## week 1, program 1

# Program to convert the 4 control points scanner coordinates

# to Mercator Projection Grid Eastings and Northings

#

# <2486040Y> <Yang, Jiale> <04/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Open the file - control points(geographical coordinates)

# the file's address is same as the address where the program was stored

madagascar = open('madagascar.txt',"r")

# Read in the data

location = csv.reader(madagascar)

# Open another file and write - to store the output result(Mercator coordinates)

# CP - control point

f=open("CP\_Mercator.txt","w")

# Convert the geographical coordinates to Mercator coordinates

for row in location:

# Print the origin data

print row

# convert the Easting coordinate and print

a = 0.0

a = float(row[1])

x = R \* math.radians(a)

print 'X = ', x

# convert the Northing coordinate and print

b = 0.0

b = float(row[0])

y = R \* math.log((math.tan(math.pi/4 + math.radians(b)/2)),math.e)

print 'Y = ', y

# Write the calculation result into file

f.write(str(x)+",")

f.write(str(y)+"\n")

# Close the written file

f.close()

# Close the read file

madagascar.close()

## week 2, program 2

# Use the two-dimensional similarity transformation constants obtained last week in excel

# Program convert the digitized coastline from scanner coordinates

# to Mercator Projection grid coordinates

#

# <2486040Y> <Yang, Jiale> <11/11/2019>

# Status - test success

# Import modules

import csv

# Open the file - digitized coastline points

# the file's address is same as the address where the program was stored

CoastDigitised = open('CoastlineDigitizedPointsInput.txt',"r")

# Read in the data

#CDP = coastline digitised points coordinates

CDP = csv.reader(CoastDigitised)

# Open another file and write - to store the coastline Mercator coordinates

CoastMerfile=open("CoastlineMercatorOutput.txt","w")

# Convert the digitized coordinates to Mercator coordinates

for row in CDP:

# Print the origin data

print row

# change the data format and assign the value to variables

x0 = 0.0

x0 = float(row[0])

y0 = 0.0

y0 = float(row[1])

# calculate the x,y Mercator coordinates

# the constants calculated by excel

x = (19889.45374\*x0 - 101.5492431\*y0) - 5010720.653

y = (101.5492431\*x0 + 19889.45374\*y0) + 3199431.827

# print the calculation result

print 'X = ', x

print 'Y = ', y

# Write the calculation result into file

CoastMerfile.write(str(x)+",")

CoastMerfile.write(str(y)+"\n")

# Close the written file

CoastMerfile.close()

# Close the read file

CoastDigitised.close()

## week 2, program 3

# Program convert the coastline coordinates

# from Mercator Projection grid coordinates to Geographical coordinates (φ,λ)

#

# <2486040Y> <Yang, Jiale> <11/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Open the file - Mercator coastline points

# the file's address is same as the address where the program was stored

CoastMercator = open('CoastlineMercatorOutput.txt',"r")

# Read in the data

#CMP = coastline mercator points coordinates

CMP = csv.reader(CoastMercator)

# Open another file and write - to store the coastline geographical coordinates

CoastGeofile=open("CoastlineGeographicOutput.txt","w")

# convert the coastline coordinates from Mercator to Geographical coordinates

for row in CMP:

# Print the origin Mercator data

print row

# convert the X coordinate and print

a = 0.0

a = float(row[0])

x = math.degrees(a/R)

print 'X = ', x

# convert the Y coordinate and print

b = 0.0

b = float(row[1])

y = math.degrees(math.atan(math.sinh(b/R)))

#inverse: y = R \* math.log((math.tan(math.pi/4 + math.radians(b)/2)),math.e)

print 'Y = ', y

# Write the calculation result into file

CoastGeofile.write(str(x)+",")

CoastGeofile.write(str(y)+"\n")

# Close the written file

CoastGeofile.close()

# Close the read file

CoastMercator.close()

## week 3, program 4

# Program convert the coastline coordinates

# from Geographical coordinates to Lambert Conformal Conical Projection grid coordinates

#

# <2486040Y> <Yang, Jiale> <18/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Inputs - convert standard parallels from degrees to radians

