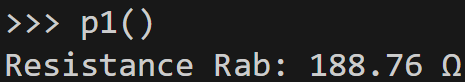
# 1)

## Code

def p1():  
 """Problem 1"""  
 r1 = 300 + 200  
 r2 = 1 / ((1 / r1) + (1 / 450))  
 r3 = 75 + r2  
 r4 = 1 / ((1 / 250) + (1 / r3))  
 Rab = 50 + r4  
 print("Resistance Rab: {:.2f} Ω".format(Rab))

## Output

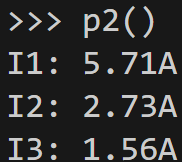


# 2)

## Code

def \_p2\_p3\_res():  
 """  
 Collapse certain branches and calculate resistances for problems 2 and 3.  
 Reduces repeat calculations.  
  
 :return tuple: collapsed resistances  
 """  
 r1 = 400 + 125  
 r2 = 1 / ((1 / 300) + (1 / r1))  
 r3 = 75 + r2  
  
 return (r1, r2, r3)  
  
def p2():  
 """Problem 2"""  
 (r1, r2, r3) = \_p2\_p3\_res()  
  
 i\_total = 10  
 i1 = i\_total \* (r3 / (r3 + 200))  
 i2 = (i\_total - i1) \* (r1 / (r1 + 300))  
 i3 = i\_total - i1 - i2  
  
 print("I1: {:.2f}A".format(i1))  
 print("I2: {:.2f}A".format(i2))  
 print("I3: {:.2f}A".format(i3))

## Output

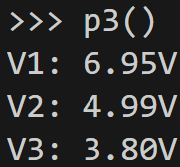


# 3)

## Code

def \_p2\_p3\_res():  
 """  
 Collapse certain branches and calculate resistances for problems 2 and 3.  
 Reduces repeat calculations.  
  
 :return tuple: collapsed resistances  
 """  
 r1 = 400 + 125  
 r2 = 1 / ((1 / 300) + (1 / r1))  
 r3 = 75 + r2  
  
 return (r1, r2, r3)  
  
def p3():  
 """Problem 3"""  
 (r1, r2, r3) = \_p2\_p3\_res()  
 r4 = 1 / ((1 / 200) + (1 / r3))  
  
 v0 = 10  
 v1 = v0 \* (r4 / (r4 + 50))  
 v2 = v1 \* (r2 / r3)  
 v3 = v2 \* (400 / r1)  
  
 print("V1: {:.2f}V".format(v1))  
 print("V2: {:.2f}V".format(v2))  
 print("V3: {:.2f}V".format(v3))

## Output



# 4)

## Code

def p4():  
 """Problem 4"""  
 #define imaginary number j  
 j = (-1) \*\* 0.5  
   
 z1 = 1 / ((1 / (200\*j)) + (1 / 100))  
 z2 = 40 + z1  
 z3 = 1 / ((1 / (-150\*j)) + (1 / z2))  
 z4 = 30 + z3  
 z5 = 1 / ((1 / (100\*j)) + (1 / z4))  
 Zab = 20 + z5  
  
 print ("Zab: {:.2f} Ω".format(Zab))

## Output

