**PRIM’S Algorithm:**

1. #include<iostream>
2. #include<conio.h>
3. #include<stdlib.h>
4. using namespace std;
5. int a[10][10],i,j,k,n,qu[10],front,rear,v,visit[10],visited[10],max1,x,y,b[10][10],d[10][10],h;
6. int bfs();
7. void maximum()
8. {
9. int i,j;
10. for(i=1;i<=n;++i)
11. {
12. for(j=1;j<=n;++j)
13. {
14. if(d[i][j]>=max1)
15. {
16. max1=d[i][j];
    1. x=i;
    2. y=j;
17. }
18. }
19. }
20. d[x][y]=0;
21. d[y][x]=0;
22. //h=max1;
23. cout<<"&"<<max1<<x<<y<<"&";
24. }
25. int main()
26. {
27. int m,c=0,t;
28. cout <<"enterno of vertices";
29. cin >> n;
30. cout <<"ente no of edges";
31. cin >> m;
32. cout <<"\nEDGES \n";
33. for(k=1;k<=m;k++)
34. {
35. cin >>i>>j;
36. cin>>t;
37. a[i][j]=t;
38. a[j][i]=t;
39. max1=t;
40. }
41. for(int s=1;s<=n;++s)
42. {
43. for(int r=1;r<=n;++r)
44. d[s][r]=a[s][r];
45. }
46. while(c!=m)
47. {
48. maximum();
49. max1=0;
50. for(int p=0;p<10;++p)
51. {
52. visited[p]=0;
53. visit[p]=0;
54. }
55. rear=0;
56. front=0;
57. for(int s=1;s<=n;++s)
58. {
59. for(int r=1;r<=n;++r)
60. b[s][r]=a[s][r];
61. }
62. b[x][y]=0;
63. b[y][x]=0;
64. t=bfs();
65. cout<<t;
66. if(t==1)
67. {
68. a[x][y]=0;
69. a[y][x]=0;
70. }
71. ++c;
72. }
73. for(int i=1;i<=n;++i)
74. {
75. for(int j=1;j<=n;++j)
76. {
77. cout<<a[i][j]<<" ";
78. }
79. cout<<"\n";
80. }
81. return 0;
82. }
83. int bfs()
84. {
85. v=x;
86. visited[v]=1;
87. k=1;
88. while(k<n)
89. {
90. for(j=1;j<=n;j++)
91. {
92. if(b[v][j]!=0 && visited[j]!=1 && visit[j]!=1)
93. {
94. visit[j]=1;
95. qu[rear++]=j;
96. }}
97. v=qu[front++];
98. k++;
99. visit[v]=0; visited[v]=1;
100. }
101. int c=0;
102. for(int o=1;o<=n;++o)
103. {
104. if(visited[o]==1)
105. ++c;
106. }
107. if(c==n)
108. return 1;
109. else
110. return 0;
111. }

KRUSKAL’S Algorithm:

1. #include<iostream>
2. #include<conio.h>
3. #include<stdlib.h>
4. using namespace std;
5. int a[10][10],i,j,k,n,qu[10],front,rear,v,visit[10],visited[10],min1,x,y,b[10][10];
6. int bfs();
7. void minimum()
8. {
9. int i,j;
10. for(i=1;i<=n;++i)
11. {
12. for(j=1;j<=n;++j)
13. {
14. if(a[i][j]<min1 && a[i][j]!=0)
15. {
    1. min1=a[i][j];
    2. x=i;
    3. y=j;
16. }
17. }
18. }
19. a[x][y]=0;
20. a[y][x]=0;
21. }
22. int main()
23. {
24. int m,c=0,t;
25. cout <<"enterno of vertices";
26. cin >> n;
27. cout <<"ente no of edges";
28. cin >> m;
29. cout <<"\nEDGES \n";
30. for(k=1;k<=m;k++)
31. {
32. cin >>i>>j;
33. cin>>t;
34. a[i][j]=t;
35. a[j][i]=t;
36. min1=t;
37. }
38. int h;
39. while(c!=m)
40. {
41. minimum();
42. h=min1;
43. min1=1000;
44. for(int p=0;p<10;++p)
45. {
46. visited[p]=0;
47. visit[p]=0;
48. }
49. rear=0;
50. front=0;
51. t=bfs();
52. if(t==1)
53. {
54. b[x][y]=h;
55. b[y][x]=h;
56. }
57. ++c;
58. }
59. for(int i=1;i<=n;++i)
60. {
61. for(int j=1;j<=n;++j)
62. {
63. cout<<b[i][j]<<" ";
64. }
65. cout<<"\n";
66. }
67. return 0;
68. }
69. int bfs()
70. {
71. v=x;
72. visited[v]=1;
73. k=1;
74. while(k<n)
75. {
76. for(j=1;j<=n;j++)
77. {
78. if(b[v][j]!=0 && visited[j]!=1 && visit[j]!=1)
79. {
80. visit[j]=1;
81. qu[rear++]=j;
82. }}
83. v=qu[front++];
84. k++;
85. visit[v]=0; visited[v]=1;
86. }
87. if(visited[y]==1)
88. return 0;
89. else
90. return 1;
91. }