# GGS 366: Lab 7

## General guidelines

**Please submit the .ipynb notebook containing all the codes.** Make sure the notebook is properly formatted by following the instructions below.

* Separate answers to each question using text cells and markdown headings. See example syntax here: [Markdown Guide - Colab](https://colab.research.google.com/notebooks/markdown_guide.ipynb)
* Write the essay questions in your Python notebook using text cells.
* The function body (where appropriate) must include a docstring to provide a general description of the function.
* Each step of the code needs to be commented.
* The code needs to be properly indented and readable. For more on formatting guidelines: [PEP 8 – Style Guide for Python Code | peps.python.org](https://peps.python.org/pep-0008/) and [PEP 257 – Docstring Conventions | peps.python.org](https://peps.python.org/pep-0257/)

You may use GenAI as a supporting tool. However, directly copying code from GenAI will be considered plagiarism and hinder your learning process. This can negatively impact your performance in course exams and, ultimately, your professional success. Therefore, it is in your best interest to thoroughly understand the fundamental concepts and make a genuine effort to solve the problems on your own.

## Question 1 (10 points)

* Get the network graph for **"Arlington, Virginia, United States"**, considering **bike** network type.
* The speed data of OSM network is mainly for driving, which is not applicable for bike routing. Instead, for each edge in the network, estimate a variable ***“travel\_time\_min”*** that calculates travel time in minutes for biking along each edge, considering a **fixed biking speed of 20 km/h**. Make sure to use projected CRS and corrected length values for these travel time estimations. Next, **update the network graph**.

## Question 2 (10 points)

* Using **this address** (414 N Fillmore St, Arlington, VA 22201), run a for loop that does the following
  + Display **isochrones of 5, 10, and 15 minutes** into one plot, with the network graph in the background
  + Print the **total network length** for each isochrone network of different time thresholds.
  + Print the **total isochrone area**, assuming anyone biking through this isochrone network can access 50-meter buffer areas around the isochrone network.
* How would you **interpret the isochrones**?

## Question 3 (5 points)

* Extract all the **school footprints** in the study area using OSMnx and calculate their **centroid**.

## Question 4 (10 points)

* In this question, we will calculate **travel times between multiple origin-destination pairs**.
* First, take a sample of the **first four schools** in the school centroid dataset.
* Create **unique origin-destination pairs for these four schools**. Hint: You may use the following function to create unique pairs of their indexes first and then retrieve the coordinates of the respective schools as origin and destination locations.
  + from itertools import combinations
  + comb = list(combinations([0, 1, 2, 3], 2))
  + comb
  + Output should be like: [(0, 1), (0, 2), (0, 3), (1, 2), (1, 3), (2, 3)]
* Create **an empty list**: *shortest\_path*
* Create a for loop that calculates the **shortest route (weighted by their travel time) and travel time** for each origin-destination pair and append the route to the *shortest\_path* list.
  + Hint: use this **code to calculate total travel time**:
    - int(sum(ox.routing.route\_to\_gdf(G\_new, route, weight="travel\_time\_min")["travel\_time\_min"]))
  + At each iteration, the for loop should **print the following**:
    - The travel time between origin school *index\_value* and destination school *index\_value* is *total\_travel\_time* minutes.
* Create a visualization that displays the **routes stored in the *shortest\_path* list using distinct colors**, with the network graph in the background.