```
from time import sleep
import snap7
from snap7.util import *
import struct
class output(object):
	bool=1
	int=2
	real=3
	word=4
	dword=5

class S71200():
	def __init__(self,ip,debug=False):
		self.debug = debug
		self.plc = snap7.client.Client()
		self.plc.connect(ip,0,1)
		self.ip = ip
	def getMem(self,mem,returnByte=False):
		area=0x83
		length=1
		type=0
		out=None
		bit=0
		start=0
		if(mem[0].lower()=='m'):
			area=0x83
		if(mem[0].lower()=='q'):
			area=0x82
		if(mem[0].lower()=='i'):
			area=0x81
		if(mem[1].lower()=='x'): #bit
			length=1
			out=output().bool
			start = int(mem.split('.')[0][2:])
		if(mem[1].lower()=='b'): #byte
			length=1
			out=output().int
			start = int(mem[2:])
		if(mem[1].lower()=='w'): #word
			length=2
			out=output().int
			start = int(mem[2:])

		if(mem[1].lower()=='d'):
			out=output().dword
			length=4
			start = int(mem.split('.')[0][2:])
		if('freal' in mem.lower()): #double word (real numbers)
			length=4
			start=int(mem.lower().replace('freal',''))
			out=output().real
		#print start,hex(area)
		if(output().bool==out):
			bit = int(mem.split('.')[1])
		if(self.debug):
			print mem[0].lower(),bit
		self.plc.read_area(area,0,start,length)
		mbyte=self.plc.read_area(area,0,start,length)
		#print str(mbyte),start,length
		if(returnByte):
			return mbyte
		elif(output().bool==out):
			return get_bool(mbyte,0,bit)
		elif(output().int==out):
				return get_int(mbyte,start)
			elif(output().real==out):
				return get_real(mbyte,0)
			elif(output().dword==out):
				return get_dword(mbyte,0)
			elif(output().word==out):
				return get_int(mbyte,start)
	def writeMem(self,mem,value):
		data=self.getMem(mem,True)
		area=0x83
		length=1
		type=0
		out=None
		bit=0
		start=0
		if(mem[0].lower()=='m'):
			area=0x83
		if(mem[0].lower()=='q'):
			area=0x82
		if(mem[0].lower()=='i'):
			area=0x81
		if(mem[1].lower()=='x'): #bit
			length=1
			out=output().bool
			start = int(mem.split('.')[0][2:])
			bit = int(mem.split('.')[1])
			set_bool(data,0,bit,int(value))
		if(mem[1].lower()=='b'): #byte
			length=1
			out=output().int
			start = int(mem[2:])
			set_int(data,0,value)
		if(mem[1].lower()=='d'):
			out=output().dword
			length=4
			start = int(mem.split('.')[0][2:])
			set_dword(data,0,value)
		if('freal' in mem.lower()): #double word (real numbers)
			length=4
			start=int(mem.lower().replace('freal',''))
			out=output().real
			#print data
			set_real(data,0,value)
		return self.plc.write_area(area,0,start,data)

plc = S71200('10.10.55.131')  #,debug=True)
#turn on outputs cascading
for x in range(0,7):
	plc.writeMem('qx0.'+str(x),True)
	sleep(.5)
sleep(1)
#turn off outputs
for x in range(0,7):
	plc.writeMem('qx0.'+str(x),False)
	sleep(.5)
plc.plc.disconnect()
```