**Modern Operating System and Computer Network**

**ASSIGNMENT**

NAME : MANNE EVANCY

ROLL.NO : 2503B05116

**Question:**

Write a C++ program to implement **Dijkstra’s Single Source Shortest Path Algorithm** for a graph represented using an **adjacency matrix**.

Number of vertices: 5

Edges:

0 1 4

0 2 8

1 4 6

2 3 2

3 4 10

Source vertex: 0

**Code:**

#include <iostream>

#include <vector>

#include <queue>

#include <climits>

using namespace std;

vector<vector<vector<int> > >constructAdj(vector<vector<int> >

&edges,int V){

//adj[u] = list of {v,wt}

vector<vector<vector<int> > >adj(V);

for(const auto &edge:edges) {

int u = edge[0];

int v = edge[1];

int wt = edge[2];

adj[u].push\_back({v,wt});

adj[v].push\_back({u,wt});

}

return adj;

}

vector<int>dijkstra(int V, vector<vector<int>>& edges, int src) {

vector<vector<vector<int> > >adj = constructAdj(edges,V);

priority\_queue<vector<int>,vector<vector<int>>,

greater<vector<int> > >pq;

vector<int>dist(V,INT\_MAX);

pq.push({0,src});

dist[src]=0;

while(!pq.empty()){

int u= pq.top()[1];

pq.pop();

for(auto x:adj[u]){

int v = x[0];

int weight = x[1];

if(dist[v]>dist[u]+weight)

{

dist[v] = dist[u]+weight;

pq.push({dist[v],v});

}

}

}

return dist;

}

int main(){

int V = 5;

int src = 0;

vector<vector<int>>edges = {{0,1,4},{0,2,8},{1,4,6},{2,3,2},{3,4,10}};

vector<int>result = dijkstra(V,edges,src);

for(int dist:result)

cout<<dist<<" ";

return 0;

}

**OUTPUT :**

