The Traveling Salesman Problem:  
The Exhaustive Algorithm  
Python Lab

Learning Objectives:

1. to start getting comfortable using Linux;
2. to start to learn to use the Python interpreter;
3. to learn how to import and run a Python program;
4. to learn how to represent a graph using an adjacency matrix in Python;
5. to learn about the Exhaustive Algorithm for TSP by experimenting with it.

Relevant Reading:

1. Chapter 1 of How to Think Like a Computer Scientist

Preparatory Labs:

1. none

Definitions:  Be sure you have index cards for the following definitions:

1. syntax
2. semantics
3. algorithm
4. program
5. compiler
6. interpreter
7. operating system
8. text editor

1. Try to login. Use your Hofstra login, and try using your 700 number as a password. If you don’t succeed, later today you need to go to McEwen Hall to have the students at the help desk tell you your login and password. Be sure to be clear with them that you need to log in to the Linux lab in Adams 204. **Make sure you can successfully log in by tomorrow.** For today, team up with someone whose id is working.
2. Open a browser and go to cs.hofstra.edu. Find the Unix documentation. Find the how-to documents for the Linux Workstations. Use these whenever you need them.

3. Open a terminal window. One way to do so is to select the Applications menu, then the System Tools menu, and then Terminal.

4. Once you’ve got your terminal window open, type the following Linux command:

```
mkdir tsp
```

`mkdir` stands for “make directory”. Make sure that `tsp` doesn’t have any spaces in it! Then type

```
ls
```

`ls` stands for “list”. You should see your directory listed. Now enter

```
cd tsp
```

`cd` stands for “change directory”. You should now be inside your `tsp` directory. You can type

```
pwd
```

to verify this. `pwd` stands for “print working directory”. Make a note of the four Linux commands you have learned and what they do.

5. Launch the python interpreter by typing `python` at the prompt. Now you’re running Python.

6. Try using the Python interpreter as a calculator. Here are some commands to try

```
4*3
4+3
4/3
4.0/3
4**3
```

Make sure you understand the answers.

7. Go to my Web page and download `tools.py`, `data.py` and `exhaustiveTSP.py`, saving them inside your `tsp` directory.
8. In the Python interpreter, enter the following commands in response to the interpreter prompt.

```python
>>> from data import *
>>> from tools import *
>>> from exhaustiveTSP import *
>>> adjacencyMatrix = example1(6)
>>> printTable(adjacencyMatrix)
```

9. Can you draw a picture of the graph using the information in `adjacencyMatrix`?

10. What do you think the 6 is there for?

11. In the Python interpreter, enter the following commands:

```python
>>> exhaustive([0,1,2,3,4,5],adjacencyMatrix)
```

What is the output telling you? Does it seem right?

12. In class we are investigating what we call the time complexity of this algorithm, by which we mean the relationship between the time it takes an implementation to run as a function of the size of the input. Try out the exhaustive algorithm on graphs of various sizes and see what you can see.

13. Have a look at `data.py`. Can you figure out how to import your own graph? Draw a picture of a complete graph with four vertices, and label the edges with any distances you like. Can you figure out how to run the program on your graph? Try.

**Write up:**

1. Write up a lab report that is self-contained: it should indicate the problem you were trying to solve, the precise way in which you solved it, and any conclusions you are able to draw.

2. Trade write-ups with a fellow student. Review your peer’s work by following the peer review guidelines.

3. Revise your write-up.

4. Show me your original write-up with your peer’s corrections and your final write-up. We’ll add it to your portfolio once we’re both happy with it.