After creating your graph with weights. The following function returns the shortest distance path (cost, and the path itself) from a certain source node (if exists) to a target (or to all other nodes):

```
                       //u,  {cost, v}
ll dijkstra(vector<vector<pair<ll,int>>>&adj, vector<int>&parent, int src, int tar){
   vector<ll>distance(adj.size(),LOO);
                   //{cost, node}
   priority_queue<pair<ll,int>, vector<pair<ll,int>>, greater<pair<ll,int>>>q;
   distance[src]=0;
   q.push({0, src});
   while(!q.empty()){
       int u=q.top().second;
       ll c=q.top().first;
       q.pop();
       if(u==tar) return distance[u];
       if(c!= distance[u]) continue;
       for(auto p:adj[u]){
           int v=p.second;
           ll cc=p.first;
           if(distance[v]>distance[u]+cc){
               distance[v]=distance[u]+cc;
               q.push({distance[v], v});
               parent[v]=u;
           }
       }
   }
   return -1;
}
void printParents(int n, vector<int>&par){
       while(n!=-1){
       ans.push_back(n+1);
       n=par[n];
   }
}
```

IDEONE: https://ideone.com/QOR96e