TEACHING ABSTRACT ALGEBRA WITH GAP[1]

Julianne G. Rainbolt
Saint Louis University
Department of Mathematics
221 North Grand Boulevard
Saint Louis, MO 63103
rainbolt@slu.edu

This talk began with providing resources for teaching abstract algebra with computer software. This was followed by a discussion on reasons to use software in an upper division mathematics course. Then several examples of projects for an undergraduate abstract algebra class that use the software GAP were provided.

The following web sites are some resources for teaching abstract algebra with software:

1) A web site that contains links to materials for using computer software in an abstract algebra class: `http://www.central.edu/AlgebraResources/`. This web page is Allen Hibbard's of Central College, Iowa.
2) The lab manual *Abstract Algebra with* GAP by J. Rainbolt and J. Gallian contains a large collection of exercises and projects that use the software GAP [1]. The preliminary 2004 version can be found at
`http://euler.slu.edu/Dept/Faculty/rainbolt/manual.html`.
The 2002 version is at
`http://college.hmco.com/mathematics/gallian/abstract_algebra/5e/students/gap.html`.
3) The GAP software home page is `http://www-gap.dcs.st-and.ac.uk/~gap/`. GAP can be obtained for free here. This web page also includes a help manual and tutorial.

GAP as a Pedagogical Tool. Software in upper division mathematics can be a useful pedagogical tool just as it is in lower division mathematics. The following are some of the ways software can be a pedagogical tool in abstract algebra:

1) The software can be used as a fancy calculator and thus eliminate long calculations. For example, it can compute all the conjugates of an element in a large group.
2) The software can provide the students larger and more complicated examples than could realistically be done by hand.
3) Students can use the software to write simple computer algorithms. This helps to solidify abstract algebra concepts.

[1]Partial support for this talk was provided by the National Science Foundation's Course, Curriculum and Laboratory Improvement Program under grant DUE-0127361.

4) The software can be used to produce large amounts of data which the students use to formulate conjectures.
5) Computer projects naturally allow the students to work in collaboration.

Examples of Projects. The remainder of this article consists of several examples of projects that use the software GAP and are appropriate for a first course in abstract algebra. These examples are based on material in *Abstract Algebra with* GAP [1].

Example 1 - Polynomial Rings. GAP will allow you to set up polynomial rings. For example the following GAP command creates the polynomial ring `P1`= $\mathbf{Z}_7[x]$:

```
gap> P1:= PolynomialRing(Integers mod 7);;
```

Suppose we want to factor the polynomial $x^2 - 2 \in \mathbf{Z}_7[x]$. The nonzero elements in $\mathbf{Z}_7$ are denoted in GAP by powers of `Z(7)` where `Z(7)` denotes a generator of the cyclic group of nonzero elements in $\mathbf{Z}_7$:

```
gap> Elements(Integers mod 7);
[ 0*Z(7), Z(7)^0, Z(7), Z(7)^2, Z(7)^3, Z(7)^4, Z(7)^5]
```

The command:

```
gap> x:= X(Integers mod 7, "x");;
```

creates the indeterminate `x` over the ring $\mathbf{Z}_7$. We can now set up a polynomial in the ring `P1` and factor it:

```
gap> Factors(x^2-2);
[ Z(7)+x, Z(7)^4+x ]
```

Once the students are given the above GAP commands they can be given the following exercise.

Student Exercise: Use GAP to factor $x^{p-1} - 1$ in $\mathbf{Z}_p[x]$ for $p = 3, 5, 7$ and 11. Make a conjecture about the factors of $x^{p-1} - 1$ in $\mathbf{Z}_p[x]$ for any prime p.

