched port or service name you write to form sender.

Use Cases:

1) Enter IP : 192.168.1.1
Binary Format: 11000000.10101000.00000001.00000001
It is a Class-C address
Default mask: 255.255.255.0
Binary Mask: 11111111.11111111.11111111.00000000
Enter the no. of addresses in each subnet: 14
Network Address:
11000000.10101000.00000001.00000000 → 192.168.1.0
:
11000000.10101000.00000001.00001111 → 192.168.1.15
Broadcast Address
Subnet mask : 255.255.255.252

Q. No.

2) Enter IP : 145.133.12.45
Binary format : 10010001.10000101.00001100.00101101
It is a class-B address
Default mask : 255.255.0.0
Binary Mask : 11111111.11111111.00000000.00000000
Enter the number of addresses in each subnet : 4
Subnet mask : 255.255.255.252

Network Address :
10010001.10000101.00001100.00101100 → 145.133.12.44
⋮ ⋮
10010001.10000101.00001100.00101111 → 145.133.12.47
Broadcast Address

Conclusion : We successfully learnt subnetting and its concepts while implementing the same in a program.

//SAMPLE CODE

```
from math import *


def IP2bin(ip):
    ipsplits = ip.split('.')
    binstr = ""
    for num in ipsplits:
        binstr += dec2binstr(num)
    return binstr
def dec2binstr(dec):
    binstr = bin(int(dec)).replace("0b","")
    additionalbs = 8-len(binstr)
    return "0"*additionalbs+binstr


IP = input("Enter IP: ")
IPsplits = IP.split('.')
print("Binary Format: ",end="")
MainIP = IP2bin(IP)
print(MainIP)
classn = int(IPsplits[0])
if classn<=127 :
    classip = "A"
elif classn<=191:
    classip = "B"
elif classn<=223:
    classip= "C"
```

```
elif classn<=239:
   classip = "D"
else:
   classip = "E"
print("It is a Class-"+classip+" address")
classiptonum = {"A":1,"B":2,"C":3,"D":4,"E":5}
classnum = classiptonum[classip]
DefaultMask = ("255."*classnum+"0."*(4-classnum))[:-1]
print("Default Mask: "+DefaultMask)
print("Binary Mask: ",end="")
DefaultMaskbin = IP2bin(DefaultMask)
print(DefaultMaskbin)
naddr = int(input("Enter the Number of addresses in each subnet: "))
host_bits = ceil(log(naddr)/log(2))
additional_bits = (8*(4-classnum))-host_bits
subnet_mask = DefaultMaskbin[:classnum*8]
subnet_mask+="1"*additional_bits
subnet_mask += "0"*host_bits
splits = [subnet_mask[i:i+8] for i in range(0, len(subnet_mask), 8)]
subnetmaskip = str(".".join(splits))
print("Subnet Mask: "+str(subnetmaskip))
decsubnet = ""
for sp in splits:
   decsubnet+=str(int(sp,2))+"."
decsubnet = decsubnet[:-1]
print("Subnet Mask (in decimals): "+decsubnet)
```

```
print("Number of subnets: "+str(2**additional_bits))
print("Number of hosts in each subnet: "+str(2**host_bits-2))

## host bit number of LSBs should be 0 for network address and 0 for broadcast smthng

tempstr = MainIP[:-host_bits]

NetworkAddress = tempstr + "0"*host_bits
broadAddress = tempstr + "1"*host_bits

netsplits = [NetworkAddress[i:i+8] for i in range(0, len(NetworkAddress), 8)]
netaddr = str(".".join(netsplits))
decnetaddr = ""
for sp in netsplits:
   decnetaddr+=str(int(sp,2))+"."
decnetaddr = decnetaddr[:-1]

broadsplits = [broadAddress[i:i+8] for i in range(0, len(broadAddress), 8)]
broadaddr = str(".".join(broadsplits))
decbroadaddr = ""
for sp in broadsplits:
   decbroadaddr+=str(int(sp,2))+"."
decbroadaddr = decbroadaddr[:-1]

print("Network Address:")
print(str(netaddr)+" -> "+decnetaddr)
```

```
print(":       :")
print(":       :")
print(str(broadaddr)+" -> "+decbroadaddr)
print("Broadcast Address")
```

//OUTPUTS