```
from __future__ import division
import sys
import random
import math
import numpy as np
from models import *
sys.dont_write_bytecode = True

"""
#Dr.M
def msecs(f):
  import time
  t1 = time.time()
  f()
  return (time.time() − t1) * 1000

class Utilities:
  def randR(self,lo,hi):
    return(lo+rand()*(hi−lo))

  def norm(self,x,lo,hi):
    "Generate a num 0..1 for lo..hi"
    tmp = (x − lo) / (hi − lo + 0.00001)
    return max(0,min(tmp,1))

  def say(self,x):
    "Print something with no trailing new line."
    sys.stdout.write(str(x)); sys.stdout.flush()



  def gn(lst,n):
    "Function to print floats in short form"
    fmt = ’%.’ + str(n) + ’f’
    return ’, ’.join([(fmt % x) for x in lst])

  def logo():

              /\
             /  \
            /    \
           / ARig \
          /        \
         /__________\


#Dr.M
  def msecs(f):
    import time
    t1 = time.time()
    f()
    return (time.time() − t1) * 1000


"""
def say(x):
    "Print something with no trailing new line."
    sys.stdout.write(str(x)+"\n"); sys.stdout.flush()

def pairs(lst):
    last=lst[0]
    for i in lst[1:]:
      yield last,i
      last = i


def xtile(lst,lo=0,hi=0.001,width=50,
             chops=[0.1 ,0.3,0.5,0.7,0.9],
             marks=["−" ," "," ","−"," "],
             bar="|",star="*",show=" %3.0f"):
    """The function _xtile_ takes a list of (possibly)
  unsorted numbers and presents them as a horizontal
  xtile chart (in ascii format). The default is a
  contracted _quintile_ that shows the
  10,30,50,70,90 breaks in the data (but this can be
  changed− see the optional flags of the function).
  """
    def pos(p)   : return ordered[int(len(lst)*p)]
    def place(x) :
      return int(width*float((x − lo))/(hi − lo))
    def pretty(lst) :
      return ’,’.join([show % x for x in lst])
    ordered = sorted(lst)
    #print ordered
    lo      = min(lo,ordered[0])
    hi      = max(hi,ordered[−1])
    what    = [pos(p)   for p in chops]
    where   = [place(n) for n in  what]
    out     = [" "] * width
    for one,two in pairs(where):
      for i in range(one,two):
        out[i] = marks[0]
      marks = marks[1:]
    out[int(width/2)]    = bar
    #print pos(0.5)
    #print place(pos(0.5))
    if(place(pos(0.5))<width):
      out[place(pos(0.5))] = star
    else:
      out[width−1] = star
    return ’’.join(out) +  "," +  pretty(what)
```