Last name _____

First name _____

LARSON—OPER 635–SAGE WORKSHEET 09
Importing DFJ Data

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 **DFJ data**, then click "Sage Worksheet".

   Here are the steps for importing the distances between the 42 cities in the Dantzig-Fulkerson-Johnson (DFJ) paper. Just put a check for any steps that don't produce an output.

2. Copy the distances. Go to `http://people.sc.fsu.edu/~jburkardt/datasets/tsp/dantzig42_d.txt`.

3. Create a new file in Sage. Call it, for instance, "dantzig42.data.txt". Paste the data here.

4. In your Sage worksheet, open this file for reading.
   Evaluate: `datafile=open("dantzig42.data.txt","r")`. This creates a new object "datafile" which is just a handle for accessing your file.

5. Now actually read in the data from the datafile. Evaluate: `data = datafile.read()`. This creates a new object "data" which is just a big ASCI string with all the data.

6. You know this string somehow has a 42x42 matrix. But we have to convert the raw data to individual integers. First use the whitespace characters to break the string into chunks: evaluate `data2 = data.split()`.

7. Evaluate: `data2` and you'll see that you now have a list of integer *strings*. We need to cast those to actual integers. As an example of such a cast, evaluate: `Integer("1234567")`. What did you get?

8. Now we'll convert the data2 list of integer strings to a list of actual integers. Evaluate: `data3 = map(lambda x:  Integer(x), data2)`.

9. Evaluate: `data3` and you'll see that you now have a list of integers. There should be $42 \cdot 42 = 1764$ integers in this string. Use `len(data3)` to check that your list has the right length.

10. Now we can turn this list into a Sage matrix named `DFJ_distances`. Evaluate: `DFJ_distances = matrix(42,42,data3)`. Remember that Python starts counting at 0. So the upper left corner of your matrix is entry (0,0) while the bottom right corner is entry (41,41).

11. Now let's build a graph with all of these cities and distances. To get a graph named `DFJ` with 42 nodes, evaluate: `DFJ = Graph(42)`.

12. This graph doesn't have arcs yet, so we'll add an arc and a "label"—representing a distance—for each pair of nodes. Evaluate:

```
for i in range(42):
    for j in range(42):
        DFJ.add_edge((i,j),label=DFJ_distances[i,j])
```

13. You can see the arcs if you evaluate: `DFJ.edges()`.

14. You can get a not-very-useful picture of your graph if you evaluate: `DFJ.show()`.

15. To find the "label" (or distance) on the arc from node 0 to node 1, evaluate: `DFJ.edge_label(0,1)`. What did you get?

16. Now find the length of the tour that goes from node 0 to node 1, node 1 to node 2, ..., node 40 to node 41, and finally, node 41 back to node 0. Write the code you used to calculate this.

17. What is the length?