Last name ______________________

First name ______________________

LARSON—MATH 756–SAGE WORKSHEET 08
Graph Eigenvalues

The **Cvetvovic bound** of a graph G is the minimum of the number of non-negative and non-positive eigenvalues of the adjacency matrix of the graph.

Notation: Let ρ_0 be the number of 0 eigenvalues, ρ_+ be the number of positive eigenvalues, and ρ_- be the number of negative eigenvalues. Then **Cvetkovic's Theorem** says $\alpha \leq \rho_0 + \min\{\rho_+, \rho_-\}$.

1. Login to your Sage/Cocalc account.
   (a) Start the Chrome browser.
   (b) Go to `http://cocalc.com` and login.
   (c) Click on the Project for our class.
   (d) Click "New", type **s08** in the blank and click "Sage Worksheet".
2. Here are functions we can use in an investigation of this useful bound:

```
def independence_number(g):
    return g.independent_set(value_only = True)

def count_zero_eigenvalues(g):
    A = g.adjacency_matrix()
    Evals = A.eigenvalues()
    count = 0
    for lam in Evals:
        if lam == 0:
            count = count + 1
    return count
```

3. Now imitate this code and define your own functions to count the numbers of positive and negative eigenvalues of a graph adjacency matrix. Call them `count_positive_eigenvalues` and `count_negaitive_eigenvalues`.
4. Now we can code the Cvetkovic bound.

```
def cvetkovic_bound(g):
    p_0 = count_zero_eigenvalues(g)
    p_plus = count_positive_eigenvalues(g)
    p_minus = count_negative_eigenvalues(g)
    return p_0 + min(p_plus, p_minus)
```

5. Let's find the Cvetkovic bound for p_3—and see all the details to check.

   (a) Define p_3. Run `p3=graphs.PathGraph(3)`. `show` it to check the graph is what you think it is: `p3.show()`.

   (b) We'll find the adjacency matrix A. Run `A = p3.adjacency_matrix()`. Evaluate `A` to see what that is.

   (c) Them Find the eigenvalues. Run `A.eigenvalues()`.

   (d) Finally find our bound. Run: `cvetkovic_bound(p3)`.

6. Repeat the above steps for the cycle on 4 vertices, c_4. Start by defining your graph: `c4=graphs.CycleGraph(4)`.

7. Repeat the above steps for the Petersen Graph. Start by defining your graph: `pete=graphs.PetersenGraph()`.

8. No characterization of the graphs where the independence number equals the Cvetkovic bound is known. Let's find some examples. To find all connected graphs on 4 vertices where these invariants are equal, run:

```
for g in graphs.nauty_geng("4 -c"):
    if independence_number(g) == cvetkovic_bound(g):
        g.show()
        print g.graph6_string()
```

   What did you get?

9. Repeat this to find all the graphs on 5 vertices where equality holds.

10. How many graphs on 6 vertices have equality? (Do you see anything in common—can you conjecture a charaterization!?)