DATA STRUCTURES

DAY-13

1.Topological Sort

Program:

```
#include <stdio.h>

#include <stdlib.h>

 #define MAX 100

 typedef struct Node {

    int vertex;

    struct Node* next;

} Node;

 Node* createNode(int v);

void addEdge(Node* adj[], int u, int v);

void topologicalSortUtil(int v, int visited[], Node* adj[], int* stack, int* stackIndex);

void topologicalSort(Node* adj[], int V);

int main() {

    int V, E, u, v;

 printf("Enter the number of vertices: ");

    scanf("%d", &V);

     Node* adj [MAX] = {NULL};

    printf("Enter the number of edges: ");

    scanf("%d", &E);

    for (int i = 0; i < E; i++) {

        printf("Enter edge (u v): ");

        scanf("%d %d", &u, &v);
        addEdge(adj, u, v);
    }

    printf("Topological Sort: ");
    topologicalSort(adj, V);
    return 0;
}
Node* createNode(int v) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->vertex = v;
    newNode->next = NULL;
    return newNode;
}
void addEdge(Node* adj[], int u, int v) {
    Node* newNode = createNode(v);
    newNode->next = adj[u];
    adj[u] = newNode;
}
void topologicalSortUtil(int v, int visited[], Node* adj[], int* stack, int* stackIndex) {
    visited[v] = 1;
    Node* temp = adj[v];
    while (temp) {
        int adjVertex = temp->vertex;
        if (!visited[adjVertex])
            topologicalSortUtil(adjVertex, visited, adj, stack, stackIndex);
        temp = temp->next;
    }
    stack[(*stackIndex)++] = v;
}

void topologicalSort(Node* adj[], int V) {
    int visited[MAX] = {0};
    int stack[MAX];
    int stackIndex = 0;
    for (int i = 0; i < V; i++) {
        if (!visited[i])
            topologicalSortUtil(i, visited, adj, &stack[0], &stackIndex);
    }
     for (int i = stackIndex - 1; i >= 0; i--) {
        printf("%d ", stack[i]);
    }
    printf("\n");
}
```

Output:

```
Enter the number of vertices: 6
Enter the number of edges: 6
Enter edge (u v): 5 2
Enter edge (u v): 5 0
Enter edge (u v): 4 0
Enter edge (u v): 4 1
Enter edge (u v): 2 3
Enter edge (u v): 3 1
```

```
Topological Sort: 5 4 2 3 1 0
```

2.Terminology sort

Program:

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


#define MAX_TERMS 100
#define MAX_LEN 100
void sortTerms(char terms[][MAX_LEN], int n);
int main() {
  int n;
  char terms[MAX_TERMS][MAX_LEN];
  printf("Enter the number of terms: ");
  scanf("%d", &n);
  getchar();
  printf("Enter the terms:\n");
  for (int i = 0; i < n; i++) {
    fgets(terms[i], MAX_LEN, stdin);
    terms[i][strcspn(terms[i], "\n")] = 0;
  }
  sortTerms(terms, n);
  printf("\nSorted terms:\n");
  for (int i = 0; i < n; i++) {
    printf("%s\n", terms[i]);
  }
  return 0;
}
void sortTerms(char terms[][MAX_LEN], int n) {
  char temp[MAX_LEN];
  for (int i = 0; i < n - 1; i++) {
    for (int j = i + 1; j < n; j++) {
      if (strcmp(terms[i], terms[j]) > 0) {
        strcpy(temp, terms[i]);
        strcpy(terms[i], terms[j]);
        strcpy(terms[j], temp);
      }
    }
  }
}
```

Output:

```
Enter the number of terms: 5
Enter the terms:
banana
apple
grape
cherry
date

Sorted terms:
apple
banana
cherry
```

date

grape