```
import arcpy
import os
import urllib.request
import requests
import zipfile
from datetime import datetime
import sys
from pathlib import Path


aprx = arcpy.mp.ArcGISProject("CURRENT")
arcpy.env.overwriteOutput = True

#Make folder for Lab 2

Lab_2_folder = r"C:\Mac\Home\Documents\ArcGIS\Projects\Lab 2"
os.makedirs(Lab_2_folder, exist_ok=True)
Lab_2_folder

# Download .LAS files from MN DNR

# Make folder for Lidar
lidar_folder = r"C:\Mac\Home\Documents\ArcGIS\Projects\Lab 2\Lidar"
os.makedirs(Lidar_folder, exist_ok=True)

# Download .LAS file from MN DNR
las_url = "https://resources.gisdata.mn.gov/pub/data/elevation/lidar/
examples/lidar_sample/las/4342-12-05.las"
las_file_path = os.path.join(lidar_folder, "4342-12-05.las")

try:
    urllib.request.urlretrieve(las_url, las_file_path)
    print(f"Successfully downloaded LAS file to {las_file_path}")
except Exception as e:
    print(f"Error downloading LAS file: {e}")

# LAS to DEM Conversion
dem_output_path = os.path.join(lidar_folder, "dem_output.tif")
try:
    arcpy.conversion.LasDatasetToRaster(
        in_las_dataset=las_file_path,
        out_raster=dem_output_path,
        value_field="ELEVATION",
        interpolation_type="BINNING AVERAGE LINEAR",
        data_type="FLOAT",
        sampling_type="CELLSIZE",
        sampling_value=1,
        z_factor=1
    )
    print(f"DEM successfully created at {dem_output_path}")
```

```
except Exception as e:
    print(f"Error in DEM conversion: {e}")

# DEM to TIN Conversion
output_tin_path = os.path.join(lidar_folder, "output_tin")
try:
    arcpy.ddd.RasterTin(
        in_raster=dem_output_path,
        out_tin=output_tin_path,
        z_tolerance=10.68,
        max_points=1500000,
        z_factor=1
    )
    print(f"TIN successfully created at {output_tin_path}")
except Exception as e:
    print(f"Error in TIN creation: {e}")

# Export DEM to PDF
pdf_dem_path = os.path.join(lidar_folder, "dem_map.pdf")
try:
    # Create a new map and add the DEM layer
    dem_map = aprx.createMap("DEM Map")
    dem_layer = dem_map.addDataFromPath(dem_output_path)

    # Remove existing ESRI basemaps for clarity
    for lyr in dem_map.listLayers():
        if lyr.isBasemapLayer:
            dem_map.removeLayer(lyr)

    # Access layout and update map frame
    layout = aprx.listLayouts()[0]
    map_frame = layout.listElements("MAPFRAME_ELEMENT")[0]
    map_frame.map = dem_map

    # Export to PDF
    layout.exportToPDF(pdf_dem_path, resolution=300)
    print(f"DEM map successfully exported to {pdf_dem_path}")
except Exception as e:
    print(f"Error exporting DEM map to PDF: {e}")

# Export TIN to PDF
pdf_tin_path = os.path.join(lidar_folder, "tin_map.pdf")
try:
    # Create a new map and add the TIN layer
    tin_map = aprx.createMap("TIN Map")
    tin_map.addDataFromPath(output_tin_path)

    # Get the second layout and update map frame
    layout = aprx.listLayouts()[1]
    map_frame = layout.listElements("MAPFRAME_ELEMENT")[0]
    map_frame.map = tin_map

    # Export to PDF
    layout.exportToPDF(pdf_tin_path, resolution=300)
    print(f"TIN map successfully exported to {pdf_tin_path}")
except Exception as e:
    print(f"Error exporting TIN map to PDF: {e}")

import requests
import os
import zipfile
from pathlib import Path
import arcpy

# Define the folder path
prism_folder = Path("C:/Mac/Home/Documents/ArcGIS/Projects/Lab 2/
Prism")
prism_folder.mkdir(parents=True, exist_ok=True)

