Language Specification/Capability

Specifications the user can make
The user will be able to specify an object type, such as satellite, planet, moon, or star. They will be able to choose from some predefined bodies which are listed below.

- Earth
- Moon
- Sun

The user can then specify the orbit type that the object is in or select a predefined orbit type. The predefined orbit types are listed below.

- LEO (Low Earth Orbit)
- Circular
- GEO (Geosynchronous)
- Polar
- Sun-Synchronous

A lifetime of the object, time spent in the orbit.

Orbital Parameters for a user-created orbit

Output the user can request.

- Orbital velocities.
- Orbital periods.
- The time spent in the shadow or sun.
- Current position or the distance traveled.
- The closest or furthest point in between two different bodies.

Possible additions based on time.

- Transfer orbits.

Error Checking
Within GRUE-42 there are certain variables that must be defined for the functionality of the language to work correctly. These variables are checked by using a chain to go through the tree and reset the add the value to the StringTable whenever one of the variables is defined. Then after all of the definitions have been read in, the required variables indexes are checked. If the indexes are still zero, the equivalent of NULL, then an error message is printed notifying the user that more information is required.

There is a special case where there are four variables that can be defined in GRUE-42, but only two of them must be defined for the language to work correctly. This is accomplished by writing an accompanying C function to do the checking. This is done by sending the indexes for each of the parameters and checking to see if they are zero. If two or more are not zero, which is the defined state, then a true value is returned and no action is taken. However, if less than two are defined a false value is returned and an error message is printed notifying the user that more input is required.
This form also handles redefinitions of variables in an elegant way. Since the StringTable is updated using the chains, every time a variable is redeclared the old StringTable value is overwritten. Thus, if the user defines a variable multiple times the last definition of the variable is the one used by GRUE-42.

**Name Analysis**

In GRUE-42 a user can define different calculations to be done. However, each calculation requires at least the name of either a satellite or a planet that has been defined in order to run properly. Thus the name of the satellite and/or planet needs to be defined before the calculations can be made. To make sure that the name was defined, the symbol NameUse inherited ChkIdUse which is a predefined in ELI. This automatically checks to see if the name that is stored for NameUse is defined elsewhere in the tree.