1. Log in to your Sage/Cocalc account.
   
   (a) Start the Chrome browser.
   (b) Go to http://cocalc.com and sign in.
   (c) You should see an existing Project for our class. Click on that.
   (d) Click “New”, call it c29, then click “Sage Worksheet”.

A **graph** is a mathematical object consisting of **dots** and **lines** (also called **vertices** and **edges**). A **tree** is a graph that contains no cycles.

Sage includes the **graphs** class which contains a number of **methods**. Some of these include constructors for making well-known graphs.

The **order** of a graph is the number of vertices it has. The **size** of a graph is the number of edges it has. We can create our own graph using the `Graph()` constructor, and the `add_vertex()` and `add_edge()` methods.

2. Make the following graph, called “the house”. Start by letting `house=Graph(5)`. When you are done you can view it with `house.show()`.

![Graph Diagram](image)

Another way to represent a graph with order $n$ is with an $n \times n$ **adjacency matrix** $A$. If the vertices of the graph are $\{v_0, v_1, \ldots, v_{n-1}\}$ (or $\{1,2,\ldots,n-1\}$ for short) then the $A_{i,j}$ is 1 if there is an edge from vertex $i$ to vertex $j$, and 0 if there is not.

3. Try:

   ```python
   g=graphs.PetersenGraph()
   g.show()
   g.adjacency_matrix()
   ```

   Make sure you understand the pattern of 0’s and 1’s.
4. Try:

```python
house.show()
house.adjacency_matrix()
```

Make sure you understand the pattern of 0's and 1's.

5. One way to make a graph is to start with a number of vertices and then for each pair of vertices \( n \) and \( m \), flip a coin to decide whether to put an edge between those vertices. Try this:

```python
g=Graph(10)
for i in [0..9]:
    for j in [0..9]:
        if i<j and random()<.5:
            g.add_edge(i,j)
g.size()
g.show()
```

6. The study of random graphs is huge and important and was initiated in a 1959 paper of Erdős and Renyi. Sage has a built in function to do this: `graphs.RandomGNP(n,p)`, where \( n \) is the number of vertices you want, and \( p \) is the probability of an edge \((0 \leq p \leq 1)\). To simulate a coin flip, use \( p = .5 \). Run the following code a few times.

```python
g=graphs.RandomGNP(10,.5)
g.size()
g.show()
```

You can change the probability depending on what you are trying to model. Make a Sage Interact to play with the possibilities.

```python
@interact
def i_random(order=slider(2,100,1,10),edge_probability=(.5,(0,1))):
    g=graphs.RandomGNP(order,edge_probability)
    print "The number of edges of graph g is %s" %(g.size())
g.show()
```

7. Try the problem at [http://projecteuler.net/problem=8](http://projecteuler.net/problem=8). How will you get that number into your program??? Do not type it in by hand.