Nicholas Zessoules
Topological Sort

A topological sort is the linear ordering of a directed graph's vertices with respect to an order of precedence. The function topological_sort is originally accessed via a call to traverse_graph. A switch allows for a variation in sort-type, allowing the option of Breadth First and Depth First as well.

Once called, the topological_sort will allocate space for an array of numbers, while initializing it with all zero entries. This array will be used to catalogue the amount of local parents (local_par) each vertex has. Next, using a repetitive call to edge_iterator, the function is able to properly catalogue these values in the array. Once the array is properly filled the sorting system will implement through a while loop. This loop is kept in check by filling in the spot of a recited vertex with a -1 value, ensuring that when completed the sum of the elements of the array will equal the negative value of the total number of vertices in the graph. To combat the issue of vertices with the same precedence and index is used. The index is set to the number of the highest vertex that still has a parent. The edge_iterator is now called again with the index vertex. Every child found of the vertex will have its value decremented, where the value of the vertex will be set to -1. Lastly the vertex will be printed through a call to the passed function write_vertex. The while loops repeats until all values of the array are -1.

I felt that this was a good example of the type of a data structure puzzle. This also heavily re-enforced the concept of a graph and the use of its primitives. Most of the time spent on this algorithm was off screen, doodling pictures on paper and in my head. As an estimate seven hours where spent thinking about the problem; three passively thinking, two actively thinking, and two in front of a screen. Once sitting down to the computer I was surprised at how little code was needed for a successful sort.