Abstract Algebra Theorem Verification

In this project, we constructed Alloy models of several algebraic structures and used them to verify theorems of abstract algebra. We made three sets of goals for this project:

- **Foundation**: Our foundation goals come from the model of a group and certain elementary properties of the group. We did not need support for subgroups to complete these, so they can be verified without the need for higher-order quantification.
- **Target**: Our target goals require support for subgroups, which requires higher-order quantification. We used Alloy* to approach this, and were able to verify all stated theorems for small groups. We did encounter issues regarding performance - the number of variables explodes as the size of the model does.

  For this reason, we made a choice to include a hard-coded model of the alternating group of degree 5 (60 elements, 3600 elements in the multiplication table), which is the smallest counterexample to a converse of the Feit-Thompson theorem. We also hard-coded the multiplication table of the cyclic group of order 60 in order to check that it would find a normal subgroup, to make sure our check of simplicity of A_5 was not over-constrained.

  We found big performance issues whenever we used prime or divisibility, since Alloy isn't the brightest bulb in the box and needs to check every integer versus every other integer to determine these properties, so we hard-coded a list of primes and a divisibility table to improve performance. Specifying that an integer's divisors are all less than it doesn't help performance, since Alloy needs to check each integer with each other integer to even determine size.
- **Reach**: To expand on our model of a group, we included definitions for rings and fields, which add additional structure and have other theorems associated with thems.

  We selected some theorems we found to be interesting about rings and verified them. Among them was a homework problem from a classic algebra textbook, about which the author wrote "that one exercise in his book gave rise to more correspondence from readers than all the other items put together". (http://mural.maynoothuniversity.ie/6982/1/SB-variations.pdf)

  We also learned a lot about rings while doing this part, because we kept checking assertions that we thought would be true and then alloy found counterexamples.

We placed a great deal of focus on verification: the `groups.als` includes 18 different assertions that test for under-constraint and over-constraint of our model. The checks for under-constraint are statements of various theorems that we have verified to be true. The checks for over-constraint are checking that if we know property A implies B but they are not equivalent, then we check that we can find groups that satisfy B and not A. The same is true with `rings.als` - we have fewer assertions because it is part of the reach goal, but we still made sure to make sure we weren't overconstraining by checking that rings exists that aren't fields, etc.

What's in the model? We have `groups.als` and `rings.als` where we made the respective models. Each file first has the sigs of elements and group/ring, and then we list out the facts of the axioms that constrain groups and rings. We also have `a5.als` where we hardcoded the multiplication tables for the groups A_5 and Z_{60}. To run this, make sure to set the memory allocations in Alloy* to the max settings and then uncomment which group you want to check if it is simple. Each source file contains comments detailing design decisions and advice for executing statements.

Some interesting models Included in the handin is the folder `instances` containing some xml files for groups we found to be interesting. These include A5, discussed above, A4, which is the smallest counterexample to the converse of Lagrange's Theorem, and the quaternion group.