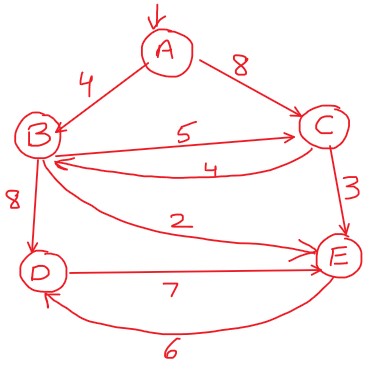
**EXPERIMENT 5**

**Aim:** Write a program to find **Single Source Shortest Path** for a directed graph using **Greedy** Technique

**Problem statement:**

Write a program to solve the Single Source shortest path problem using Dijkstras algorithm.

**Input:** Graph as a cost adjacency matrix for the below graph, with source as node A.



**Output:**

Display the final cost of travelling to all the other nodes.

Also display the path to each node from A.

**(Paste your code and output below) Code:**

#include<stdio.h>

#include<conio.h> #define INF 1000 void dijk(int s,int n,int cost[10][10],int dist[10])

{ int i,u,count,v,visited[10],min; for(i=1;i<=n;i++)

{

visited[i]=0,dist[i]=cost[s][i];

} count=2; while(count<=n)

{ min=99; for(v=1;v<=n;v++)

{ if(dist[v]<min && !visited[v])

{min=dist[v],u=v; }

} visited[u]=1; count++; for(v=1;v<=n;v++)

{ if(dist[u]+cost[u][v]<dist[v] && !visited[v])

{

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

}

}

}

}

int main()

{

int v,n,s,cost[10][10],i,dist[10],j; printf("Enter no. of nodes: "); scanf("%d",&n); printf("Enter Adjacency matrix: \n"); for(i=1;i<=n;i++)

{ for(j=1;j<=n;j++)

{ scanf("%d",&cost[i][j]); if(cost[i][j]==0)

cost[i][j]=INF;

}

}

printf("Enter Starting Node: \n"); scanf("%d",&s);

dijk(s,n,cost,dist); printf("Shortest path: \n"); for(v=1;v<=n;v++)

{ if(v!=s)

{ printf("%c->%c=%d\n",64+s,64+v,dist[v]);

}

}

getch(); return 0;

}

**Output:**