Q1 = math.radians(-10)

Q2 = math.radians(-25)

# cone constant calculation

n1 = math.log((math.cos(Q1)/math.cos(Q2)),math.e)

n2 = math.log(((math.tan(math.pi/4+Q2/2))/(math.tan(math.pi/4+Q1/2))),math.e)

n = float(n1/n2)

# calculate F

F = (math.cos(Q1))\*(math.tan((math.pi/4)+(Q1/2))\*\*n)/n

# check information

print Q1, Q2, n, F

# Open the file - coastline geographical coordinates

# the file's address is same as the address where the program was stored

CoastGeographic = open('CoastlineGeographicOutput.txt',"r")

# Read in the data

#CGP = coastline geographic points coordinates

CGP = csv.reader(CoastGeographic)

# Open another file and write - to store Lambert Conformal Conical Projection grid coordinates

CoastLCCfile=open("CoastlineLambertCCOutput.txt","w")

# Convert Geographical coordinates to Lambert coordinates

for row in CGP:

# Print the origin geographical data

print row

# change the data format, convert degree to radians, and assign the value to variables

a = 0.0

a = math.radians(float(row[0]))

b = 0.0

b = math.radians(float(row[1]))

# calculate G, RhoO, Rho

G = n\*(a-0)

RhoO = R\*F/(math.tan(math.pi/4)\*\*n)

Rho = R\*F/(math.tan(math.pi/4+b/2)\*\*n)

# calculate the x,y LCC coordinates and print them

x = Rho\*math.sin(G)

print 'X = ', x

y = RhoO-Rho\*math.cos(G)

print 'Y = ', y

# Write the calculation result into file

CoastLCCfile.write(str(x)+",")

CoastLCCfile.write(str(y)+"\n")

# Close the written file

CoastLCCfile.close()

# Close the read file

CoastGeographic.close()

## week 4, program 5

# Program to convert the 4 control points scanner coordinates

# to Mercator Projection Grid Eastings and Northings

#

# <2486040Y> <Yang, Jiale> <25/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Open the file - control points(geographical coordinates)

# the file's address is same as the address where the program was stored

madagascar = open('madagascar.txt',"r")

# Read in the data

location = csv.reader(madagascar)

# Open another file and write - to store the output result(Mercator coordinates)

f=open("mercator.txt","w")

# Convert the geographical coordinates to Mercator coordinates

for row in location:

# Print the origin data

print row

# convert the Easting coordinate and print

a = 0.0

a = float(row[1])

x = R \* math.radians(a)

print 'X = ', x

# convert the Northing coordinate and print

b = 0.0

b = float(row[0])

y = R \* math.log((math.tan(math.pi/4 + math.radians(b)/2)),math.e)

print 'Y = ', y

# Write the calculation result into file

f.write(str(x)+",")

f.write(str(y)+"\n")

# Close the written file

f.close()

# Close the read file

madagascar.close()

## week 4, program 6

## all calculation of this program based on least square

# Use the two-dimensional similarity transformation constants obtained in excel

# Program convert the digitized coastline from scanner coordinates

# to Mercator Projection grid coordinates

#

# <2486040Y> <Yang, Jiale> <25/11/2019>

# Status - test success

# Import modules

import csv

# Open the file - digitized coastline points

# the file's address is same as the address where the program was stored

CoastDigitised = open('CoastlineDigitizedPointsInput.txt',"r")

# Read in the data

#CDP = coastline digitised points coordinates

CDP = csv.reader(CoastDigitised)

# Open another file and write - to store the coastline Mercator coordinates

CoastMerfile=open("CoastlineMercatorOutput\_LSq.txt","w")

# Convert the digitized coordinates to Mercator coordinates

for row in CDP:

# Print the origin data

print row

# change the data format and assign the value to variables

x0 = 0.0

x0 = float(row[0])

y0 = 0.0

y0 = float(row[1])

