**C Program for distance vector algorithm**

Solution:

1. #include<stdlib.h>
2. #define nul 1000
3. #define nodes 10
4. int no;
5. struct node
6. {
7. int a[nodes][4];
8. }router[nodes];
9. void init(int r)
10. {
11. int i;
12. for(i=1;i<=no;i++)
13. {
14. router[r].a[i][1]=i;
15. router[r].a[i][2]=999;
16. router[r].a[i][3]=nul;
17. }
18. router[r].a[r][2]=0;
19. router[r].a[r][3]=r;
20. }
21. void inp(int r)
22. {
23. int i;
24. printf("**\n**Enter dist from the node %d to other nodes",r);
25. printf("**\n**Pls enter 999 if there is no direct route**\n**",r);
26. for(i=1;i<=no;i++)
27. {
28. if(i!=r)
29. {
30. printf("**\n**Enter dist to the node %d:",i);
31. scanf("%d",&router[r].a[i][2]);
32. router[r].a[i][3]=i;
33. }
34. }
35. }
36. void display(int r)
37. {
38. int i,j;
39. printf("**\n\n**The routing table for node %d is as follows:",r);
40. for(i=1;i<=no;i++)
41. {
42. if(router[r].a[i][2]>=999)
43. printf("**\n\t\t\t** %d **\t** no link **\t** no hop",router[r].a[i][1]);
44. else
45. printf("**\n\t\t\t** %d **\t** %d **\t\t** d",router[r].a[i][1],router[r].a[i][2],router[r].a[i][3]);
46. }
47. }
48. void dv\_algo(int r)
49. {
50. int i,j,z;
51. for(i=1;i<=no;i++)
52. {
53. if(router[r].a[i][2]!=999 && router[r].a[i][2]!=0)
54. {
55. for(j=1;j<=no;j++)
56. {
57. z=router[r].a[i][2]+router[i].a[j][2];
58. if(router[r].a[j][2]>z)
59. {
60. router[r].a[j][2]=z;
61. router[r].a[j][3]=i;
62. }
63. }
64. }
65. }
66. }
67. int main()
68. {
69. int i,j,x,y;
70. char choice;
71. printf("Enter the no. of nodes required (less than 10 pls):");
72. scanf("%d",&no);
73. for(i=1;i<=no;i++)
74. {
75. init(i);
76. inp(i);
77. }
78. printf("**\n**The configuration of the nodes after initialization is as follows:");
79. for(i=1;i<=no;i++)
80. display(i);
81. for(i=1;i<=no;i++)
82. dv\_algo(i);
83. printf("**\n**The configuration of the nodes after computation of paths is as follows:");
84. for(i=1;i<=no;i++)
85. display(i);
86. while(1)
87. {
88. printf("**\n\n**Wanna continue (y/n):");
89. scanf("%c",&choice);
90. if(choice=='n')
91. **break**;
92. printf("**\n**Enter the nodes btn which shortest path is to be found:**\n**");
93. scanf("%d %d",&x,&y);
94. printf("**\n**The length of the shortest path is %d",router[x].a[y][2]);
95. }
96. }

**Output C Program for distance vector algorithm to find suitable path for transmission**

[root@localhost ~]# ./a.out

Enter the no. of nodes required (less than 10 pls):4

Enter dist from the node 1 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 2:5

Enter dist to the node 3:3

Enter dist to the node 4:7

Enter dist from the node 2 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:5

Enter dist to the node 3:999

Enter dist to the node 4:6

Enter dist from the node 3 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:3\

Enter dist to the node 2:

Enter dist to the node 4:

Enter dist from the node 4 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:

Enter dist to the node 2:

Enter dist to the node 3:

The configuration of the nodes after initialization is as follows:

The routing table for node 1 is as follows:

1 0 1

2 5 2

3 3 3

4 7 4

The routing table for node 2 is as follows:

1 5 1

2 0 2

3 no link no hop

4 6 4

The routing table for node 3 is as follows:

1 3 1

2 no link no hop

3 0 3

4 no link no hop

The routing table for node 4 is as follows:

1 no link no hop

2 no link no hop

3 no link no hop

4 0 4

The configuration of the nodes after computation of paths is as follows:

The routing table for node 1 is as follows:

1 0 1

2 5 2

3 3 3

4 7 4

The routing table for node 2 is as follows:

1 5 1

2 0 2

3 8 1

4 6 4

The routing table for node 3 is as follows:

1 3 1

2 8 1

3 0 3

4 10 1

The routing table for node 4 is as follows:

1 no link no hop

2 no link no hop

3 no link no hop

4 0 4

Wanna continue (y/n):

Enter the nodes btn which shortest path is to be found:

^C

[root@localhost ~]# clear

[root@localhost ~]# ./a.out

Enter the no. of nodes required (less than 10 pls):4

Enter dist from the node 1 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 2:5

Enter dist to the node 3:3

Enter dist to the node 4:7

Enter dist from the node 2 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:5

Enter dist to the node 3:999

Enter dist to the node 4:6

Enter dist from the node 3 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:3

Enter dist to the node 2:999

Enter dist to the node 4:2

Enter dist from the node 4 to other nodes

Pls enter 999 if there is no direct route

Enter dist to the node 1:7

Enter dist to the node 2:6

Enter dist to the node 3:2

The configuration of the nodes after initialization is as follows:

The routing table for node 1 is as follows:

1 0 1

2 5 2

3 3 3

4 7 4

The routing table for node 2 is as follows:

1 5 1

2 0 2

3 no link no hop

4 6 4

The routing table for node 3 is as follows:

1 3 1

2 no link no hop

3 0 3

4 2 4

The routing table for node 4 is as follows:

1 7 1

2 6 2

3 2 3

4 0 4

The configuration of the nodes after computation of paths is as follows:

The routing table for node 1 is as follows:

1 0 1

2 5 2

3 3 3

4 5 3

The routing table for node 2 is as follows:

1 5 1

2 0 2

3 8 1

4 6 4

The routing table for node 3 is as follows:

1 3 1

2 8 1

3 0 3

4 2 4

The routing table for node 4 is as follows:

1 5 3

2 6 2

3 2 3

4 0 4

Wanna continue (y/n):

Enter the nodes btn which shortest path is to be found:

1 4

The length of the shortest path is 7

Wanna continue (y/n):

Enter the nodes btn which shortest path is to be found-