```
print ('==== Problem 1 ====')
# density and viscosity of water
viscosity = q(0.8848, 'centipoise') # LEAVE IN cp
density = q(1.012, 'g/cm**3') # LEAVE IN g/mol

Re = HW3.ReynoldsNumber(L = q(1, 'cm').to('m'),
                        rho = density.to('kg/m**3'),
                        v = q(.2, 'm/s'),
                        mu = viscosity.to('kg/(m*s)')
                       )

d_AB = q(HW3.WilkeChangDiffusivity(298, # temperature in K
                        15.999 + 2, # molecular weight of B in g/mol
                        viscosity.magnitude, # viscosity of B in cp
                        43.82, # molar volume of A at normal boiling in cm**3/mol
                        2.6 # association factor ??
                       ), 'cm**2/s')

Sc = HW3.SchmidtNumber(mu = viscosity.to('kg/(m*s)'),
                       rho = density.to('kg/m**3'),
                       d_AB = d_AB.to('m**2/s')
                      )

Sh = 2 + (0.6 * Re**(1/2) * Sc**(1/3)) # CHECK COORELATION EQN
print(f"Sherwood Number: {Sh}")

c = (density.magnitude * gram / cm**3) / (18 * gram / mol)
print(f"Molarity: {c}")

kX = HW3.SherwoodSolver(Sh, # Sherwood Number
                    1*cm, # characteristic length
                    c, # molarity: mol/volume
                    d_AB.magnitude * cm**2 / second, # diffusivity: area/time
                    cnst.k, # mtc: mol/(area*time)
                    cnst.k, # variable to solve for
                    False # Low mass transfer
                   )
```

```
==== Problem 1 ====
Reynolds Number: 2287.5226039783
WilkeChangDiffusivity: 1.8×10⁻⁵ cm²/s
Schmidt Number: 495.39805035123237
Sherwood Number: 229.06596638023737
Molarity: 0.0562222222222222*mole/centimeter**3
Mass transfer coefficient: 0.00022728924403936 7*mole/(centimeter**2*second)
```

```python
print('==== Problem 2 ====')
print('==== GAS CALCULATIONS ====')
# density and viscosity of N2 at 298K
viscosity = q(.01724, 'centipoise') # LEAVE IN cp
density = q(0.0012506, 'g/cm**3') # LEAVE IN g/mol

Re = HW3.ReynoldsNumber(L = q(5, 'cm').to('m'),
                        rho = density.to('kg/m**3'),
                        v = q(1, 'foot/s').to('m/s'),
                        mu = viscosity.to('kg/(m*s)')
                       )

# toulene is C7H8
d_AB = q(HW3.FullerDiffusivity(
                        1, # pressure in atmospheres
                        298, # temperature in kelvin
                        (12 * 7) + (1 * 8), # molecular weight species A
                        14.007 * 2, # molecular weight species B
                        (15.9 * 7) + (2.31 * 8) - 18.3, # diffusion volume species A
                        18.5, # diffusion volume species B
                        'Toluene', # name of species A
                        'N2' # name of species B
                       ), 'cm**2/s')


Sc = HW3.SchmidtNumber(mu = viscosity.to('kg/(m*s)'),
                       rho = density.to('kg/m**3'),
                       d_AB = d_AB.to('m**2/s')
                      )

Sh = 1.2 * Re**(.64) * Sc**(1/3) # CHECK COORELATION EQN
print(f"Sherwood Number: {Sh}")

length, p, t = 5*cm, 1*atm, 298*kelvin
c = convert_to(p / (molar_gas_constant * t), mol/m**3)
print(f"Molarity: {c}")

kY = HW3.SherwoodSolver(Sh, # Sherwood Number
                        length, # characteristic length
                        c, # molarity: mol/volume
                        d_AB.magnitude * cm**2 / second, # diffusivity: area/time
                        cnst.k, # mtc: mol/(area*time)
                        cnst.k, # variable to solve for
                        False # Low mass transfer
                       )
```

```
==== Problem 2 ====
==== GAS CALCULATIONS ====
Reynolds Number: 1105.51879350348
mole_weight_ratios_Toluene_N2: 42.95
FullerDiffusivityToluene_N2: 0.083856 cm²/s
Schmidt Number: 1.643928545820097
Sherwood Number: 125.61009616121304
Molarity: 40.8946187070611*mole/meter**3
Mass transfer coefficient: 8.61502629562702e-5*mole/(centimeter**2*second)
```

```python
print('==== Liquid CALCULATIONS ====')
# density and viscosity of water at 298K
viscosity = q(0.8848, 'centipoise') # LEAVE IN cp
density = q(1.012, 'g/cm**3') # LEAVE IN g/mol

Re = HW3.ReynoldsNumber(L = q(5, 'cm').to('m'),
                        rho = density.to('kg/m**3'),
                        v = q(1, 'foot/s').to('m/s'),
                        mu = viscosity.to('kg/(m*s)')
                       )

d_AB = q(HW3.WilkeChangDiffusivity(298, # temperature in K
                        15.999 + 2, # molecular weight of B in g/mol
                        viscosity.magnitude, # viscosity of B in cp
                        118.5, # molar volume of A at normal boiling in cm**3/mol
                        2.6 # association factor ??
