README

Welcome to the project README! This document serves as a comprehensive guide to utilizing the tools and functionalities provided within this project. Whether you're a seasoned GIS professional or new to the world of GIS and spatial analysis, this README is designed to provide clear instructions and insights into each tool's purpose, parameters, and usage.

The toolbox comprises five different tools, each with an embedded Python script accessible through the Execution tab under the tool's properties. To edit the embedded script, click the 'Open in Script Editor' button within the same tab. This action opens a temporary file in your default Integrated Development Environment (IDE), allowing you to make modifications. Any changes saved in the temporary file will automatically update the embedded script. Additionally, each tool features several parameters, with inputs ranging from layers provided by the user to layers generated as outputs during tool execution.

Slope Raster Reclassification: This tool is straightforward: the input raster becomes the output slope raster. The output slope raster can then have its values reclassified via a ValueTable. For our purposes, different slope ranges between 0 and 30 degrees were reclassified with weights from 1 to 5, while anything steeper than 30 degrees was reclassified with a weight of 100.

- **Input Raster**: Select a raster layer.
- **Output Raster**: New raster layer (ensure it is saved to the desired workspace).
- **Reclassification Values**: The slope raster uses a degree output measurement. In the value table, start and end values create a slope degree range, and the new value is the new weight given to the slope range.

Avoidance Raster Reclassification: This tool reclassifies input features that are meant to be avoided, as well as any specified buffer around said input features, to the same value as the steep slopes from the first tool. In our case, this value was 100. It also reclassifies anything that isn't an input feature or its buffer to 0. However, due to the numerous tools built into the script working behind the scenes, the value of 100 is hardcoded into the script. If the desired weight value for the input features and their buffers in this tool is something different than 100, the script will require manual editing on the user's part. When viewing the script in an IDE, a comment can be seen mentioning that two specific variable values should be changed if desired.

- **Input Features**: Select one or multiple feature layers that the route wants to avoid.
- **Buffer Distances**: Assign buffer values for each input layer (ensure the buffers are in the same order as the input layers so the buffers are assigned to the correct corresponding inputs).
- **Output Raster**: New raster layer (ensure it is saved to the correct workspace).

Least Cost Path: The least cost path tool combines the two raster outputs created by the two previous tools. Weights can be optionally added to each raster. The weight assigned to each raster will simply multiply the raster's values by the weight value assigned to it. A final cost raster is created from the combined input raster layers which are used in the subsequent cost distance and cost path tools within the script. A source and destination point are also required to run those tools. The least cost path has two different outputs: the original straight-line polyline route and the polyline route with the smoothing applied.

- **Input Raster Layers**: Select one or multiple raster layers to use as input.
- **Weights**: Optionally assign a weighting value to the input raster layers. Ensure that they are assigned to the corresponding inputs by writing the weights in the same order the raster layers were inputted.
- **Output Cost Raster**: New cost raster layer (ensure it is saved to the desired workspace).
- **Source Point**: Select a starting point coordinate for the cost distance tool.
- **Destination Point**: Select an endpoint coordinate for the cost path tool.
- **Least Cost Path**: Outputs the two polyline least cost paths, both the non-smoothed and the smoothed route (ensure it is saved to the desired workspace).
- **Smoothing Tolerance**: The algorithm used is "Polynomial approximation with exponential kernel (PAEK). A tolerance must be specified for the PAEK algorithm, and it must be greater than zero. You can choose a preferred unit; the default is the feature unit.

Route Segmentation: The route segmentation tool segments a polyline into straight-line segments based on its vertices. It takes an input polyline and outputs a segmented polyline feature class. When the segmented polyline is created, the symbology will automatically be single symbol. It can be a useful visualization to have one color represent segments less than a certain length, and another color represent segments longer than said length. To accomplish this, right-click on the newly created layer, and click on symbology. An easy way to create the above-described visualization is to first change the primary symbology from 'Single Symbol' to 'Graduated Colors'. From there, change the number of classes to two, and then set the method to 'Manual Interval'. All that is left is to change the upper value to the desired cut off segment length and ensure the two classes have different, distinguishable colors. In our case, we wanted any segments that were less than 2 miles to be red, and any segments that were longer than 2 miles to be green.

- **Input Route**: Select a least cost path polyline to be segmented. This will work much better with the non-smoothed polyline vs. the smoothed polyline.
- **Output Route Segments**: Outputs the segmented polyline as a single symbol feature class (ensure it is saved to the desired workspace).

Report Creation: The report creation tool outputs a basic report containing information regarding the least-cost path length, the start and end coordinates, and the coordinate reference system of the map. If a Layout is made in the project within ArcGIS Pro, an image of the Layout will also be included in the final report. This is an optional tool meant to summarize key components of the potential route created by the previous tools. The script will use the first layout saved within one's project for the report. If using one's own layout, proceed normally. If using the layout file included in this project package, import said layout file then navigate to Maps within the Catalog pane. Open the map associated with the newly imported layout file and manually add the features to be reported on to this new map, before proceeding with the Report Creation tool.
To create the report, one must have the Open Source Python library 'ReportLab' installed to their environment in ArcGIS Pro. This involves going to the Project tab then clicking Package Manager. From here, select Add Package, search for ReportLab, and click Install in the package details pane. In order to install successfully, one may need to clone their environment. To do so, click the gear icon next to the Active Environment, then click Clone and ensure both Source and Destination environment pathways are sufficient before clicking OK. From here, ensure that the new Active Environment is indicated as being the cloned version before installing ReportLab as described above.

- **Input:** Select route from drop down menu or folder that will be featured in the report.
- **Output:** Outputs the final report as a pdf. Choose a name and select the folder in which it will be saved. Ensure the name includes ".pdf" at the end so that it will be saved in the proper format.