Data Structures Odyssey: Exploring the Foundations of Computing

Ex. No.: 11	Topological Sorting	Date:09/05/2024

Write a C program to create a graph and display the ordering of vertices.

Algorithm:

1) Start
2) Initialize an empty stack to store the topologically sorted elements.
3) Initialize a set to track visited nodes.
4) Start a depth-first search (DFS) from any unvisited node in the graph.
5) During DFS traversal: a. Mark the current node as visited.
b. Recursively visit all adjacent nodes that are not visited yet.
6) Once a node has no unvisited adjacent nodes, push it onto the stack.
7) Repeat steps 3-5 until all nodes are visited.
Pop elements from the stack to get the topologically sorted order. 8)
Stop

Data Structures Odyssey: Exploring the Foundations of Computing

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>

int s[100], j, res[100; void
AdjacencyMatrix(int a[][100], int n) {
 int i, j; for (i = 0; i < n;
i++) { for (j = 0; j <=
n; j++) { a[i][j] = 0;
} } for (i = 1; i < n;
i++) { for (j = 0; j < i;
j++) { a[i][j] = rand()
% 2; a[j][i] = 0;
}
}
}

void dfs(int u, int n, int a[][100]) {
int v; s[u] = 1; for (v = 0; v < n - 1;
v++) { if (a[u][v] == 1 && s[v] ==
0) { dfs(v, n, a);

}
} j +=
1;
res[j]
= u;

}

void topological_order(int n, int a[][100]) {
```

```
int i, u; for (i = 0; i < n;
i++) { s[i] = 0; } j = 0;
for (u = 0; u < n; u++) {
if (s[u] == 0) { dfs(u, n,
a);
}
}
return;
}
int main() { int
a[100][100], n, i, j;

printf("Enter number of vertices\n");
scanf("%d", &n);

AdjacencyMatrix(a, n);  printf("\t\tAdjacency Matrix of the graph\n"); /* PRINT
ADJACENCY MATRIX */

for (i = 0; i < n; i++) {
for (j = 0; j < n; j++) {
printf("\t%d", a[i][j]);
}
printf("\n");
}


printf("\nTopological order:\n");

topological_order(n, a);

for (i = n; i >= 1; i--) { printf("-->%d",
res[i]);
}
```

return 0;

}

OUTPUT:

```
aiml2315011167@cselab:~$ gcc program12.c
aiml2315011167@cselab:~$ ./a.out
Enter number of vertices
2
                Adjacency Matrix of the graph
        0       0
        1       0

Topological order:
-->1-->0aiml2315011167@cselab:~$
```

RESULT: Thus, the program was successfully executed.