```python
# -*- coding: cp1252 -*-
import pcraster
import os

"""NICOLAS ANTONIO LOPEZ ROZO:

This module is used for the TACD2 model, given that there is a DEM file in
Arc/Info format (.asc). For that kind of format, the first lines of the
.asc file follow a specific format as below:

ncols         850
nrows         593
xllcorner     830557.373698523270
yllcorner     956339.482658421620
cellsize      92.769992529088
NODATA_value -32768
....(continues with map description)

This script takes advantage of that format to create CLONE, DEM, LDD and
other maps in .MAP format (required for PCRaster)


"""
print os.popen("time /t").readline()
#Archivos de origen y destino para hacer la conversión
origendem = "Cuenca_DEM.asc"
origenldd = "LDD.asc"
dem = "dem.map"
ldd = "ldd01.map"
origenurban = "Urban.asc"
urban = "urban.map"

#stream = os.popen("asc2col -a -clone " + )
#pcraster.asc2col(origen, destino)

def createCloneDEM(destinationFile, originFile):
  print os.popen("time /t").readline()
  #Reading properties from .ASC file
  f = open(originFile, 'r')
  line = f.readline().strip().split()
  colcount = int(line[1])
  line = f.readline().strip().split()
  rowcount = int(line[1])
  line = f.readline().strip().split()
  xllcorner = float(line[1])
  line = f.readline().strip().split()
  yllcorner = float(line[1])
  line = f.readline().strip().split()
  cellsize = float(line[1])
  line = f.readline().strip().split()
  nodatavalue = int(line[1])
  f.close()
  #print colcount, rowcount, xllcorner, yllcorner, cellsize, nodatavalue
  #print "mapattr -s -C {0} -R {1} -B -x {2} -y {3} -l {4} clone.map".format(colcount,
rowcount, xllcorner, yllcorner, cellsize, nodatavalue)
  #Delete previous clone.map if it existed
  stream = os.popen("del clone.map")
  print "creating basis clone.map ..."
  stream = os.popen("mapattr -s -C {0} -R {1} -B -x {2} -y {3} -l {4} clone.map".format(colcount, rowcount, xllcorner, yllcorner, cellsize, nodatavalue))
  for line in stream.readlines():
    print line
  print "creating {0} ...".format(destinationFile)
  stream = os.popen("asc2map --clone clone.map -S -m {0} -a {1} {2}".format(nodatavalue, originFile, destinationFile))
  for line in stream.readlines():
    print line
  print "creating actual clone.map ..."
  stream = os.popen("pcrcalc clone1.map={0} ne {1}".format(destinationFile, nodatavalue))
  stream = os.popen("erase clone.map")
  stream = os.popen("ren clone1.map clone.map")
  return

def transformLDD(originFile):
  #function not used because LDD is worse than ArcGIS
  #ldd = pcraster.ldd(dem)
  #ldd2 = pcraster.lddrepair(ldd)
  #pcraster.report(ldd2, LDDFile)
  #return
  print os.popen("time /t").readline()
  stream = os.popen("del myLDD.asc")
  print "Converting LDD File from {0}".format(originFile)
  #transforming 8D notation from 0-1-2-4-8-16-32-64-128-255 to 0-9
  mydict=dict()
  mydict[0] = 5
  mydict[1] = 6
  mydict[2] = 3
  mydict[4] = 2
  mydict[8] = 1
  mydict[16] = 4
  mydict[32] = 7
  mydict[64] = 8
  mydict[128] = 9
  f = open(originFile, 'r')
  f2= open("myLDD.asc", 'w')
  # file header is the same
  """original header:yllcorner    956339.482658421620
cellsize     92.769992529088
"""
  for i in range(5):
    line=f.readline().strip()
    f2.write("{0}\n".format(line))
  #reading nodatavalue from file
  line = f.readline().strip()
  f2.write("{0}\n".format(line))
  nodatavalue = line.split()[1]
  mydict[int(nodatavalue)] = int(nodatavalue)
  # now, converting pixel values as required
```

```python
  i,j = 1,1
  for line in f.readlines():
    vals = line.strip().split()
    j=1
    for val in vals:
      ################################################################
      # BEWARE: CHANGE THIS CODE TO SELECT YOUR EXIT CELL AS PIT
      ################################################################
      # this is the cell that should be the pit (sink)
      if j==844 and i==501 and mydict[int(val)]==9:
        f2.write(" 5")
      else:
        f2.write(" {0}".format(mydict[int(val)]))
      j+=1
      ################################################################
      # BEWARE: CHANGE THIS CODE TO SELECT YOUR EXIT CELL AS PIT
      ################################################################
    f2.write("\n")
    i+=1
  f.close()
  f2.close()
  return

def createLDD(destinationFile):
  print os.popen("time /t").readline()
  #As clone.map is already created, we can go directly to this step
  print "Creating LDD File from myLDD.asc ..."
  stream = os.popen("del {0}".format(destinationFile))
  stream = os.popen("asc2map --clone clone.map -L -m 255 -a myLDD.asc {0}".format(destinationFile))
  for line in stream.readlines():
    print line

def convertUrban(destinationFile, originFile):
  print os.popen("time /t").readline()
  print "creating Urban map from {0} ...".format(originFile)
  stream = os.popen("del {0}".format(destinationFile))
  stream = os.popen("asc2map --clone clone.map -S -m 255 -a {1} {0}".format(destinationFile, originFile))
  for line in stream.readlines():
    print line
  return

def createLDDPCR():
  print os.popen("time /t").readline()
  pcraster.setglobaloption("lddin")
  dem = pcraster.readmap("dem.map")
  res = pcraster.lddcreate(dem, 999999, 999999999, 999999, 999999)
  pcraster.report(res, "myldd.map")

#script starts here
#createCloneDEM(dem, origendem) # <---it's better to do pcrcalc clone.map = dem.tif>=0, as it preserves spatial reference
transformLDD(origenldd)
```

```
createLDD(ldd)
#convertUrban(urban, origenurban)

print os.popen("time /t").readline()
print "exiting script ..."

```