mport arcpy

arcpy.env.workspace = r"C:\GE\Test\_workspace"

arcpy.env.overwriteOutput = True

dem\_file = r"C:\Users\JULFIKAR\Downloads\rasters\_NASADEM\output\_NASADEM.tif"

contour\_shapefile = r"C:\GE\Test\_workspace\output\contour\contour\_lines.shp"

hillshade\_output = r"C:\GE\Test\_workspace\output\hillshade\hillshade.tif"

def create\_contour\_lines(dem, output\_shapefile, interval):

try:

arcpy.sa.Contour(dem, output\_shapefile, interval)

print("Contour lines created: {}".format(output\_shapefile))

except Exception as e:

print("Error generating contour lines: {}".format(str(e)))

def create\_hillshade(dem, output\_hillshade):

try:

hillshade = arcpy.sa.Hillshade(dem, azimuth=315, altitude=45)

hillshade.save(output\_hillshade)

print("Hillshade created: {}".format(output\_hillshade))

except Exception as e:

print("Error generating hillshade: {}".format(str(e)))

if \_\_name\_\_ == "\_\_main\_\_":

try:

arcpy.CheckOutExtension("Spatial")

contour\_interval = 10

create\_contour\_lines(dem\_file, contour\_shapefile, contour\_interval)

create\_hillshade(dem\_file, hillshade\_output)

arcpy.CheckInExtension("Spatial")

except Exception as main\_e:

print("An error occurred: {}".format(str(main\_e)))

import arcpy

import os

arcpy.env.workspace = r"C:\GE\data\input"

arcpy.env.overwriteOutput = True

shapefiles = [shp for shp in arcpy.ListFeatureClasses() if shp.endswith('.shp')]

output\_mxd = r"C:\GE\map\automated\_map.mxd"

output\_pdf = r"C:\GE\Output\_map\islamicUniversity.pdf"

template\_mxd = r"C:\GE\data\templates\blank\_map.mxd"

mxd = arcpy.mapping.MapDocument(template\_mxd)

df = arcpy.mapping.ListDataFrames(mxd, "\*")[0]

def add\_layer(layer\_path, data\_frame):

try:

layer = arcpy.mapping.Layer(layer\_path)

arcpy.mapping.AddLayer(data\_frame, layer, "TOP")

print("Layer added: {}".format(os.path.basename(layer\_path)))

except Exception as e:

print("Failed to add layer {}: {}".format(os.path.basename(layer\_path), str(e)))

def apply\_labels(layer\_name):

try:

layer = arcpy.mapping.ListLayers(mxd, layer\_name, df)[0]

layer.showLabels = True

label\_class = layer.labelClasses[0]

label\_class.expression = "[NAME]" # Change this based on your attribute field for labels

print("Labels applied to layer: {}".format(layer.name))

except Exception as e:

print("Failed to apply labels for {}: {}".format(layer\_name, str(e)))

for shp in shapefiles:

layer\_path = os.path.join(arcpy.env.workspace, shp)

add\_layer(layer\_path, df)

apply\_labels(os.path.basename(shp).replace(".shp", "")) # Apply labels based on the layer name

try:

arcpy.mapping.ExportToPDF(mxd, output\_pdf)

print("Map exported to PDF: {}".format(output\_pdf))

except Exception as e:

print("Failed to export map to PDF: {}".format(str(e)))

try:

mxd.saveACopy(output\_mxd)

print("Map document saved: {}".format(output\_mxd))

except Exception as e:

print("Failed to save map document: {}".format(str(e)))

del mxd