**Thought Criminal Handling Software Package**

* Design
  + Structure:
    - We first build a Graph of Nodes.
      * Graph objects contain a dictionary that map node names to nodes.
      * Nodes represent individual users.
        + They contain connections, sent message DPs and received message DPs.
    - Then we query against this graph of nodes:
      * Split input into a node name, a rule and a distance,
      * Process the rule into a relation, and operator type, and a boundary.
      * Explore the graph based on the node and maximum distance.
      * Report nodes that were explored and meet relation/operator/boundary rules.
* Opportunities for Improvement
  + Nodes can maintain max and min DP for sent and received messages. This would allow for faster lookup, but the full sets are also necessary for equality checks. Depending on the python implementation of a set(), this may already be handled.
  + Json or other format could be used for processed graph storage instead of Pickle, because pickle storage is very space-inefficient.
* Testing Options:
  + Unit Testing:
    - Nodes:
      * Test connection processing functionality.
        + Test that dissident probabilities and connections are maintained appropriately.
    - Graphs:
      * Test node retrieval.
      * Test node set retrieval.
      * Test graph saving.
    - String Parsing:
      * Test exit functionality
      * Test functionality to split into node, operator, distance.
      * Test node parsing and retrieval functionality (case sensitive).
      * Test operator parsing and application functionality (case insensitive).
      * Basic test for distance functionality.
  + System Testing:
    - Test graph is constructed with correct nodes in simple setup with hard coded correct solution
      * Test correct nodes exists.
      * Test connections.
      * Test DPs per node and per connection type.
    - Test network correctly loads graph if exists, or builds otherwise
    - Test parsing system end to end from input string to graph function call
      * Include types of incorrect inputs that should fail.
    - Test graph parsing system from function call to nodes returned, including cases:
      * dp >1, dp < 2
        + (Currently system does not handle inf as user input due to regex match).
      * dp < negative, dp > negative
      * Negative distance, zero distance, distance greater than total number of nodes
        + Currently breaks due to maximum recursion depth in python for excessively high depths.
      * Node disconnected
        + Isolated within graph with pointer only to itself.
        + isolated within small subgraph not connected to major graph