Utilisation d'une "classe" pour des données de solvants chimiques

On peut utiliser la structure de classe pour créer une “table” de données sur des solvants. Il est alors possible d'effectuer des traitements de tris, sélection, impression...

<sxh python; title : organic_solvents_data_class.py> #!/usr/bin/env python # -*- coding: utf-8 -*-

a solvent database using a list of class instances # a modification of: http://www.daniweb.com/code/snippet390.html # EU tested with Python 2.4 10/30/2006 # # modification UMONS - cours programmation python - 2007-2012

import operator # for attrgetter()

class Solvent(object):

```
  """a structure class for solvents, self refers to the instance"""
  def __init__(self, name=9999, bp=9999, mp=9999, fp=9999, dy=9999, de=9999,
nr=9999, mu=9999, so=9999, fo=9999):
      self.name = name
      # 9999 --> not applicable
      self.bp = bp     # boiling point degC
      self.mp = mp     # melting point degC
      self.fp = fp     # flash point degC
      self.dy = dy     # density g/ml
      self.de = de     # dielectric constant
      self.nr = nr     # refractive index
      self.mu = mu     # dipole moment in Debye
      self.so = so     # solubility in water (g/100g water) --> 9000 =
miscible
      self.fo = fo     # chemical formula
```

def table(solvent_list):

```
  """print a table of all solvent attributes"""
  title_str = "%-20s  %8s  %8s  %8s  %8s  %8s  %8s  %8s  %8s  %20s"
  data_str  = "%-20s  %8.1f  %8.1f  %8.1f  %8.3f  %8.2f  %8.2f  %8.2f  %8.2f
%-20s"
  print "-"*122
  print title_str % ("Name", "Bp", "Mp", "Fp", "Density", "Dielect", "Ref.
Ind", "Dipole", "H2O Sol.", "Formula")
  for solvent in solvent_list:
      print data_str % (solvent.name, solvent.bp, solvent.mp, solvent.fp,
solvent.dy, solvent.de, solvent.nr, solvent.mu, solvent.so, solvent.fo)
  print "-"*122
  print "9999 --> not applicable; 9000 --> miscible"
  print
```

def table_bp(solvent_list, bp_limit):

```
  """print a table of all solvent attributes, with bp restrictions"""
  title_str = "%-20s  %8s  %8s  %8s  %8s  %8s"
  data_str  = "%-20s  %8.1f  %8.1f  %8.1f  %8.3f    %8.2f"
  print "-"*72
  print title_str % ("Name", "Bp", "Mp", "Fp", "Density", "Dielectric")
  for solvent in solvent_list:
      if solvent.bp > bp_limit:
          print data_str % (solvent.name, solvent.bp, solvent.mp,
solvent.fp, solvent.dy, solvent.de)
  print "-"*72
  print "9999 --> not applicable; 9000 --> miscible"
  print
```

make a list of class Solvent instances # also adds all the data/information # data order = name, boiling point, melting point, flash point, density, dielectric constant, # refractive index, dipole moment in Debye, solubility in water (g/100g water), chemical formula # 9999 –> not applicable; 9000 –> miscible solvent_list = [] solvent_list.append(Solvent(“methanol”, 64.7, -97.7, 11, 0.791, 32.7, 1.3284, 1.7, 9900, “CH4O”)) solvent_list.append(Solvent(“ethanol”, 78.3, -114.1, 8, 0.789, 24.55, 1.3614, 1.69, 9900, “C2H6O”)) solvent_list.append(Solvent(“propanol iso”, 82.3, -88, 22, 0.785, 19.92, 1.3772, 1.66, 9900, “C3H8O”)) solvent_list.append(Solvent(“butanol normal”, 117.7, -88.6, 35, 0.81, 17.51, 9999, 9999, 7.7, “C4H10O”)) solvent_list.append(Solvent(“butanol secondary”, 88.8, -114.7, 26, 0.805, 16.56, 9999, 9999, 9999, “C4H10O”)) solvent_list.append(Solvent(“butanol tertiary”, 82.2, 25.5, 4, 0.786, 10.9, 1.3877, 1.66, 9900, “C4H10O”)) #solvent_list.append(Solvent(“benzyl alcohol”, 205.4, -15.3, 100, 1.045, 13.1, 9999, 9999, 9999, “unknown”)) solvent_list.append(Solvent(“acetone”, 56.3, -94.7, -17, 0.791, 20.7, 1.3587, 2.85, 9900, “C3H6O”)) solvent_list.append(Solvent(“toluene”, 110.6, -94.9, 4, 0.867, 2.38, 1.4969, 0.43, 0.05, “C7H8”)) solvent_list.append(Solvent(“water”, 100, 0, 9999, 1, 78.5, 1.333, 1.82, 9900, “H2O”)) solvent_list.append(Solvent(“dimethyl formamide”, 153, -61, 58, 0.944, 36.7, 1.4305, 3.86, 9900, “C3H7NO”)) solvent_list.append(Solvent(“acetic acid”, 118, 17, 39, 1.049, 6.15, 1.3716, 1.68, 9900, “C2H4O2”)) # got the drift, add more solvents here …

print “Sort the solvent_list by name …” solvent_list.sort(key=operator.attrgetter('name')) # … now show a table of the solvent_list table(solvent_list)

print

print “Sort the solvent_list by melting point …” solvent_list.sort(key=operator.attrgetter('mp')) # … now show a table of the solvent_list table(solvent_list)

print

print “Sort the solvent_list by boiling point …” solvent_list.sort(key=operator.attrgetter('bp')) # … now show a table of the solvent_list table(solvent_list)

print

ajouter suivant d'autres tris (solubilité ,…)

bp_limit = 75 print “Show only solvents boiling higher than %0.1f degC:” % bp_limit # show a boiling point restricted table table_bp(solvent_list, bp_limit)

on peut chercher un solvant de densité supérieure à l'eau, mais # peu soluble ou insoluble (solubilité < 0.05 par exemple)

</sxh>