The Problem

The problem associated with this programming assignment is trying to find the shortest path starting at city 0 and stopping at every city until every city has been stopped at. This is used in modern applications found at places such as UPS in a much more complicated manner.

The Algorithm

This algorithm uses a stack to test the paths instead of using recursion. It starts by creating an array of visitedCities that is used to tell whether a city has already been travelled to or not by holding either a 1 or a 0, respectively. It starts with city 0 at index 0 being the only city travelled. Then we set the other variables that will be used; closestCity and minFlag.

Next we use a while loop that iterates until the pathStack is empty. This will take care of every city because the value is constantly updated within the algorithm. Then we set up a for loop to iterate through the number of cities. Next, we make sure that for the currentCity the distance is greater than 0 by checking the adjacency table and making sure it hasn't been visited yet. If it has been visited, the next iteration of the for loop takes place. If it meets all of the conditions of the if statement, we check to see if it is a smaller distance. This lets us set the smallest path from point to point. Then we add the city to the stack and reset the Boolean variable minFlag. This will run until it adds every city in our data in the shortest path possible.

Results

12 Cities

```
C:\Users\rwohl\Documents\NetBeansProjects\ProgrammingAssignment6\src>java TSP tsp12.txt
Start City: 0
Path: 1 2 5 7 3 9 4 6 8 10 11
```

29 Cities

```
C:\Users\rwohl\Documents\NetBeansProjects\ProgrammingAssignment6\src>java TSP tsp29.txt
Start City: 0
Path: 1 2 4 10 23 27 3 6 5 15 17 18 26 7 8 11 9 13 19 22 12 14 25 16 20 21 24 28
```

Observations

This was very difficult but I also believed I learned something important from this lab. I learned that using a specific data structure to just maintain your spot can be much more efficient than using other methods available and the difference in runtime it actually ends up making. I assume this will be the last abstract you read that I have turned in, so I would just like to say that this has been the hardest class I have had yet but also the most interesting. Thank You.