Python经典网络协议 第一天作业

抄工具代码

代码一: IP排序 特殊的排序方案

import ipaddress  
  
ip\_list = [  
 '172.16.12.123',  
 '172.16.12.3',  
 '172.16.12.234',  
 '172.16.12.12',  
 '172.16.12.23',  
 '10.1.2.3',  
 '192.168.2.3',  
 '172.16.1.2'  
  
]  
  
def sort\_ip(ips):  
 return sorted(ips,key=lambda ip:ipaddress.ip\_address(ip))  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print(sort\_ip(ip\_list))

代码二: IPv6工具 了解特殊的IPv6地址

import re  
import ipaddress  
  
  
def full\_ipv6(ipv6):  
 ipv6\_section = ipv6.split(':')  
 ipv6\_section\_len = len(ipv6.split(':'))  
 if ipv6\_section.index(''):  
 null\_location = ipv6\_section.index('')  
 ipv6\_section.pop(null\_location)  
 add\_section = 8 - ipv6\_section\_len + 1  
 for x in range(add\_section):  
 ipv6\_section.insert(null\_location, "0000")  
 new\_ipv6 = []  
 for s in ipv6\_section:  
 if len(s) < 4:  
 new\_ipv6.append((4 - len(s)) \* '0' + s)  
 else:  
 new\_ipv6.append(s)  
 return ':'.join(new\_ipv6)  
 else:  
 return ipv6  
  
  
def full\_ipv6(ipv6):  
 return ipaddress.ip\_address(ipv6).exploded  
  
  
def solicited\_node\_multicast\_address(ipv6):  
 return 'FF02::1:FF' + full\_ipv6(ipv6)[-7:]  
  
  
def mac\_to\_ipv6\_linklocal(mac):  
 mac\_value = int(re.sub('[ :.-]', '', mac), 16)  
 high2 = mac\_value >> 32 & 0xffff ^ 0x0200  
 high1 = mac\_value >> 24 & 0xff  
 low1 = mac\_value & 0xff  
 low2 = mac\_value & 0xffff  
 return 'fe80::{:04x}:{:02x}ff:fe{:02x}:{:04x}'.format(high2, high1, low1, low2)  
  
  
def ipv6\_to\_mac(ipv6):  
 ipv6\_address = full\_ipv6(ipv6)  
 last\_4\_sections = ipv6\_address.split(':')[-4:]  
 mac\_1 = int(last\_4\_sections[0][:2], 16) ^ 0x02  
 mac\_2 = int(last\_4\_sections[0][2:], 16)  
 mac\_3 = int(last\_4\_sections[1][:2], 16)  
 mac\_4 = int(last\_4\_sections[2][2:], 16)  
 mac\_5 = int(last\_4\_sections[3][:2], 16)  
 mac\_6 = int(last\_4\_sections[3][2:], 16)  
 return '{:02x}:{:02x}:{:02x}:{:02x}:{:02x}:{:02x}'.format(mac\_1, mac\_2, mac\_3, mac\_4, mac\_5, mac\_6)  
  
  
def mac\_to\_eui64(mac, prefix):  
 mac\_value = int(re.sub('[ :.-]', ',mac'), 16)  
 high2 = mac\_value >>32 & 0xffff ^ 0x0200  
 high1 = mac\_value >> 24& 0xff  
 low1 = mac\_value >> 16& 0xff  
 low2 = mac\_value & 0xff  
 host\_id = '{:04x}:{:02x}ff:fe{:02x}:{:04x}'.format(high2, high1, low1, low2)  
 net = prefix.split("/")[0]  
 return net+ host\_id  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print(full\_ipv6('2001::f107:94ac:2717:a736'))  
 print(mac\_to\_eui64(mac = '00:50:56:ab:4d:19',prefix='2001::f107:94ac:2717:a736'))  
 print(mac\_to\_ipv6\_linklocal(mac = '00:50:56:ab:4d:19'))  
 print(solicited\_node\_multicast\_address('2001::f107:94ac:2717:a736'))

