Last name ______________________

First name ______________________

LARSON—MATH 750–SAGE WORKSHEET 03
Posets!

1. Create a Sage/Cocalc account.

   (a) Start the Chrome browser.

   (b) Go to `http://cocalc.com`

   (c) Login. You should see an existing Project for our class. Click on that.

   (d) Click "New", then "Worksheets", then call it **s03**.

   Graphs in Sage

2. To create the (built-in) Petersen graph is Sage and call it "pete", evaluate (*run*): `pete=graphs.PetersenGraph()`, and then to see what the graph looks like evaluate `pete.show()`

3. To find a maximum independent set in a graph g, run `g.independent_set()`. We haven't defined any graph named g yet, but we have pete, so try it for pete.

4. To find the independence number for a graph g, run `g.independent_set(value_only = True)`. We haven't defined any graph named g yet, but we have pete, so again try it for pete.

5. We'll want to construct a variety of subgraphs of a given graph. To get the subgraph *induced* by a subset S of vertices, use: `g.subgraph(S)`.
   Evaluate: `pete.subgraph([0,1,2,3,4])`.

6. If I want to see what this subgraph looks like I can give it a name and `show` it. Run `H=pete.subgraph([0,1,2,3,4])`. Then run: `H.show()`.

7. To check if a graph g is connected use `g.is_connected()`. Check if pete and H are connected.

8. We will want to generate subgraphs of various kinds. How can we generate all connected induced subgraphs of pete using the vertices $[0, 1, 2, 3, 4]$?

   Posets in Sage

   We can create a poset in Sage using the `Poset` *constructor*. The inputs are a set of objects and a (reflexive, anti-symmetric, transitive) relation on those objects. For these purposes a relation is a 2-variable function over those objects that returns a boolean (so `True` for *comparable* and `False` for *incomparable*). We'll look at our two prototype examples and view them, and calculate various things we've talked about in class.

9. To construct the poset P of all subsets of $[4] = \{1, 2, 3, 4\}$, run: `P=Poset((Subsets(4),lambda S,T: S.issubset(T)))`. To see the Hasse diagram, run `view(P)`.

10. To generate the collection of anti-chains, run: `P.antichains()`. Sage returns a *generator*. How can we see them or get more information?

11. To get the list of all maximum anti-chains, run: `P.maximal_antichains()`.

12. We see what the width is, but can also ask Sage. Run: `P.width()`.

13. To generate the collection of chains, run: `P.chains()`. Again Sage returns a *generator*.

14. To get the list of all maximum chains, run: `P.maximal_chains()`. We see what the height is, but can also ask Sage. Run: `P.height()`.

15. We can easily generate families of set with any properties we want. Let's generate the family F of subsets of a 5 elements set with between 2 and 4 elements. Run: `F = [S for S in Subsets(5) if 2 <= len(S) <= 4]`.

16. And let's generate the poset of F with respect to inclusion. Run: `P2=Poset((F,lambda A,B: A.issubset(B)))`.

17. Now `view` this poset, find the maximum anti-chains, maximum chains, height and width.

18. **Bonus**. Use Sage to find a minimum anti-chain partition of either of these posets.

19. Now let's generate and example of a divisibility poset. Run: `P3=Poset(([2..10], lambda x,y:  x.divides(y)))`.

20. Now `view` this poset, find the maximum anti-chains, maximum chains, height and width.

21. Let's investigate our question: What is the width of $([n], |)$? Start by collecting data. Generate these for various values of n and record the height h and width w.

22. We conjectured that the width is $\binom{h}{\lfloor \frac{h}{2} \rfloor}$. Does your data agree with the conjecture?