Solution: The point of this exercise is to lead the students to the conjecture that $x^{p-1} - 1$ factors into $p - 1$ distinct linear factors in $\mathbf{Z}_p[x]$. This in turn leads to the realization that every nonzero element of $\mathbf{Z}_p$ is a zero of $x^{p-1} - 1$. The following would be the student work for the prime $p = 7$:

```
gap> z:= X(Integers mod 7, "z");;
gap> Factors(z^6-1);
[ Z(7)^0+z, Z(7)+z, Z(7)^2+z, -Z(7)^0+z, Z(7)^4+z, Z(7)^5+z ]
```

Example 2 - Sylow Theorems. The purpose of this example is to have the students discover that the number of p-Sylow subgroups is congruent to 1 mod p. GAP can be used to list the Sylow subgroups in a given group. For example the following series of commands provides the 3-Sylows in S_4.

```
gap> G:= SymmetricGroup(4);; H:= SylowSubgroup(G,3);;
gap> ConjugateSubgroups(G,H);
[ Group([ (1,2,3) ]), Group([ (1,3,4) ]), Group([ (2,4,3) ]),
  Group([ (1,4,2) ]) ]
```

From the above output we see that S_4 has four Sylow 3-subgroups. The first Sylow 3-subgroup in the list is the subgroup of S_4 generated by the permutation $(1, 2, 3)$.

Student Exercise: Find all the Sylow p-subgroups of S_5 for every prime p that divides the order of S_5. Repeat this exercise for the groups A_6, S_7 and for the cyclic group of order 60. Make a conjecture about the number of Sylow p-subgroups of a group mod p.

Solution: The student work for the group A_6 and the prime 5 would be:

```
gap> a6:= AlternatingGroup(6);; H:= SylowSubgroup(a6,5);;
gap> Size(ConjugateSubgroups(a6,H));
36
```

After producing all the data required the students can anticipate that the number of Sylow p-subgroups is equal to 1 mod p.

Example 3 - Finite Simple Groups. We will use GAP to help us show A_6 is a simple group. We will need the command `ConjugacyClasses(G)`. For example the following is a list of all the conjugacy classes in A_4:

```
gap> ConjugacyClasses(AlternatingGroup(4));
[ ()^G, (1,2)(3,4)^G, (1,2,3)^G, (1,2,4)^G ]
```

For $a \in G$, the notation $a\hat{}G$ above means the set of all conjugates of a in G. The command `ConjugacyClass(G,a)` creates the conjugacy class of G containing a:

```
gap> ConjugacyClass(AlternatingGroup(4),(1,2,3));
(1,2,3)^G
```

Student Exercises:
1. Suppose you have disjoint sets T, U, V, W, X, Y and Z with cardinalities 1, 40, 40, 45, 72, 72 and 90 respectively. Suppose H is a set that is formed by taking the union

of T with one or more of the other sets. List all the possible cardinalities of H. Which of these answers divide 360?
2. Use GAP to find all the conjugacy classes of A_6 and their cardinalities.
3. Let G be a group and H a normal subgroup of G. Let $h \in H$. Show the conjugacy class of h is a subset of H.
4. Use Exercises 1 - 3 to prove that A_6 is simple.

Solution: In Exercise 1 the possible cardinalities are 1, 41, 46, 73, 81, 86, 91, 113, 118, 126, 131, 136, 145, 153, 158, 163, 171, 176, 185, 190, 198, 203, 208, 216, 225, 230, 235, 243, 248, 270, 275, 280, 288, 315, 320 and 360. Only 1 and 360 divide 360. In Exercise 2 the students discover that the conjugates classes in A_6 have the same sizes as the sets given in Exercise 1. The GAP work for the beginning of Exercise 2 might be the following:

```
gap> ConjugacyClasses(AlternatingGroup(6));
[ ()^ G, (1,2)(3,4)^G, (1,2,3)^G, (1,2,3)(4,5,6)^G,
(1,2,3,4)(5,6)^G, (1,2,3,4,5)^G, (1,2,3,4,6)^G ]
gap> Size(ConjugacyClass(AlternatingGroup(6),(1,2)(3,4)));
45
```

