# Prim's Spanning Tree Algorithm

**For**

**minimum spanning tree**

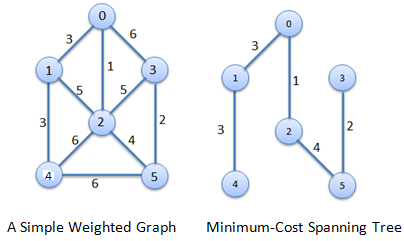
Prim's algorithm is a greedy algorithm that finds a minimum spanning tree for a connected weighted undirected graph.

It finds a subset of the edges that forms a tree that includes every vertex, where the total weight of all the edges in the tree is minimized.

This algorithm is directly based on the MST( minimum spanning tree) property.

Example

|  |  |
| --- | --- |
|  |  |
|  |  |



#### Procedure for finding Minimum Spanning Tree

**Step1**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | http://scanftree.com/Data_Structure/step1.png |

**Step2**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | http://scanftree.com/Data_Structure/step2.png |

**Step3**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | http://scanftree.com/Data_Structure/step3.png |

**Step4**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | |  |  |  |  |  |  |  | | http://scanftree.com/Data_Structure/step4.png |

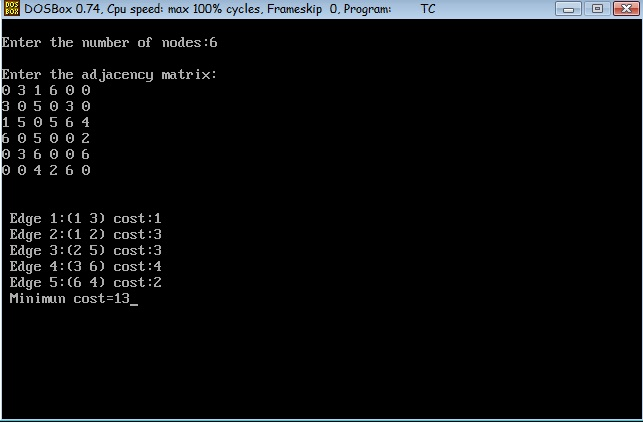
**Step5**

|  |  |
| --- | --- |
| http://scanftree.com/Data_Structure/step5.png |  |
| **Minimum Cost** = 1+2+3+3+4 = 13 |  |

### C IMPLEMETATION of prim's Algorithm

1. #include<stdio.h>
3. #include<conio.h>
5. inta,b,u,v,n,i,j,ne=1;
7. int visited[10]={0},min,mincost=0,cost[10][10];
9. void main()
11. {
13. clrscr();
15. printf("\nEnter the number of nodes:");
17. scanf("%d",&n);
19. printf("\nEnter the adjacency matrix:\n");
21. for(i=1;i<=n;i++)
23. for(j=1;j<=n;j++)
25. {
27. scanf("%d",&cost[i][j]);
29. if(cost[i][j]==0)
31. cost[i][j]=999;
33. }
35. visited[1]=1;
37. printf("\n");
39. while(ne < n)
41. {
43. for(i=1,min=999;i<=n;i++)
45. for(j=1;j<=n;j++)
47. if(cost[i][j]< min)
49. if(visited[i]!=0)
51. {
53. min=cost[i][j];
55. a=u=i;
57. b=v=j;
59. }
61. if(visited[u]==0|| visited[v]==0)
63. {
65. printf("\n Edge %d:(%d %d) cost:%d",ne++,a,b,min);
67. mincost+=min;
69. visited[b]=1;
71. }
73. cost[a][b]=cost[b][a]=999;
75. }
77. printf("\n Minimun cost=%d",mincost);
79. getch();
81. }

### output



**Example**