# calculate the x,y Mercator coordinates

# the constants calculated by excel

x = (19889.45374\*x0 - (-5.18503E-09)\*y0) - 4985181.019

y = ((-5.18503E-09)\*x0 + 19889.45374\*y0) + 3250460.322

# print the calculation result

print 'X = ', x

print 'Y = ', y

# Write the calculation result into file

CoastMerfile.write(str(x)+",")

CoastMerfile.write(str(y)+"\n")

# Close the file

CoastMerfile.close()

CoastDigitised.close()

## week 4, program 7

## all calculation of this program based on least square

# Program convert the coastline coordinates

# from Mercator Projection grid coordinates to Geographical coordinates (φ,λ)

#

# <2486040Y> <Yang, Jiale> <25/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Open the file - Mercator coastline points

# the file's address is same as the address where the program was stored

CoastMercator = open('CoastlineMercatorOutput\_LSq.txt',"r")

# Read in the data

#CMP = coastline mercator points coordinates

CMP = csv.reader(CoastMercator)

# Open another file and write - to store the coastline geographical coordinates

CoastGeofile=open("CoastlineGeographicOutput\_LSq.txt","w")

# convert the coastline coordinates from Mercator to Geographical coordinates

for row in CMP:

# Print the origin Mercator data

print row

# convert the Y coordinate and print

a = 0.0

a = float(row[0])

x = math.degrees(a/R)

print 'X = ', x

# convert the Y coordinate and print

b = 0.0

b = float(row[1])

y = math.degrees(math.atan(math.sinh(b/R)))

#inverse y = R \* math.log((math.tan(math.pi/4 + math.radians(b)/2)),math.e)

print 'Y = ', y

# Write the calculation result into file

CoastGeofile.write(str(x)+",")

CoastGeofile.write(str(y)+"\n")

# Close the file

CoastGeofile.close()

CoastMercator.close()

## week 4, program 8

## all calculation of this program based on least square

# Program convert the coastline coordinates

# from Geographical coordinates to Lambert Conformal Conical Projection grid coordinates

#

# <2486040Y> <Yang, Jiale> <25/11/2019>

# Status - test success

# Import modules

import math

import csv

# Inputs - Declare R(Earth radius)

R = 6378000

# Inputs - convert standard parallels from degrees to radians

Q1 = math.radians(-10)

Q2 = math.radians(-25)

# cone constant calculation

n1 = math.log((math.cos(Q1)/math.cos(Q2)),math.e)

n2 = math.log(((math.tan(math.pi/4+Q2/2))/(math.tan(math.pi/4+Q1/2))),math.e)

n = float(n1/n2)

# calculate F

F = (math.cos(Q1))\*(math.tan((math.pi/4)+(Q1/2))\*\*n)/n

# check information

print Q1, Q2, n, F

# Open the file - coastline geographical coordinates

# the file's address is same as the address where the program was stored

CoastGeographic = open('CoastlineGeographicOutput\_LSq.txt',"r")

# Read in the data

#CGP = coastline geographic points coordinates

CGP = csv.reader(CoastGeographic)

# Open another file and write - to store Lambert Conformal Conical Projection grid coordinates

CoastLCCfile=open("CoastlineLambertCCOutput\_LSq.txt","w")

# Convert Geographical coordinates to Lambert coordinates

for row in CGP:

# Print the origin geographical data

print row

# change the data format, convert degree to radians, and assign the value to variables

a = 0.0

a = math.radians(float(row[0]))

b = 0.0

b = math.radians(float(row[1]))

# calculate G, RhoO, Rho

G = n\*(a-0)

RhoO = R\*F/(math.tan(math.pi/4)\*\*n)

Rho = R\*F/(math.tan(math.pi/4+b/2)\*\*n)

# calculate the x,y LCC coordinates and print them

x = Rho\*math.sin(G)

print 'X = ', x

y = RhoO-Rho\*math.cos(G)

print 'Y = ', y

# Write the calculation result into file

CoastLCCfile.write(str(x)+",")

CoastLCCfile.write(str(y)+"\n")

# Close the file

CoastLCCfile.close()

CoastGeographic.close()