Walks on graphs

Create a graph with 4 vertices

```
G = Graph(4)
G
```

```
Graph on 4 vertices
```

This graph has no edges yet!

```
G.vertices()
```

```
[0, 1, 2, 3]
```

```
G.edges()
```

```
[]
```

First we need to tell Sage that our graph can have loops and multiple edges.

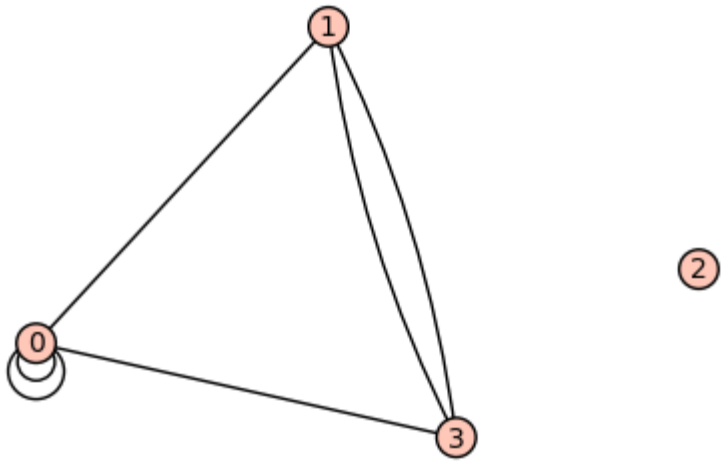
```
G.allow_loops(True)
G.allow_multiple_edges(True)
```

Now we are ready to add our edges by specifying tuple of vertices that are connected by an edge. If there are multiple edges, we need to add the tuple with multiplicity.

```
G.add_edges([(0,0),(0,0),(0,1),(0,3),(1,3),(1,3)])
```

Now let's look at the graph!

```
show(G)
```



We can construct the adjacency matrix.

```
A = G.adjacency_matrix()
A
```

```
[2 1 0 1]
[1 0 0 2]
[0 0 0 0]
[1 2 0 0]
```

In class we learned that the entry in row i and column j of the ℓ-th power of A gives us the number of paths from vertex i to vertex j

```
A**2
```

```
[6 4 0 4]
[4 5 0 1]
[0 0 0 0]
[4 1 0 5]
```

There are 4 paths of length 2 from vertex 0 to vertex 1: take either loop at 0 and then the edge 01 (2 choices) or take the edge 03 and then either of the two edges 31 (two choices):

```
(A**2)[0,1]
```

```
4
```

To count the number of closed walks, we can also look at the sum of the ℓ-th powers of the eigenvalues:

```
A.eigenvalues()
```

```
[0, -2, 0.5857864376269049?, 3.414213562373095?]
```

```
sum(la**2 for la in A.eigenvalues())
```

```
16.000000000000000?
```

We can achieve the same by looking at the trace of the ℓ-th power of the matrix:

```
(A^2).trace()
```

```
16
```

```
```