Constants in SciPy # As SciPy is more focused on scientific implementations, it provides many built-in scientific constants. # These constants can be helpful when you are working with Data Science.

In [3]:
```
from scipy import constants
print(constants.liter)
print(constants.pi)
```

```
0.001
3.141592653589793
```

In [4]:
```
#list all constants
print(dir(constants))
```

```
['Avogadro', 'Boltzmann', 'Btu', 'Btu_IT', 'Btu_th', 'ConstantWarning', 'G', 'Jul
ian_year', 'N_A', 'Planck', 'R', 'Rydberg', 'Stefan_Boltzmann', 'Wien', '__all_
_', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name_
_', '__package__', '__path__', '__spec__', '_codata', '_constants', '_obsolete_co
nstants', 'acre', 'alpha', 'angstrom', 'arcmin', 'arcminute', 'arcsec', 'arcsecon
d', 'astronomical_unit', 'atm', 'atmosphere', 'atomic_mass', 'atto', 'au', 'bar',
'barrel', 'bbl', 'blob', 'c', 'calorie', 'calorie_IT', 'calorie_th', 'carat', 'ce
nti', 'codata', 'constants', 'convert_temperature', 'day', 'deci', 'degree', 'deg
ree_Fahrenheit', 'deka', 'dyn', 'dyne', 'e', 'eV', 'electron_mass', 'electron_vol
t', 'elementary_charge', 'epsilon_0', 'erg', 'exa', 'exbi', 'femto', 'fermi', 'fi
nd', 'fine_structure', 'fluid_ounce', 'fluid_ounce_US', 'fluid_ounce_imp', 'foo
t', 'g', 'gallon', 'gallon_US', 'gallon_imp', 'gas_constant', 'gibi', 'giga', 'go
lden', 'golden_ratio', 'grain', 'gram', 'gravitational_constant', 'h', 'hbar', 'h
ectare', 'hecto', 'horsepower', 'hour', 'hp', 'inch', 'k', 'kgf', 'kibi', 'kilo',
'kilogram_force', 'kmh', 'knot', 'lambda2nu', 'lb', 'lbf', 'light_year', 'liter',
'litre', 'long_ton', 'm_e', 'm_n', 'm_p', 'm_u', 'mach', 'mebi', 'mega', 'metric_
ton', 'micro', 'micron', 'mil', 'mile', 'milli', 'minute', 'mmHg', 'mph', 'mu_0',
'nano', 'nautical_mile', 'neutron_mass', 'nu2lambda', 'ounce', 'oz', 'parsec', 'p
ebi', 'peta', 'physical_constants', 'pi', 'pico', 'point', 'pound', 'pound_forc
e', 'precision', 'proton_mass', 'psi', 'pt', 'short_ton', 'sigma', 'slinch', 'slu
g', 'speed_of_light', 'speed_of_sound', 'stone', 'survey_foot', 'survey_mile', 't
ebi', 'tera', 'test', 'ton_TNT', 'torr', 'troy_ounce', 'troy_pound', 'u', 'unit',
'value', 'week', 'yard', 'year', 'yobi', 'yocto', 'yotta', 'zebi', 'zepto', 'zero
_Celsius', 'zetta']
```

In [10]:
```
#metric prefixes
print(constants.yotta)
print(constants.zetta)
print(constants.exa)
print(constants.peta)
print(constants.tera)
print(constants.giga)
print(constants.mega)
print(constants.kilo)
print(constants.hecto)
print(constants.deka)
print(constants.deci)
print(constants.centi)
print(constants.milli)
print(constants.micro)
print(constants.nano)
print(constants.pico)
print(constants.femto)
print(constants.atto)
print(constants.zepto)
```

```
1e+24
1e+21
1e+18
1000000000000000.0
1000000000000.0
1000000000.0
1000000.0
1000.0
100.0
10.0
0.1
0.01
0.001
1e-06
1e-09
1e-12
1e-15
1e-18
1e-21
```

In [11]:
```
#Binary prefixes
print(constants.kibi)
print(constants.mebi)
print(constants.gibi)
print(constants.tebi)
print(constants.pebi)
print(constants.exbi)
print(constants.zebi)
print(constants.yobi)
```

```
1024
1048576
1073741824
1099511627776
1125899906842624
1152921504606846976
1180591620717411303424
1208925819614629174706176
```