After the short proof in Exercise 3, Exercises 1 - 3 can be combined to provide a proof that A_6 is simple. Suppose H is a normal subgroup of A_6. By Exercise 3, H is a union of conjugacy classes of A_6. Since H is a group it must contain the conjugacy class consisting of only the identity. Thus, by Exercises 1 and 2, the possible orders for H are 1, 41, 46, 73, 81, 86, 91, 113, 118, 126, 131, 136, 145, 153, 158, 163, 171, 176, 185, 190, 198, 203, 208, 216, 225, 230, 235, 243, 248, 270, 275, 280, 288, 315, 320 and 360. But the order of H must divide the order of G. Thus $|H| = 1$ or 360. Thus H must be either all of A_6 or just the identity. Thus A_6 is simple. This series of exercises can be easily adjusted to show A_5 is simple. Also they lead naturally to the proof that A_n is simple for $n \geq 5$. (This example was adapted from a set of exercises created by Christine Stevens at Saint Louis University.)

Example 4 - Generators and Relations. Groups defined using generators and relations can be easily created in GAP. If you want to create an n-generated group, start with a free group on n generators. Then create the group by moding out by the relations. For example, let D_n denote the dihedral group of order n. D_8 is a 2-generated group with relations $a^4 = b^2 = (ab)^2 = e$, where a and b are the generators. The following creates the group D_8 by first defining the group `f` which is the free group on 2 generators and then moding out by the appropriate relations:

```
gap> f:=FreeGroup(2);;
gap> d8:= f/[f.1^4, f.2^2, (f.1*f.2)^2];;
```

The first command above creates a free group with two generators. The element `f.1` denotes the first generator and `f.2` denotes the second generator in the group `f`.

Student Exercise: Let $G = < a, b \mid a^3 = b^3 = (ab)^2 = e >$. Use GAP to help determine to which familiar group G is isomorphic.

Solution: First set up the group G in GAP and then determine its order:

```
gap> f:=FreeGroup(2);; G:= f/[f.1^3, f.2^3, f.1*f.2*f.1*f.2];;
gap> Size(G);
12
```

Since G is of order 12, it must be isomorphic to one of $\mathbf{Z}_{12}, \mathbf{Z}_6 \oplus \mathbf{Z}_2, D_{12}, A_4$ or Q_6 where Q_6 is the group $\{a, b \mid a^6 = e, a^3 = b^2, ba = a^{-1}b\}$.

```
gap> IsAbelian(G);
false
```

Thus G must be isomorphic to one of D_{12}, A_4 or Q_6. To find the order of each element in G we can use a subroutine on the lab manual web site called `orderFrequency`. (See [1].) When this new file is read into GAP it can then be used to list the number of elements of each order in a finite group. For example we see below that G has 1 element of order one, 3 of order two and 8 of order three:

```
gap> Read("orderFrequency");
gap> orderFrequency(G);
[ [ 1, 1 ], [ 2, 3 ], [ 3, 8 ] ]
```

As the dihedral groups contain a large number of reflections, clearly D_{12} has more than three elements of order 2. So G must be isomorphic to either A_4 or Q_6.

We can now compare this output to the order of each element in A_4 and Q_6.

```
gap> orderFrequency(AlternatingGroup(4));
[ [ 1, 1 ], [ 2, 3 ], [ 3, 8 ] ]
gap> q6:= f/[f.1^6, f.1^3*f.2^-2, f.2*f.1*f.2^-1*f.1];;
gap> orderFrequency(q6);
[ [ 1, 1 ], [ 2, 1 ], [ 3, 2 ], [ 4, 6 ], [ 6, 2 ] ]
```

Since A_4 is the only non-Abelian group that has the same number of elements of each order as G, we must have that G is isomorphic to A_4.

The lab manual *Abstract Algebra with* GAP includes more details on the above examples as well as a collection of additional abstract algebra exercises that use GAP. [1]

Reference:

[1] J. G. Rainbolt and J. A. Gallian, *Abstract Algebra with* GAP, Houghton Mifflin, 2002, Used with Permission. `http://college.hmco.com/mathematics/gallian/abstract_algebra/5e/students/gap.html`