Cost of Gas

How Does it Work?

The cost of gas problem is a pretty simple networking problem. Any second or third year CS student, or even math student, should be able to approach this problem. You are given a cost of different edges mapped to different nodes.

Simple Example Approach

A -> B 2
B -> C 1
A -> C 4
C -> A 2

This example below shows a Floyd-Warshall approach.
The first adjacency matrix should be generated:

-	A	B	C
A	0	2	4
B	inf	0	1
C	2	inf	0

How someone may approach the initial adjacency matrix could be different, but infinity represents no direct connection, 0 being a self loop. Obviously, there does exist a path from B to A, but this first matrix is just the information we are given.

Below is code I wrote for generating a final product.

```
def floydWarshall(G):
    nodes = G.keys();
    # For each node
    # If we compare each node's relationship with all other nodes
    # With each nodes' relationship with all other nodes and all possible
intermediate nodes
    # We will take at least N^3 to find all possible values.

    # Outerloop interchanges N times,
    # middleloop interchanges N times for each combination,
```

```
    # innerloop interchanged N times for each intermediate node
combination
    for k in nodes:
        print('\ni: ', k)
        printMatrix(G)
        # For each other node
        for i in nodes:
            # For each possible intermediate node
            for j in nodes:
                G[i][j] = min(G[i][j], G[i][k] + G[k][j])
```

The Floyd-Warshall algorithm runs in O(N^3) time, but we don't particularly care about proficiency. Code to check every intermediate node is one way to approach it.

This should be the end product of our small scale example:

-	A	B	C
A	0	2	3
B	3	0	1
C	2	4	0

The flag was told to competitors to be described as a single line with no letters or spaces, like the following:

STOUTCTF{023301240}

In my original code, there was an issue with one cell having an incorrect value representing an edge. I am still unsure why it occurred, as my code is identical to the pseudo code presented in documentation. I do know running my end result back through itself resolved the issue.

Other Approaches

As this is a math problem, there exists many algorithms or hand written approaches to the actual problem. Above I have a simple example so it would be easier to understand. I attempted to make the numbers pretty random and big to discourage people from doing it handwritten, but it's still fully possible someone did do it that way.

Cost of Gas Solution

The following adjacency matrix should have been produced:

	A	B	C	D	E	F	G
A	0	109782	32324	52229	110107	140106	51779
B	26786	0	59110	79015	136893	166892	78565
C	19456	77458	0	19905	77783	107782	19455
D	64678	174460	97002	0	57878	87877	116457
E	63332	173114	95656	115561	0	29999	115111
F	33333	143115	65657	85562	143440	0	85112
G	1	109783	32325	52230	110108	140107	0

From this matrix the flag is provided.

A lot of encryption is about being able to have information go one way without it being able to go the other way. Getting the original information to the matrix is very straightforward, but being able to go backwards is near impossible without bruteforce.