In [12]:
```
#Mass
print(constants.gram)
print(constants.metric_ton)
print(constants.grain)
print(constants.lb)
print(constants.pound)
print(constants.oz)
print(constants.ounce)
print(constants.stone)
print(constants.long_ton)
print(constants.short_ton)
print(constants.troy_ounce)
print(constants.troy_pound)
print(constants.carat)
print(constants.atomic_mass)
print(constants.m_u)
print(constants.u)
```

```
0.001
1000.0
6.479891e-05
0.45359236999999997
0.45359236999999997
0.028349523124999998
0.028349523124999998
6.3502931799999995
1016.0469088
907.1847399999999
0.031103476799999998
0.37324172159999996
0.0002
1.6605390666e-27
1.6605390666e-27
1.6605390666e-27
```

In [13]:
```
#Angles
print(constants.degree)
print(constants.arcmin)
print(constants.arcminute)
print(constants.arcsec)
print(constants.arcsecond)
```

```
0.017453292519943295
0.0002908882086657216
0.0002908882086657216
4.84813681109536e-06
4.84813681109536e-06
```

In [14]:
```
#Time
print(constants.minute)
print(constants.hour)
print(constants.day)
print(constants.week)
print(constants.year)
print(constants.Julian_year)
```

```
60.0
3600.0
86400.0
604800.0
31536000.0
31557600.0
```

In [15]:
```
#Length
print(constants.inch)
print(constants.foot)
print(constants.yard)
print(constants.mile)
print(constants.mil)
print(constants.pt)
print(constants.point)
print(constants.survey_foot)
print(constants.survey_mile)
print(constants.nautical_mile)
print(constants.fermi)
print(constants.angstrom)
print(constants.micron)
print(constants.au)
print(constants.astronomical_unit)
```

```
print(constants.light_year)
print(constants.parsec)
```

```
0.0254
0.30479999999999996
0.9143999999999999
1609.3439999999998
2.5399999999999997e-05
0.00035277777777777776
0.00035277777777777776
0.3048006096012192
1609.3472186944373
1852.0
1e-15
1e-10
1e-06
149597870700.0
149597870700.0
9460730472580800.0
3.085677581491367e+16
```

In [16]:
```
#Pressure
print(constants.atm)
print(constants.atmosphere)
print(constants.bar)
print(constants.torr)
print(constants.mmHg)
print(constants.psi)
```

```
101325.0
101325.0
100000.0
133.32236842105263
133.32236842105263
6894.757293168361
```

In [17]:
```
#Areas
print(constants.hectare)
print(constants.acre)
```

```
10000.0
4046.8564223999992
```

In [18]:
```
#Volume
print(constants.liter)
print(constants.litre)
print(constants.gallon)
print(constants.gallon_US)
print(constants.gallon_imp)
print(constants.fluid_ounce)
print(constants.fluid_ounce_US)
print(constants.fluid_ounce_imp)
print(constants.barrel)
print(constants.bbl)
```

```
0.001
0.001
0.0037854117839999997
0.0037854117839999997
0.00454609
2.9573529562499998e-05
2.9573529562499998e-05
2.84130625e-05
0.15898729492799998
0.15898729492799998
```

In [19]:
```
#Speed
print(constants.kmh)
print(constants.mph)
print(constants.mach)
print(constants.speed_of_sound)
print(constants.knot)
```

```
0.2777777777777778
0.44703999999999994
340.5
340.5
0.5144444444444445
```

In [20]:
```
#Temperature
print(constants.zero_Celsius)
print(constants.degree_Fahrenheit)
```

```
273.15
0.5555555555555556
```

In [21]:
```
#Energy
print(constants.eV)
print(constants.electron_volt)
print(constants.calorie)
print(constants.calorie_th)
print(constants.calorie_IT)
print(constants.erg)
print(constants.Btu)
print(constants.Btu_IT)
print(constants.Btu_th)
print(constants.ton_TNT)
```

```
1.602176634e-19
1.602176634e-19
4.184
4.184
4.1868
1e-07
1055.05585262
1055.05585262
1054.3502644888888
4184000000.0
```

In [22]:
```
#Power
print(constants.hp)
print(constants.horsepower)
```

```
745.6998715822701
745.6998715822701
```

In [4]:
```
#Force
print(constants.dyn)
print(constants.dyne)
print(constants.lbf)
print(constants.pound_force)
print(constants.kgf)
print(constants.kilogram_force)
```

```
1e-05
1e-05
4.4482216152605
4.4482216152605
9.80665
9.80665
```

In [1]:
```

```

```
[-0.73908513]
```

In [ ]:
```

```