CTF PROJECTS & RESEARCH - 2025 RECRUITMENT TASK

PROJECT NAME: Intelligent Traffic Signal System for Ambulance Priority Using GPS

TASK:

Dijkstra's shortest path algorithm

1**. Performing removal operation wrongly:** u is a vertex from Q to be removed but the code uses k, which is wrong for removal.

2. Priority Queue operation: Priority queue shall need adjustment based on the neighbor node n instead of v.

3. Incorrect updation of distance: Instead of updating the distance dist[n] (representing the neighbor), the distance dist[v] is updated.

4. Incorrect variable names: The code inconsistently uses k, n, v, and neighbor—creating confusion. For example, k is used but not initialized, and it mixes k with neighbor n in the loop.

Pseudocode :

```
function Dijkstra(Graph, source):
    dist[source] <- 0
    for each vertex v in Graph:
        if v ≠ source:
            dist[v] <- ∞
    Q ← priority queue of all vertices with dist[] as the key
    while Q is not empty:
        u ← vertex in Q with min dist[u]
        remove u from Q
        for each neighbor v of u:
            alt <- dist[u] + length(u, v)
            if alt < dist[v]:
                dist[v] <- alt
                decrease priority of v in Q
    return dist
```

Code in C++ :

```cpp
#include <iostream>
#include <vector>
#include <queue>
#include <limits.h>

using namespace std;
struct Node {
    int vertex, distance;
    bool operator>(const Node& other) const {
        return distance > other.distance;
    }
};
vector<int> Dijkstra(const vector<vector<pair<int, int>>>& Graph, int source) {
    int n = Graph.size();
    vector<int> dist(n, INT_MAX);
    dist[source] = 0;
    priority_queue<Node, vector<Node>, greater<Node>> pq;
    pq.push({source, 0});
    while (!pq.empty()) {
        int u = pq.top().vertex;
        pq.pop();
        for (const auto& neighbor : Graph[u]) {
            int v = neighbor.first;
            int weight = neighbor.second;
            int alt = dist[u] + weight;
            if (alt < dist[v]) {
                dist[v] = alt;
                pq.push({v, dist[v]});
            }
        }
    }
    return dist;
}
```

2. Write a python code

1. Generates a random number between 1 and 6 simulating a dice roll.

2.when the same code is simulated in 2 system at a time, output should not be the same.

Welcome | app.py | summa.py

summa.py > ...

```python
import random

def roll_dice():
    return random.randint(1, 6)

result = roll_dice()
print("Dice roll result:", result)

```

```python
import random
import time
import os

def roll_dice():

    current_time = time.time_ns()
    system_factor = os.getpid()
    random.seed(current_time + system_factor)
    return random.randint(1, 6)
result = roll_dice()
print("Dice roll result:", result)
```