                       ), 'cm**2/s')

Sc = HW3.SchmidtNumber(mu = viscosity.to('kg/(m*s)'),
                       rho = density.to('kg/m**3'),
                       d_AB = d_AB.to('m**2/s')
                      )

Sh = 0.1 * Re**0.3 * Sc**0.5 # CHECK COORELATION EQN
print(f"Sherwood Number: {Sh}")

c = (density.magnitude * gram / cm**3) / (18 * gram / mol)
print(f"Molarity: {c}")

kX = HW3.SherwoodSolver(Sh, # Sherwood Number
                        5*cm, # characteristic length
                        c, # molarity: mol/volume
                        d_AB.magnitude * cm**2 / second, # diffusivity: area/time
                        cnst.k, # mtc: mol/(area*time)
                        cnst.k, # variable to solve for
                        False # Low mass transfer
                       )
```

```
==== Liquid CALCULATIONS ====
Reynolds Number: 17430.922242314642
WilkeChangDiffusivity: 1×10⁻⁵ cm²/s
Schmidt Number: 899.8743634273847
Sherwood Number: 56.16775288836209
Molarity: 0.0562222222222222*mole/centimeter**3
Mass transfer coefficient: 6.13631682320372e-6*mole/(centimeter**2*second)
PS C:\Users\Hunter Violett\OneDrive\Desktop\CHE362\exc_hw>
```

Inherited methods:

```python
@staticmethod
def SherwoodSolver(Sh: float, # Sherwood Number
                   L, # characteristic length
                   c, # molarity: mol/volume
                   d_AB, # diffusivity: area/time
                   k, # mtc: mol/(area*time)
                   solveFor = cnst.k, # variable to solve for
                   low_mass_transfer: bool = False, # low mass transfer rates
                  ):

  if low_mass_transfer:
    soln = solve(Eq((k  * L ) / d_AB, Sh), solveFor)
  else:
    soln = solve(Eq((k * L) / (c * d_AB), Sh), solveFor)

  print(f'Mass transfer coefficient: {soln[0]}')
  return soln
```

```python
@staticmethod
def ReynoldsNumber(L: pint.Quantity,
                   rho: pint.Quantity,
                   v: pint.Quantity,
                   mu: pint.Quantity
                  ):

  Re = (L * rho * v) / mu
  print(f'Reynolds Number: {Re}')
  return Re
```

```python
@staticmethod
def SchmidtNumber(mu: pint.Quantity,
                  rho: pint.Quantity,
                  d_AB: pint.Quantity,
                 ):

  Sc = mu / (rho * d_AB)
  print(f'Schmidt Number: {Sc}')
  return Sc
```

```python
@staticmethod
def FullerDiffusivity(
                  p: float, # pressure in atmospheres
                  t: float, # temperature in kelvin
                  molW1: float, # molecular weight species A
                  molW2: float, # molecular weight species B
                  dVol1: float, # diffusion volume species A
                  dVol2: float, # diffusion volume species B
                  specA: str = 'A', # name of species A
                  specB: str = 'B', # name of species B
                 ):

  m_AB = 2 / ((1 / molW1) + (1 / molW2))
  d_AB = (0.00143 * t**1.75) / (p * m_AB**(1/2) * (dVol1**(1/3) + dVol2**(1/3))**2 )

  print(f'mole_weight_ratios_{specA}_{specB}: {round(m_AB,2)}')
  print(f'FullerDiffusivity{specA}_{specB}: {q(round(d_AB, 6), "cm**2/s")}')

  return d_AB
```

```python
@staticmethod
def WilkeChangDiffusivity( # A:solute, B:solvent
    t: float, # temperature in K
    mW_B: float, # molecular weight of B in g/mol
    mu_B: float, # viscosity of B in cp
    mVol_A_normal: float, # molar volume of A at normal boiling in cm**3/mol
    aF: float = 1.5 # association factor {2.6: water, 1.9: methanol, 1.5: ethanol, 1.0: unassociated}
  ):

  d_AB = (7.4 * 10**-8 * (aF*mW_B)**(1/2) * t) / (mu_B * mVol_A_normal**0.6)

  print(f'WilkeChangDiffusivity: {q(round(d_AB, 6), "cm**2/s")}')
  return d_AB
```