import binascii    *# Conversiones binarias*

import ipaddress   *# Manejo de direcciones IP y redes*

import socket      *# Operaciones de red de bajo nivel*

*#Operadores Aritmeticos*

*"""prioridad\_sw1=4096*

*prioridad\_sw2=32768*

*SW1\_IP="192.168.10.100"*

*SW2\_IP="192.168.11.200"*

*print(f"Suma para aumentar la prioridad de SW1 Brigde ID SW1: {prioridad\_sw1+4096}")*

*print(f"Resta para convertir a SW2 en root bridge Root ID SW2: {prioridad\_sw2-4096}" )*

*print(f"Multiplicacion ICMP de   a 192.168.11.200 x4:")*

*for contador\_icmp in range(4):*

*print(f"Ping de {SW1\_IP} a {SW2\_IP}")*

*print(f"Valor decimal de 8 bits es: {2\*\*8} ")*

*print(f"Valores de 8 bits son: 0 a {2\*\*8-1}")*

*#Operadores de comparacion*

*MAC\_SW1="aa.04.23.45.56.36"*

*MAC\_SW2="45.36.89.ff.ab.45"*

*if (MAC\_SW2==MAC\_SW1):*

*print(f"Mac SW1 es igual Mac SW2")*

*else:*

*print("No son iguales Mac SW1 con Mac SW2")*

*if (MAC\_SW2!=MAC\_SW1):*

*print("No son iguales Mac SW1 con Mac SW2")*

*else:*

*print(f"Mac SW1 es igual Mac SW2")*

*if(prioridad\_sw2>prioridad\_sw1):*

*print("Prioridad SW2  tiene mayor que SW1")*

*else:*

*print("Prioridad SW1  tiene mayor que SW2")*

*#Operadores Logicos*

*#AND  TRUE TRUE los dos deben ser true*

*#OR   TRUE solo uno debe ser*

*#NOT  AL TRUE lo vuelve FALSE y FALSE lo vuelve TRUE*

*if((MAC\_SW1>MAC\_SW2) and (prioridad\_sw1<prioridad\_sw2)):*

*print("TRUE")*

*else:*

*print("FALSE")*

*#Operadores de asignacion*

*SVI\_VLAN10="192.168.10.5"*

*print(f"IP INT VLAN10= {SVI\_VLAN10}")*

*loop=0*

*while(loop<=10):*

*print(loop)*

*loop+=1*

*#Operadores de Identidad*

*#MAC\_SW2=MAC\_SW1*

*print(f"MAC\_SW1 no es igual MAC\_SW2: {MAC\_SW1 is not MAC\_SW2}")*

*#Operadores de pertenecia*

*print(f"'aa' pertenece a la MAC\_SW1 {'aa' in MAC\_SW1}")*

*print(f"'00' No pertenece a la MAC\_SW1 {'00' not  in   MAC\_SW1}")*

*"""*

*#Operadores de Bit*

*"""ip = ipaddress.IPv4Address('192.168.10.240')*

*ip +=5*

*print(ip)*

*red=ipaddress.IPv4Network('192.168.10.0/24')*

*print(f"Esta es l mascara de red: {red.netmask}")*

*print(f"Esta es la ip de broadcast: {red.broadcast\_address}")*

*print(f"Esta es la ip de red: {red.network\_address}")*

*"""*

*"""ip=ipaddress.IPv4Address('192.168.10.10')*

*ip\_int=int(ip)*

*#Obtenemos la IP de red:*

*mascara=0xFFFFFF00 #/24*

*red\_resultante=ip\_int & mascara*

*print(ipaddress.IPv4Address(red\_resultante))"""*

valorb1=0b1100

valorb2=0b1010

*#Operador and*

print(bin(valorb1&valorb2))

*#Operadpr or*

print(bin(valorb1| valorb2))

*#Operador xor*

print(bin(valorb1))