1.Quick sort

```
#include <stdio.h>


Void swap(int* a, int* b) {

   Int temp = *a;

   *a = *b;

   *b = temp;

}


Int partition(int arr[], int low, int high) {

   Int pivot = arr[high];

   Int I = low – 1;


   For (int j = low; j < high; j++) {

      If (arr[j] < pivot) {

         I++;

         Swap(&arr[i], &arr[j]);

      }

   }

   Swap(&arr[I + 1], &arr[high]);

   Return I + 1;

}


Void quickSort(int arr[], int low, int high) {

   If (low < high) {

      Int pi = partition(arr, low, high);
```

```
        quickSort(arr, low, pi – 1);

        quickSort(arr, pi + 1, high);

    }

}


Void printArray(int arr[], int size) {

    For (int I = 0; I < size; i++)

        Printf(“%d “, arr[i]);

    Printf(“\n”);

}


Int main() {

    Int n;

    Printf(“Enter the number of elements: “);

    Scanf(“%d”, &n);


    Int arr[n];

    Printf(“Enter the elements: “);

    For (int I = 0; I < n; i++)

        Scanf(“%d”, &arr[i]);


    quickSort(arr, 0, n – 1);


    printf(“Sorted array: \n”);

    printArray(arr, n);

    return 0;
```

```
}
```

Input

```
Enter the number of elements: 6

Enter the elements: 10 7 8 9 1 5
```

Output

```
Sorted array:

1 5 7 8 9 10
```

2.Topological sort

```
#include <stdio.h>

#include <stdlib.h>


#define MAX 100


Int n; // Number of vertices in the graph

Int adj[MAX][MAX]; // Adjacency matrix

Int visited[MAX]; // Array to mark visited nodes

Int stack[MAX]; // Stack to store the topological sort

Int top = -1;


Void dfs(int v) {

   Visited[v] = 1;

   For (int I = 0; I < n; i++) {

      If (adj[v][i] == 1 && visited[i] == 0) {

         Dfs(i);

      }

   }
```

```
    Stack[++top] = v;
}

Void topologicalSort() {
    For (int I = 0; I < n; i++) {
        Visited[i] = 0;
    }

    For (int I = 0; I < n; i++) {
        If (visited[i] == 0) {
            Dfs(i);
        }
    }

    Printf("Topological Sort: ");
    While (top != -1) {
        Printf("%d ", stack[top--]);
    }
    Printf("\n");
}

Int main() {
    Printf("Enter the number of vertices: ");
    Scanf("%d", &n);

    Printf("Enter the adjacency matrix:\n");
    For (int I = 0; I < n; i++) {
        For (int j = 0; j < n; j++) {
            Scanf("%d", &adj[i][j]);
        }
    }

    topologicalSort();
    return 0;
}
```

INPUT

```
Enter the number of vertices: 6
Enter the adjacency matrix:
0 1 0 0 0 0
0 0 1 1 0 0
0 0 0 0 0 0
0 0 0 0 1 1
0 0 0 0 0 1
0 0 0 0 0 0
```

Output

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