COSC 311 - Lab 2

Dr. Joe Anderson

Due: 17 September 2020

1 Objectives

1. Gain familiarity with the Python programming language
2. Use the OOP aspects of Python
3. Do some basic data plotting and visualization

2 Tasks

1. Write a Python class called `Vector` to represent a 2-dimensional point/vector, with `x` and `y` coordinates
   (a) Include methods to overload addition and subtraction with other vectors
   (b) Include a method to compute the dot product with another vector
   (c) Include a method to compute the distance to another vector
2. Write a Python class called `Geometry` that represents a polygon with several vertices and faces, given by vectors.
   (a) You will add a face to the geometry by adding both of its constituent vectors. So when you add a face, check to see if you're connecting to a vertex that is already present!
      i. One way to make the "bookkeeping" simple is to store the vertices and faces separately: maintain a list or dictionary of vertices and then a list of pairs of those vertices, where a pair represents a face.
      ii. Include a method to add a face to the geometry. You do not need to worry about removing or "splitting" faces
      iii. Include a method to check that the geometry is "closed" – the faces create a cycle that start and end at the same vertex.
   (b) Write a method to draw the outline of the shape with `matplotlib`.
   (c) Write a method to determine if a given point is inside the geometry.
      i. To do this, there is a nice simple geometric algorithm:
      ii. Consider the query point, then "cast a ray" outward from that point in any direction (straight to the right will the be the easiest to program).
      iii. Count the number of faces that the ray passes through: if it is odd, the point is inside the shape; if even, it lies outside.
      iv. Two special cases to consider:
         A. If the ray passes along a face (the body must be non-convex for this), don't count it as an intersection;.

   B. If the ray passes through a vertex exactly, only count it as a single intersection (not both faces that meet at that vertex).

3. Test your `Geometry` class and its methods on various shapes: triangle, rectangle, star, etc. Be creative! Show some examples of the interior/exterior testing (you do not have to illustrate/visualize the ray-casting process).

3 Submission

Zip your source files and upload them to the assignment page on MyClasses. Be sure to include all source files, properly documented, a `README` file to describe the program and how it works, along with answers to any above discussion questions.