**22BCE0476**

**AMAN CHAUHAN**

****

BCSE308P - Computer Networks Lab

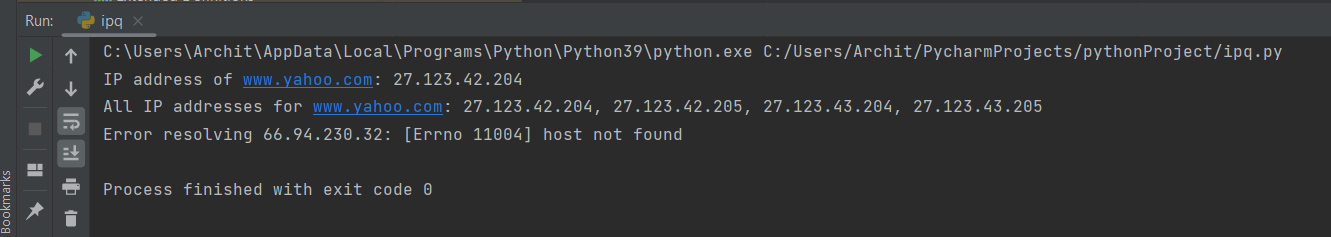
LAB REPORT

**IP Error:**

**Code:**

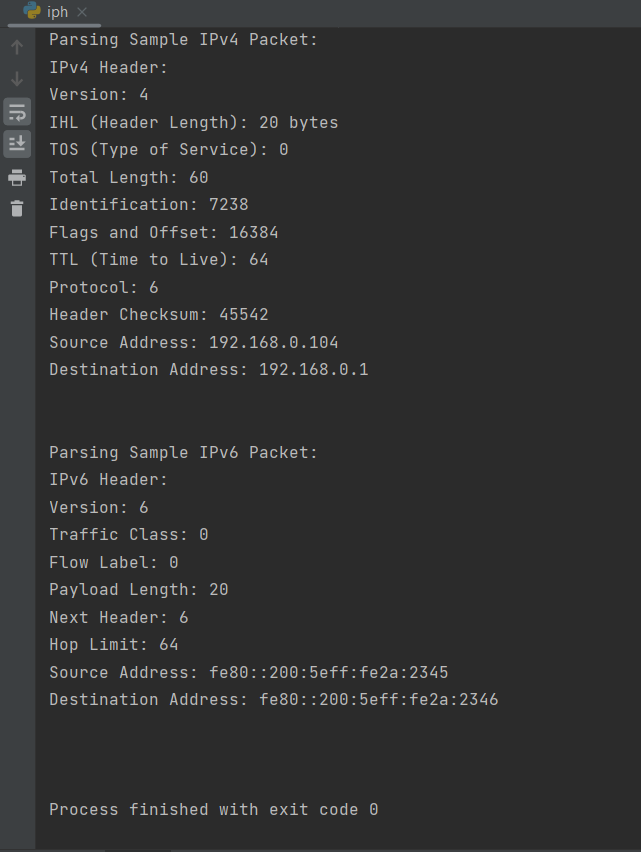
import socket  
  
  
def get\_ip\_from\_hostname(hostname):  
 try:  
 # Get the IP address of a given hostname  
 ip\_address = socket.gethostbyname(hostname)  
 print(f"IP address of {hostname}: {ip\_address}")  
  
 # Get all IP addresses associated with the hostname (if multiple exist)  
 ip\_addresses = socket.gethostbyname\_ex(hostname)[2]  
 print(f"All IP addresses for {hostname}: {', '.join(ip\_addresses)}")  
 except socket.gaierror as e:  
 print(f"Error resolving {hostname}: {e}")  
  
  
def get\_hostname\_from\_ip(ip\_address):  
 try:  
 # Get the hostname for a given IP address  
 hostname = socket.gethostbyaddr(ip\_address)[0]  
 print(f"Hostname for IP {ip\_address}: {hostname}")  
 except socket.herror as e:  
 print(f"Error resolving {ip\_address}: {e}")  
  
  
# Example usage  
hostname = "www.yahoo.com"  
get\_ip\_from\_hostname(hostname)  
  
ip\_address = "66.94.230.32"  
get\_hostname\_from\_ip(ip\_address)

Output:



IP Header:

import socket  
import struct  
  
# Sample IPv4 packet (20-byte header with sample values)  
# Version (4) + IHL (5), TOS, Total Length, ID, Flags + Fragment Offset, TTL, Protocol (TCP), Header Checksum, Source IP, Destination IP  
ipv4\_sample\_packet = b'\x45\x00\x00\x3c\x1c\x46\x40\x00\x40\x06\xb1\xe6\xc0\xa8\x00\x68\xc0\xa8\x00\x01'  
  
# Sample IPv6 packet (40-byte header with sample values)  
# Version (6) + Traffic Class + Flow Label, Payload Length, Next Header (TCP), Hop Limit, Source IP (16 bytes), Destination IP (16 bytes)  
ipv6\_sample\_packet = (  
 b'\x60\x00\x00\x00\x00\x14\x06\x40' +  
 b'\xfe\x80\x00\x00\x00\x00\x00\x00\x02\x00\x5e\xff\xfe\x2a\x23\x45' +  
 b'\xfe\x80\x00\x00\x00\x00\x00\x00\x02\x00\x5e\xff\xfe\x2a\x23\x46'  
)  
  
def parse\_ipv4\_header(packet):  
 # Unpack the first 20 bytes of the packet for IPv4  
 ipv4\_header = struct.unpack('!BBHHHBBH4s4s', packet[:20])  
  
 # Extract information from the unpacked header  
 version\_ihl = ipv4\_header[0]  
 version = version\_ihl >> 4  
 ihl = version\_ihl & 0xF  
 tos = ipv4\_header[1]  
 total\_length = ipv4\_header[2]  
 identification = ipv4\_header[3]  
 flags\_offset = ipv4\_header[4]  
 ttl = ipv4\_header[5]  
 protocol = ipv4\_header[6]  
 checksum = ipv4\_header[7]  
 source\_address = socket.inet\_ntoa(ipv4\_header[8])  
 destination\_address = socket.inet\_ntoa(ipv4\_header[9])  
  
 print("IPv4 Header:")  
 print(f"Version: {version}")  
 print(f"IHL (Header Length): {ihl \* 4} bytes")  
 print(f"TOS (Type of Service): {tos}")  
 print(f"Total Length: {total\_length}")  
 print(f"Identification: {identification}")  
 print(f"Flags and Offset: {flags\_offset}")  
 print(f"TTL (Time to Live): {ttl}")  
 print(f"Protocol: {protocol}")  
 print(f"Header Checksum: {checksum}")  
 print(f"Source Address: {source\_address}")  
 print(f"Destination Address: {destination\_address}")  
 print("\n")  
  
def parse\_ipv6\_header(packet):  
 # Unpack the first 40 bytes of the packet for IPv6  
 ipv6\_header = struct.unpack('!IHBB16s16s', packet[:40])  
  
 # Extract information from the unpacked header  
 version\_traffic\_flow = ipv6\_header[0]  
 version = version\_traffic\_flow >> 28  
 traffic\_class = (version\_traffic\_flow >> 20) & 0xFF  
 flow\_label = version\_traffic\_flow & 0xFFFFF  
 payload\_length = ipv6\_header[1]  
 next\_header = ipv6\_header[2]  
 hop\_limit = ipv6\_header[3]  
 source\_address = socket.inet\_ntop(socket.AF\_INET6, ipv6\_header[4])  
 destination\_address = socket.inet\_ntop(socket.AF\_INET6, ipv6\_header[5])  
  
 print("IPv6 Header:")  
 print(f"Version: {version}")  
 print(f"Traffic Class: {traffic\_class}")  
 print(f"Flow Label: {flow\_label}")  
 print(f"Payload Length: {payload\_length}")  
 print(f"Next Header: {next\_header}")  
 print(f"Hop Limit: {hop\_limit}")  
 print(f"Source Address: {source\_address}")  
 print(f"Destination Address: {destination\_address}")  
 print("\n")  
  
# Parse the sample packets  
print("Parsing Sample IPv4 Packet:")  
parse\_ipv4\_header(ipv4\_sample\_packet)  
  
print("Parsing Sample IPv6 Packet:")  
parse\_ipv6\_header(ipv6\_sample\_packet)



SLACC Configuration and EUI-68:

