Lab 03: USPS - ArcGIS Pro

The scenario:

Reilly and Randy are two USPS drivers. Their boss gave them overtime work for a Saturday to get packages delivered before the holidays. They start at 8am and need to deliver 10 packages to 10 different locations that are scattered around Western Twin Cities.

Your job is to help them find the best 2 routes between their two trucks so that it minimizes the amount of time they have to spend working before a holiday. Provide directions for them that they can print off. They're old school and don't carry smart phones.

This script obtains the locations of the starting point and stops as point features in Minnesota. A network dataset is created for the seven county metropolitan area where the deliveries are to be made and the vehicle routing problem is solved.

This script requires the network analysis toolset available with the network analyst license from ESRI.

In [ ]:
```
import requests
from zipfile import ZipFile
import os
```

Get Network Dataset

In [ ]:
```
roads_url = "https://resources.gisdata.mn.gov/pub/gdrs/data/pub/us_mn_state_metrogis/trans_road_centerlines_gac/shp_trans_road_centerlines_gac.zip"
out_path = r"C:\Users\msong\Desktop\arc2\lab3\data"

r = requests.get(roads_url)
assert r.status_code is 200

with open("roads.zip", "wb") as file:
    file.write(r.content)

with ZipFile("roads.zip", "r") as zipped:
    zipped.extractall()
```

Getting Locations

The goal of this part of the code is to create point features for the starting point of the delivery and the stops. Google API will be used to obtain the coordinates and then those will be turned into point features in UTM ZONE 15N.

In [ ]:
```
def format_findsearch(in_search):
    ''' Remove spaces from find place from text query search location
    and format for google places api url

    Parameter
    ---------
    in_search: str
        address, name, or phone number of search location

    Return
    ------
    out: str
        formatted search location to be used for google places api url
    '''

    out = in_search.replace(",", "")
    out = out.replace(" ", "%20")
    return out
```

In [ ]:
```
start = "1436 Lone Oak Rd, St Paul, MN 55121"
```

In [ ]:
```
stop_addresses = ["5525 Cedar Lake Rd S, St Louis Park, MN 55416",
                  "225 Thomas Ave N, Minneapolis, MN 55405",
                  "701 N 5th St, Minneapolis, MN 55401",
                  "920 E Lake St, Minneapolis, MN 55407",
                  "783 Harding St NE, Minneapolis, MN 55413",
                  "4165 W Broadway Ave, Robbinsdale, MN 55422",
                  "1321 E 78th St, Bloomington, MN 55425",
                  "12547 Riverdale Blvd, Coon Rapids, MN 55448",
                  "9875 Hospital Dr, Maple Grove, MN 55369",
                  "3300 Oakdale Ave N, Robbinsdale, MN 55422 "
                 ]
```

In [ ]:
```
# Obtain lat/long coordinates for start using google places api
key = ""
base_url = "https://maps.googleapis.com/maps/api/place/"
query_type = "findplacefromtext"
out_type = "json"

inputtype = "textquery"
fields = "fields=formatted_address,name,geometry"

place = format_findsearch(start)
url = f"{base_url}{query_type}/{out_type}?input={place}&inputtype={inputtype}&
{fields}&key={key}"

r = requests.get(url)
assert r.status_code is 200
out_search = r.json()
coords = [out_search['candidates'][0]['geometry']['location']['lat'],
          out_search['candidates'][0]['geometry']['location']['lng']]

# Create point feature for starting delivery
# Spatial reference set to GCS_WGS_1984
sr = arcpy.SpatialReference(4326)
pt = arcpy.Point()
ptGeoms = []

pt.X = coords[1]
pt.Y = coords[0]
ptGeoms.append(arcpy.PointGeometry(pt, sr))

arcpy.CopyFeatures_management(ptGeoms,
                              r"C:\Users\msong\Desktop\arc2\lab3\data\start.sh
p")


# reproject to NAD83 15 N
out_coordinate_system = arcpy.SpatialReference('NAD 1983 UTM ZONE 15N')
arcpy.management.Project("start",
                         "start_project",
                         out_coordinate_system)
```

In [ ]:
```
# Obtain lat/long coordinates for each stop using google places api
key = "AIzaSyCYPFFiQg2gvFhLwv17r9FEjJalSiqwNrM"
base_url = "https://maps.googleapis.com/maps/api/place/"
query_type = "findplacefromtext"
out_type = "json"

inputtype = "textquery"
fields = "fields=formatted_address,name,geometry"

out_coords = []
for stop in stop_addresses:
    place = format_findsearch(stop)
    url = f"{base_url}{query_type}/{out_type}?input={place}&inputtype={inputty
pe}&{fields}&key={key}"

    r = requests.get(url)
    assert r.status_code is 200

    out_search = r.json()
    coords = [out_search['candidates'][0]['geometry']['location']['lat'],
            out_search['candidates'][0]['geometry']['location']['lng']]
    out_coords.append(coords)


# Create point features for stops
# Spatial reference set to GCS_WGS_1984
sr = arcpy.SpatialReference(4326)
pt = arcpy.Point()
ptGeoms = []
for p in out_coords:
    pt.X = p[1]
    pt.Y = p[0]
    ptGeoms.append(arcpy.PointGeometry(pt, sr))

arcpy.CopyFeatures_management(ptGeoms,
                              r"C:\Users\msong\Desktop\arc2\lab3\data\stops.sh
p")

# reproject to NAD83 15 N
out_coordinate_system = arcpy.SpatialReference('NAD 1983 UTM ZONE 15N')
arcpy.management.Project("stops",
                         "stops_project",
                         out_coordinate_system)
```

Create Network Dataset

The network dataset will be for the seven county metropolitan area of Minnesota. Some of the functionalities of the Network Dataset must be done in ArcPro manually.

In [ ]:
```
# Make feature dataset where network dataset reside in
arcpy.management.CreateFeatureDataset(r"C:\Users\msong\Desktop\arc2\lab3\usps_
proj\usps_proj.gdb",
                                      "routes_ND",
                                      "PROJCS['NAD_1983_UTM_Zone_15N',GEOGCS
['GCS_North_American_1983',DATUM['D_North_American_1983',SPHEROID['GRS_1980',6
378137.0,298.257222101]],PRIMEM['Greenwich',0.0],UNIT['Degree',0.0174532925199
433]],PROJECTION['Transverse_Mercator'],PARAMETER['False_Easting',500000.0],PA
RAMETER['False_Northing',0.0],PARAMETER['Central_Meridian',-93.0],PARAMETER['S
cale_Factor',0.9996],PARAMETER['Latitude_Of_Origin',0.0],UNIT['Meter',1.0]];-5
120900 -9998100 10000;-100000 10000;-100000 10000;0.001;0.001;0.001;IsHighPrec
ision"
                                     )

# Import road centerlines for seven county metropolitan area into feature data
set
arcpy.conversion.FeatureClassToFeatureClass("RoadCenterline",
                                            r"C:\Users\msong\Desktop\arc2\lab3
\usps_proj\usps_proj.gdb\routes_FD",
                                            "mpls_roads")


# Create Network Dataset from minneapolis roads
arcpy.na.CreateNetworkDataset(r"C:\Users\msong\Desktop\arc2\lab3\usps_proj\usp
s_proj.gdb\routes_FD",
                              "usps_ND",
                              "mpls_roads",
                              "ELEVATION_FIELDS")
```

Within the Network properties of my usps_ND, I created a new time cost which calculated the cost of travel in seconds. I used the equation [Shape_Length]/[SPEED_IMP] * 0.44704. Then I created a new Travel Mode that used impedence time as the impedence. Lastly I also enabled my network properties to output directions.

In [ ]:
```
# Build network after changing the properties
arcpy.na.BuildNetwork(r"C:\Users\msong\Desktop\arc2\lab3\usps_proj\usps_proj.g
db\routes_FD\usps_ND")
```

Vehicle Routing Problem Analysis

In [ ]:
```
# Create vehicle routing problem analysis layer to input data
arcpy.na.MakeVehicleRoutingProblemAnalysisLayer(r"C:\Users\msong\Desktop\arc2
\lab3\usps_proj\usps_proj.gdb\routes_FD\usps_ND",
                                                "Vehicle Routing Problem",
                                                "travel",
                                                "Minutes",
                                                "Miles",
                                                "3/27/2021 8:00:00 AM",
                                                "LOCAL_TIME_AT_LOCATIONS",
                                                "ALONG_NETWORK",
                                                "Medium",
                                                "Medium",
                                                "DIRECTIONS",
                                                "CLUSTER")
```

In [ ]:
```
# Add stops to VRP layer
arcpy.na.AddLocations("Vehicle Routing Problem",
                      "Orders",
                      r"C:\Users\msong\Desktop\arc2\lab3\usps_proj\usps_proj.g
db\stops_project",
                      "Name # #;Description # #;ServiceTime # #;TimeWindowStar
t # #;TimeWindowEnd # #;MaxViolationTime # #;TimeWindowStart2 # #;TimeWindowEn
d2 # #;MaxViolationTime2 # #;InboundArriveTime # #;OutboundDepartTime # #;Deli
veryQuantity_1 # #;DeliveryQuantity_2 # #;DeliveryQuantity_3 # #;DeliveryQuant
ity_4 # #;DeliveryQuantity_5 # #;DeliveryQuantity_6 # #;DeliveryQuantity_7 #
 #;DeliveryQuantity_8 # #;DeliveryQuantity_9 # #;PickupQuantity_1 # #;PickupQu
antity_2 # #;PickupQuantity_3 # #;PickupQuantity_4 # #;PickupQuantity_5 # #;Pi
ckupQuantity_6 # #;PickupQuantity_7 # #;PickupQuantity_8 # #;PickupQuantity_9
 # #;Revenue # #;AssignmentRule # 3;RouteName # #;Sequence # #;CurbApproach #
 0",
                      "5000 Meters",
                      None,
                      "mpls_roads SHAPE;usps_ND_Junctions NONE",
                      "MATCH_TO_CLOSEST",
                      "APPEND",
                      "NO_SNAP",
                      "5 Meters",
                      "EXCLUDE",
                      None)
```

In [ ]:
```
# Add starting point to VRP layer
arcpy.na.AddLocations("Vehicle Routing Problem",
                      "Depots",
                      "start_project",
                      "Name # #;Description # #;TimeWindowStart # #;TimeWindow
End # #;TimeWindowStart2 # #;TimeWindowEnd2 # #;CurbApproach # 0",
                      "5000 Meters",
                      None,
                      "mpls_roads SHAPE;usps_ND_Junctions NONE",
                      "MATCH_TO_CLOSEST",
                      "APPEND",
                      "NO_SNAP",
                      "5 Meters",
                      "EXCLUDE",
                      None)
```

In [ ]:
```
# Create two empty routes because there are two usps fleets
arcpy.na.AddVehicleRoutingProblemRoutes("Vehicle Routing Problem",
                                        2,
                                        "Route",
                                        "Location 1",
                                        "Location 1",
                                        "8:00:00 AM",
                                        None,
                                        5,
                                        None,
                                        None,
                                        "# 1 # # #",
                                        None,
                                        "APPEND")
```

In [ ]:
```
# Add road barriers. These were created by buffering the closed highway center
lines by 1 m then erasing the open road centerlines.
#This allows for routes to pass over the closed highways if there are perpendi
cular roads.
arcpy.na.AddLocations("Vehicle Routing Problem",
                      "Polygon Barriers",
                      r"C:\Users\msong\Desktop\arc2\lab3\usps_proj\usps_proj.g
db\closed_highways_Buffer_Erase",
                      "Name # #;BarrierType # 0;Attr_impedence_time # 1;Attr_L
ength # 1;Shape_Length Shape_Length #;Shape_Area Shape_Area #",
                      "5000 Meters",
                      None,
                      "mpls_roads SHAPE;usps_ND_Junctions NONE",
                      "MATCH_TO_CLOSEST",
                      "APPEND",
                      "NO_SNAP",
                      "5 Meters",
                      "EXCLUDE",
                      None)
```

In [ ]:
```
"""
Before solving the VRP layer, I added time frames for the start and end time
Start time: 3/27/2021 8:00 AM
Latest endtime: 3/27/2021 11:00 AM
Stops 2 and 10 had a specific time window: 10:00 AM - 11:00 AM
"""

arcpy.na.Solve("Vehicle Routing Problem",
               "HALT",
               "TERMINATE",
               None,
               '')
```

In [ ]:
```

```