Last name _____

First name _____

LARSON—MATH 556–SAGE WORKSHEET 10
Computing the Degree Relaxation

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 **s10**, then click "Sage Worksheet".
2. Draw a graph with 4 nodes labeled 0,1,2,3 and with weights/costs/distances equal to the sum of the node labels.
3. Find a shortest tour for a salesman.
4. Can you prove its a shortest tour?
5. Use the following code to define the graph (named g here) to Sage.

```
g=Graph(4)
for i in range(4):
    for j in range(4):
        g.add_edge((i,j),label=i+j)
```

6. Use the following code to represent the *degree relaxation* TSP LP. Here we define a variable corresponding to each edge, and require that the sum of the decision variables corresponding to edges incident equal to any node be 2.

```
LP = MixedIntegerLinearProgram(maximization=False)
x = LP.new_variable(nonnegative=True)
LP.set_objective(sum([g.edge_label(i,j)*x[(i,j)] for i in range(4) \
    for j in range(4) if j>i]))
for v in g.vertices():
    LP.add_constraint(sum([x[u] for u in [(i,j) for i in range(4) \
        for j in range(4) if (i==v and j>i) or (j==v and i<j)]]) == 2)
for e in g.edges(labels=False):
    LP.add_constraint(x[e], max=1)
```

7. What role does the $j > i$ condition play in this code?
8. Evaluate `LP.show()` to print out the model. Explain what the variables stand for and what the constraints mean.

9. To solve this LP evaluate, `LP.solve()`. What do you get? What does this number mean?

10. To get the corresponding values for the decision variables evaluate `LP.get_values(x)`. What is the optimal tour?

11. Now we can do the same thing for the Dantzig-Fulkerson-Johnson 42 cities data. First re-initialize your DFJ graph.

12. Now we will set up the corresponding LP, called `Lpd` here. Evaluate:

```
LPd = MixedIntegerLinearProgram(maximization=False)
xd = LPd.new_variable(nonnegative=True)
LPd.set_objective(sum([DFJ.edge_label(i,j)*xd[(i,j)] for i in range(42) \
    for j in range(42) if j>i]))
for v in DFJ.vertices():
    LPd.add_constraint(sum([xd[u] for u in [(i,j) for i in range(42) \
        for j in range(42) if (i==v and j>i) or (j==v and i<j)]]) == 2)
for e in DFJ.edges(labels=False):
    LPd.add_constraint(x[e], max=1)
```

13. You can use `LPd.show()` to visualize your model. Use `LPd.solve()` to find the optimum. What do you get?

14. We would like to see the values of the decision variables corresponding to the optimum. Try `LPd.get_values(xd)`.

15. Since there are more than 1600 edges, this wasn't very useful. Most of the corresponding variables will be 0. To get just the non-trivial variables, we'll give the solution dictionary the name D and then only print the values when they are positive.

```
D = LPd.get_values(xd)
for key in D:
    if D[key] > 0:
        print key, D[key]
```

    Write these out. Better, make a map of the US, with these edges, showing your current "tour".

16. What can we add to get a better tour?