代码三: 通过注册表 获取WIN下的唯一码

import netifaces as ni  
  
  
def get\_connection\_name\_from\_guid(iface\_guids):  
 wr = \_\_import\_\_('winreg', globals(), locals(), ['wr'])  
 iface\_dict = {}  
 reg = wr.ConnectRegistry(None, wr.HKEY\_LOCAL\_MACHINE)  
 print(reg)  
 reg\_key = wr.OpenKey(reg, r'SYSTEM\CurrentControlSet\Control\Network\{07374750-E68B-490E-9330-9FD785CD71B6}')  
 print(reg\_key)  
 for i in iface\_guids:  
 try:  
 reg\_subkey = wr.OpenKey(reg\_key, i + r'\Connection')  
 iface\_dict[wr.QueryValueEx(reg\_subkey, 'Name')[0]] = i  
 except FileNotFoundError:  
 pass  
  
 return iface\_dict  
  
  
def win\_from\_name\_id(ifname):  
 x = ni.interfaces()  
 return get\_connection\_name\_from\_guid(x).get(ifname)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 import platform  
  
 if platform.system() == 'Windows':  
 print(win\_from\_name\_id('Net1'))

代码四: 获取WIN和LINUX的IP地址

from netifaces import interfaces,ifaddresses,AF\_INET,AF\_INET6  
  
import platform  
  
  
def get\_ip\_address(ifname):  
 if platform.system() == 'Linux':  
 try:  
 return ifaddresses(ifname)[AF\_INET][0]['addr']  
 except ValueError:  
 return None  
 elif platform.system() == 'Windows':  
 from win\_ifname import win\_from\_name\_get\_id  
 if\_id = win\_from\_name\_get\_id(ifname)  
 if not if\_id:  
 return  
 else:  
 return ifaddresses(if\_id)[AF\_INET][0]['addr']  
 else:  
 print('操作系统不支持，本脚本只能工作在Windows或者Linux环境')  
def get\_ipv6\_address(ifname):  
 if platform.system() == 'Linux':  
 try:  
 return ifaddresses(ifname)[AF\_INET6][0]['addr']  
 except ValueError:  
 return None  
 elif platform.system() == 'Windows':  
 from win\_ifname import win\_from\_name\_get\_id  
 if\_id = win\_from\_name\_get\_id(ifname)  
 if not if\_id:  
 return  
 else:  
 return ifaddresses(if\_id)[AF\_INET6][0]['addr']  
  
 else:  
 print('操作系统不支持，本脚本只能工作在Windows或者Linux环境')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 if platform.system() == 'Linux':  
 print(get\_ip\_address('ens33'))  
 print(get\_ipv6\_address('ens33'))  
 elif platform.system() == 'Windows':  
 print(get\_ip\_address('Net1'))  
 print(get\_ipv6\_address('Net1'))

代码五: 获取WIN和LINUX的MAC地址

import netifaces  
import platform  
import pprint  
  
pp = pprint.PrettyPrinter(indent=4)  
  
def get\_mac\_address(ifname):  
 if platform.system() == "Linux":  
 try:  
 return netifaces.ifaddresses(ifname)[netifaces.AF\_LINK][0]['addr']  
 except ValueError:  
 return None  
 elif platform.system() == 'Windows':  
 from win\_ifname import win\_from\_name\_get\_id  
 if\_id = win\_from\_name\_get\_id(ifname)  
 if not if\_id:  
 return None  
 else:  
 return netifaces.ipaddresses(if\_id)[netifaces.AF\_LINK][0]['addr']  
 else:  
 print('操作系统不支持，本脚本只能工作在Windows或者Linux环境')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 import platform  
 if platform.system() == 'Linux':  
 print(get\_mac\_address('ens33'))  
 elif platform.system() == 'Windows':  
 print(get\_mac\_address('Net1'))