using namespace std;

struct edge{

int a,b,w;

};

int main()

{

int n,m,s;

cout<<"\nEnter the number of nodes and edges in the graph: ";

cin>>n>>m;

cout<<"\nEnter the source Node: ";

cin>>s;

vector<edge> e(m+1);

vector<int> d(n+1,INT\_MAX);

for(int i=0;i<m;i++)

{

cout<<"\nEnter the starting edge node: ";

cin>>e[i].a;

cout<<"\nEnter the ending edge node: ";

cin>>e[i].b;

cout<<"\nEnter the weight of edge node: ";

cin>>e[i].w;

}

d[s] = 0;

for(int i=1;i<n;i++)

{

for(int j=0;j<m;j++)

{

if(d[e[j].a] < INT\_MAX)

{

d[e[j].b] = min(d[e[j].b],d[e[j].a]+e[j].w);

}

}

}

cout<<"\n";

for(int i=1;i<=n;i++)

{

cout<<d[i]<<" ";

}

/\*

OUTPUT-->

Enter the number of nodes and edges in the graph: 7 10

Enter the source Node: 1

Enter the starting edge node: 1

Enter the ending edge node: 2

Enter the weight of edge node: 6

Enter the starting edge node: 1

Enter the ending edge node: 3

Enter the weight of edge node: 5

Enter the starting edge node: 1

Enter the ending edge node: 4

Enter the weight of edge node: 5

Enter the starting edge node: 3

Enter the ending edge node: 2

Enter the weight of edge node: -2

Enter the starting edge node: 4

Enter the ending edge node: 3

Enter the weight of edge node: -2

Enter the starting edge node: 2

Enter the ending edge node: 5

Enter the weight of edge node: -1

Enter the starting edge node: 3

Enter the ending edge node: 5

Enter the weight of edge node: 1

Enter the starting edge node: 4

Enter the ending edge node: 6

Enter the weight of edge node: -1

Enter the starting edge node: 5

Enter the ending edge node: 7

Enter the weight of edge node: 3

Enter the starting edge node: 6

Enter the ending edge node: 7

Enter the weight of edge node: 3

0 1 3 5 0 4 3

\*/

}