1. Log in to your Sage Cloud account.

   (a) Start Firefox or Chrome browser.
   (b) Go to http://cloud.sagemath.com
   (c) Click “Sign In”.
   (d) Click project Classroom Worksheets.
   (e) Click “New”, call it c32, then click “Sage Worksheet”.

Now we will define a new graph function for computing the hard-to-compute stability number of a graph. We could also add it to the existing Graph class as a method—and extend the Graph class by pre-computing the stability number of any graph object we initialize. This can make some computations more efficient. Stability number of a graph is a number we never want to compute more than once.

The stability number of a graph is the largest number of vertices in the graph that have no edges between them. So if \( g = \text{graphs.PetersenGraph()} \), its stability number is 4.

2. Find the stability number of \( c5 = \text{graphs.CycleGraph(5)} \) by hand. What is the largest stable set of vertices that you can find?

3. The first thing we did was to define a test to check whether the vertices \( S \) from a graph \( g \) are stable. This means there are no edges between the vertices in \( S \). So we need to search through the edges of \( g \).

   Here we tested every pair \( i \) and \( j \) from \( S \). If \( S \) is stable then the test for \((i,j)\) will be false for each possible pair.

   ```python
   def is_stable(g, S):
       E=g.edges(labels=False)
       for i in S:
           for j in S:
               if (i,j) in E:
                   return False
       return True
   
   The naive (and inefficient) way to find a largest stable set in a graph is to test every subset of vertices, check if it is stable, and then keep track of the largest one you’ve seen up to that point.

4. Here is a first function to find a maximum stable set of a graph.
def naive_maximum_stable_set(g):
    stable = []
    L = subsets(g.vertices())
    for S in L:
        if is_stable(g, S) == True:
            if len(S) > len(stable):
                stable = S
    return stable

5. Use this function to find a maximum stable set of the Petersen graph. (You should get 4).

The next big idea for finding a maximum stable set was due to Tarjan and Trojanowski in the 1970s: they noted that each vertex \( v \) of a graph is either in a maximum stable set or it is not. And, if \( v \) is in a maximum stable set then none of the vertices it is touching (that it is adjacent to is), called the neighbors of \( v \), can be in that set.

Finding the neighbors of a vertex \( v \) in a graph \( g \) is a built-in graph method: \( g . n e i g h b o r s ( v ) \)

6. Let \( g = \text{graphs.PetersenGraph()} \). Find the neighbors of vertex 0 in \( g \). Use \( g . s h o w () \) to check.

7. Find the neighbors of vertex 9 in the Petersen graph.

8. So our problem of finding a maximum stable set in a graph can be reduced to the problem of finding a maximum stable set in two smaller subgraphs: (1) the graph formed by removing vertex \( v \) and (2) the graph formed by removing \( v \) and its neighbors. In this case, we assume that \( v \) is in the maximum stable set.

9. So we need to be able to form these graphs. Let \( g \) be a graph with vertex set \( V \). Let \( S \) be any subset of \( V \). Then you can find the graph formed by \( S \) together with all the edges that are between vertices of \( S \) in the original graph \( g \) with the command \( g . s u b g r a p h ( S ) \). This is a new graph. We can give it a name, say \( h \) by \( h = g . s u b g r a p h ( S ) \).

10. Let \( S = [ 0, 1, 2 ] \). Now try \( h = g . s u b g r a p h ( S ) \) and then \( h . s h o w () \).

11. Let \( S = [ 5, 6, 7, 8, 9 ] \). Now try \( h = g . s u b g r a p h ( S ) \) and then \( h . s h o w () \).

12. Let \( S = [ 2, 3, 5, 7, 8 ] \). Now try \( h = g . s u b g r a p h ( S ) \) and then \( h . s h o w () \).

13. (Ramanujan) 2, 9, 16, etc. can be written (uniquely) as the sum of 2 cubes (\( 1^3 + 1^3, 1^3 + 2^3, 2^3 + 2^3 \), etc.). Find the smallest integer which can be written as the sum of 2 cubes in 2 different ways.