# Define download URL =
url = "https://ftp.prism.oregonstate.edu/normals/monthly/4km/ppt/
PRISM_ppt_30yr_normal_4kmM4_all_bil.zip"
output_zip = prism_folder / "PRISM_ppt_30yr_normal_4kmM4_all_bil.zip"

# Download the zip file
if not output_zip.exists():
    response = requests.get(url, stream=True)
    with open(output_zip, 'wb') as file:
        for chunk in response.iter_content(chunk_size=8192):
            file.write(chunk)

# Extract the zip file
with zipfile.ZipFile(output_zip, 'r') as zip_ref:
    zip_ref.extractall(prism_folder)

# Identify all .bil filesd
bil_files = list(prism_folder.glob("*.bil"))

# =Geodatabase path
gdb_path = Path("C:/Mac/Home/Documents/ArcGIS/Projects/Lab 2/
Lab_2.gdb")
if not arcpy.Exists(str(gdb_path)):
    arcpy.management.CreateFileGDB(str(gdb_path.parent),
gdb_path.name)

# Create a Mosaic Dataset - geoprocessing tool
mosaic_name = "Mosaic_Dataset_Prism"
mosaic_dataset_path = gdb_path / mosaic_name
arcpy.management.CreateMosaicDataset(
    in_workspace=str(gdb_path),
    in_mosaicdataset_name=mosaic_name,

coordinate_system="GEOGCS['GCS_WGS_1984',DATUM['D_WGS_1984',SPHEROID['
WGS_1984',6378137.0,298.257223563]],PRIMEM['Greenwich',0.0],UNIT['Degr
ee',0.0174532925199433]]"
)

# Add Rasters to the Mosaic Dataset - geoprocessing tool
arcpy.management.AddRastersToMosaicDataset(
    in_mosaic_dataset=str(mosaic_dataset_path),
    raster_type="Raster Dataset",
    input_path=str(prism_folder),
    update_cellsize_ranges="UPDATE_CELL_SIZES",
    update_boundary="UPDATE_BOUNDARY",
    update_overviews="NO_OVERVIEWS",
    maximum_cell_size=0,
    minimum_dimension=1500,
    duplicate_items_action="ALLOW_DUPLICATES",
    calculate_statistics="NO_STATISTICS"
)

# Add fields for Time and Variable for precipitation data -
geoprocessing tools
arcpy.management.AddFields(
    in_table=str(mosaic_dataset_path),
    field_description="Timestamp DATE # 255 # #;Variable TEXT # 255 #
#"
)

# Calculate Fields - Timestamp
arcpy.management.CalculateFields(
    in_table=str(mosaic_dataset_path),
    expression_type="ARCADE",
    fields=[["Timestamp", "DateAdd(Date(1991,1,1), $feature.OBJECTID -
1, 'years')"]]
)

# Calculate Fields - Variable
arcpy.management.CalculateFields(
    in_table=str(mosaic_dataset_path),
    expression_type="PYTHON3",
    fields=[["Variable", '"Precipitation"']]
)

# Build Multidimensional Info
arcpy.md.BuildMultidimensionalInfo(
    in_mosaic_dataset=str(mosaic_dataset_path),
    variable_field="Variable",
    dimension_fields="Timestamp # #",
    variable_desc_units="Precipitation # mm"
```

```
)

# Create Multidimensional Raster Layer - geoprocessing tool
multi_raster_layer = "Prism_MultidimLayer"
arcpy.md.MakeMultidimensionalRasterLayer(
    in_multidimensional_raster=str(mosaic_dataset_path),
    out_multidimensional_raster_layer=multi_raster_layer,
    variables="Precipitation",
    dimension_def="ALL"
)

# Create Space-Time Cube - geoprocessing tool
space_time_cube_output = prism_folder / "space_time_cube_prism.nc"
arcpy.stpm.CreateSpaceTimeCubeMDRasterLayer(
    in_md_raster=str(mosaic_dataset_path),
    output_cube=str(space_time_cube_output),
    fill_empty_bins="ZEROS"
)

# Animation using multidim raster "Prism_MultidimLayer"
# Set Up Time & variabled on Arc GIS Pro GUI

# Multidimensional Raster: C:\Mac\Home\Documents\ArcGIS\Projects\Lab
2\Lab_2.gdb\Mosaic_Dataset_Prism
# Space-Time Cube : C:\Mac\Home\Documents\ArcGIS\Projects\Lab
2\Lab_2.gdb\space_time_cube_prism
```