INEN471/571: Linear and Integer Programming
Computer Group Project (Option 2) (both 471 and 571)

Objective: Your objective is to implement a branch-and-cut computer code using Gurobi to solve the Traveling Salesman Problem (TSP).

Due Date: Nov. 30th! The research report and a 10-15 minute presentation should be given in class and the source files should be submitted to canvas or emailed to me at ivhicks@rice.edu.

Minimum Requirements (100 points): Lower bounds will be calculated by Gurobi; Gurobi can **only** be used to solve LPs (**not IPs**). Also, use the referenced article to derive an algorithm to generate subtour elimination constraints on your own. **Do not** use any code found online! **You have to create your own code for everything computationally except Gurobi and basic data queries.** Also, the code has to be written in Python; talk to me if your group would like to use another language. Number indexes start counting at 0 instead of 1. Please provide a readme file to give instructions on how to run your code.

I/O R**equirements:** The code must read in a file through standard input. The input file will look like the following:

```
n    m  //# of nodes and # of edges
end1 end2  weight  //for edge(0)
. . .
end1 end2 weight //for edge(m-1)
```

Standard output must be the ends of the edges chosen with their cost and the cost of the best tour found like the following:

```
end1 end2 weight //for edge(0) of tour
. . .
end1 end2 weight //for edge(n-1) of tour
The cost of the best tour is:  (the cost the best tour)
```

Test Instances: The test instance files will be sent by email to you. Below are the names and the optimal costs of the tours:

att48: 10628   berlin52: 7542   gr21: 2707   hk48: 11461   ulysses22: 7013

The Research Report: The group must also turn in a detailed report entailing an intro, a description of the problem, the steps of your algorithm, computational results (i.e. running times) of your code on the five problems and concluding remarks. In terms of computational results, the report should have a table which includes runtimes for your code on the different test instances and the number of each type of constraint that was added during each run. References would be welcomed. Also email me a critique for each of your group member and their participation in the project; rate their participation from 1 to 5 (most positive experience).

Extra Points: Extra points will be given to groups who enhance the code by adding an upper bound algorithm or some other algorithm to the code.

Technical Support: The codes should be written to run on a Unix machine (no Microsoft templates). Feel free to contact me if you have any technical questions. Good luck!

Subtour Elimination Reference:
M. Stoer and F. Wagner. A Simple Min-Cut Algorithm, J of ACM, 1997