Relationship Graphing Language – RGL

Project Manager: Elisheva Aeder (ea2621)

Language Guru: Nicholas Ray Falba (nrf2118)

System Architect: Johan M. Mena (jmm2371)

Verification & Validation: Jialun Liu (jl4347)

# Motivation

In today’s world there is an escalating interest in relationships. Be it the connection of people on a social networking platform, of family members in a family tree, of variables in a mathematical equation, or the connection of trains in a city subway system, we are consistently trying to find networks of people and things and analyze how they interrelate. Most often, relationships are implemented in a programming setting via the graph data structure containing a set of nodes and edges defining connections between the nodes. In standard programming languages, however, graphs can be tedious to create and manipulate, requiring the creation of separate classes for nodes and edges, and burdening the programmer to keep track of their graphs manually. Finding and analyzing relationships between nodes of a network can be challenging. We therefore created [insert name here], a language that provides high level abstraction to create, maintain, and manipulate graphs, with a specific focus on the relationship between the nodes.

Our Relational Graphing Language has the node, edge and graph as built-in data types to assist a programmer in the simplistic construction and maintenance of graphs. When one node is connected to another, the relationship between them is named and specific to those nodes. The graphs can be bidirectional in that there can be two different relationships between two nodes depending on the direction of the relationship. For example, node A has the relationship of Parent to node B, while node B has the relationship of Child with node A. The storage and maintenance of the connections and node data occurs under the hood, without the programmer’s need to create a data structure with which to implement the graph. This alleviates the difficulties of developing networks of nodes and eases the creation of algorithms to manipulate them.

# Syntax

# Sample Code