AIMS DMG Exercises 10
AIMS 2013-14: Designs, Matroids and Graphs

Rob Beezer
University of Puget Sound

Nancy Neudauer
Pacific University

January 10, 2014

Exercise 1. From the matroid version of the Fano Plane we can construct a "Graph of the Day".

1. Build the matroid version of the Fano Plane with the Sage constructor `matroids.PG(2,2)`.
2. The bases of this matroid are all the 3-sets of the 7-set which are not lines of the projective geometry version (or not blocks of the design version). Obtain these sets and save them in a list. Be sure to retain them as "frozenset"'s.
3. Construct a graph whose vertices are the bases you just computed and saved. Join two vertices with an edge if and only if the sets are disjoint, as described next.
4. You can compute the intersection using the `.intersection()` method, but you will need to convert the result with the `list()` function to check for disjointness.
5. You can construct the graph by simply giving the `Graph()` constructor a list of pairs of adjacent vertices.
6. Your graph should be the Coxeter graph, which is built into Sage. Build the Coxeter graph and then use the `.is_isomorphic()` method to check that your graph really is the Coxeter graph.

Full marks for all your code in a single Sage cell, which when evaluated returns true from the isomorphism check in the last step. You can do this with five lines of Sage code, each corresponding to one of the steps above.

Exercise 2. This exercise asks you to construct the finite geometry known as an affine plane. Specifically, $AG(2, 17)$.

1. Create $F = GF(17)$ with the Sage command `FiniteField(17)`.
2. Use a Python list comprehension to make a list of all the pairs of elements from F. These are the points of the geometry.
3. Consider all 17^2 possible lines, including the vertical ones. In other words, these lines have the form $y = mx + b$ for the various choices of m and b from F.
4. For each such line, compute all the pairs (x, y) that lie on the line. (You should have 17 each time.) This is the geometry.
5. Now we will realize the geometry as a design. Each line is a block of the design. You should have 17^2 blocks, each of size 17.

6. Here is the hardest part of this exercise. Convert your elements of the geometry to the integers 0 through $17^2 - 1$. The Python `.index()` command could be a useful way to go. Use your conversion process to convert the elements of each block.

7. Now provide all your blocks to the Sage constructor `BlockDesign()`.

8. If all is well, the design will be created. Check it with the `.is_block_design()` method and see if it returns the parameters you expect. Just what parameters are you expecting? Give that some thought *before* doing this final step. (This query can take a while to run, so be patient.)

This is a significant programming exercise and will require some effort. Build up your computations slowly. In other words, get each of the above steps working one by one.

To receive full marks submit your computation all in one cell, which can be executed to return just the output of the `.is_block_design()` computation.

Extra credit: set `p=17` as the first line of your program, and then never hard-code 17 again. In this way, you have a program which can be re-run for other values of `p` with just a single edit.