Getting Started

1. Download and unzip Boogie binaries from http://boogie.codeplex.com/. This folder includes the Dafny compiler called **Dafny.exe**.

2. Depending on where/how you unzipped Dafny, Windows may not allow Dafny to load certain DLLs. If you get a strange exception regarding loading of DLLs, then place the **Dafny.exe.config** file in the same folder as Dafny.exe. This config file explains to Windows that it is safe to run Dafny.

3. I have included the a Dafny library for the Celestial Units that only uses the integers. It consists of the data structures **RightAscension**, **Declination**, and **CelestialPoint**. It also contains the helper data structure **Trig** for computing angular distance. You shouldn't use the Sin/Cos/Acos functions in Trig, as these are scaled in a funny way to compute angular distance. The result of angular distance is in tenths of degrees. For example, 234 means 23.4 degrees. See the file **Main.dfy** for an example of using this library.

4. To verify and compile your Dafny program just list all the required files from the command line:

   ```
   Dafny.exe YourFile.dfy Main.dfy Units.dfy Trig.dfy
   ```

   The result will be either the file **YourFile.exe** or **YourFile.dll** depending on whether you compiled with Main.dfy. However, if the program does not verify, then nothing will be compiled (unless you force compilation with the switch `/compile:2` .)

Some Language Notes

1. There are no default constructors. Instead, any method can be used to create an initialize a new object with the syntax:

   ```
   var obj := new Class.Method(arg1, arg2, ...);
   ```

2. Functions are different from methods and cannot modify state. See example in the library of how functions are declared. Functions are ghost by default. Use the keywords `function method` to declare a function that is not a ghost.

3. Inside of functions you must use conditional *expressions* as opposed to conditional *statements*: `if (cond_1) then expr_1 else if (cond_2) then expr_2 ... else expr_n`

4. For assignment use "`:=`" instead of "=".

5. Must write framing conditions using `modifies`, `reads`, `old`, and `fresh`. Please see this paper for a very nice overview:

   http://research.microsoft.com/en-us/um/people/leino/papers/krml203.pdf

6. Current parser does not support nesting of the `new` operator. This expression does not work:

```
new CelestialPoint.Init(
   new RightAscension.Init(7,14,0),
   new Declination.Init(60,20,0))
```

   Instead, break this into several small steps:

```
var ra1 := new RightAscension.Init(6,0,0);
var dc1 := new Declination.Init(-45,0,0);
var p1  := new CelestialPoint.Init(ra1, dc1);
```