Cost of Gas

ChatGPT ftw. But always read the chatgpt response and improvise from there not copy and paste. If only copy and paste you will not get the flag.

```
import numpy as np

# Add edges with their costs
edges = [
    ("NodeA", "NodeC", 32324),
    ("NodeB", "NodeA", 26786),
    ("NodeC", "NodeB", 77458),
    ("NodeC", "NodeD", 19905),
    ("NodeC", "NodeG", 19455),
    ("NodeD", "NodeA", 64678),
    ("NodeD", "NodeE", 57878),
    ("NodeE", "NodeF", 29999),
    ("NodeE", "NodeA", 82356),
    ("NodeF", "NodeC", 77777),
    ("NodeF", "NodeA", 33333),
    ("NodeF", "NodeD", 88888),
    ("NodeF", "NodeG", 88888),
    ("NodeG", "NodeA", 1)
]

# Initialize the nodes and map them to indices
nodes = ["NodeA", "NodeB", "NodeC", "NodeD", "NodeE", "NodeF", "NodeG"]
num_nodes = len(nodes)
node_index = {node: i for i, node in enumerate(nodes)}

# Initialize the adjacency matrix with infinity (high cost)
inf = float('inf')
matrix = np.full((num_nodes, num_nodes), inf)
np.fill_diagonal(matrix, 0)  # Cost to self is 0

# Populate the adjacency matrix
for src, dest, cost in edges:
    matrix[node_index[src], node_index[dest]] = cost

# Floyd-Warshall algorithm to compute shortest paths
for k in range(num_nodes):
    for i in range(num_nodes):
        for j in range(num_nodes):
            matrix[i, j] = min(matrix[i, j], matrix[i, k] + matrix[k, j])

# Convert the matrix into the required flag format
result = []
for i in range(num_nodes):
    for j in range(num_nodes):
        result.append(str(int(matrix[i, j])))

flag = "STOUTCTF{" + "".join(result) + "}"

print(flag)
```

STOUTCTF{0109782323245222911010714010651779267860591107901513689316689
27856519456774580199057778310778219455646781744609700205787887877116 45
76333217311495656115561029999115111333331431156565785562143440085112 11
09783323255223011010814010 70}