Selection Sort:

Definition:

Selection Sort is a simple comparison-based sorting algorithm. It divides the input list into two parts: a sorted part and an unsorted part. It repeatedly selects the smallest (or largest) element from the unsorted part, swaps it with the leftmost unsorted element, and moves the boundary of the sorted part one element to the right.

Time Complexity: $O(n^2)$, where `n` is the number of elements.

Example Program in C:

```
#include <stdio.h>
void selectionSort(int arr[], int n) {
    int i, j, min_idx;
    for (i = 0; i < n-1; i++) {
        min_idx = i;
        for (j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;
        int temp = arr[min_idx];
        arr[min_idx] = arr[i];
        arr[i] = temp;
    }
}
void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
int main() {
    int arr[] = {64, 25, 12, 22, 11};
```

```
    int n = sizeof(arr)/sizeof(arr[0]);
    printf("Original array: ");
    printArray(arr, n);
    selectionSort(arr, n);
    printf("Sorted array: ");
    printArray(arr, n);
    return 0;
}
```

Merge Sort:

Definition:

Merge Sort is an efficient, stable, divide-and-conquer sorting algorithm. It works by recursively dividing the unsorted array into two halves until each subarray contains a single element. Then, it merges the sorted subarrays to produce a sorted array.

Time Complexity : O(n log n), where `n` is the number of elements.

```
#include <stdio.h>
void merge(int arr[], int left, int mid, int right) {
    int n1 = mid - left + 1;
    int n2 = right - mid;
    int leftArr[n1], rightArr[n2];
    for (int i = 0; i < n1; i++)
        leftArr[i] = arr[left + i];
    for (int i = 0; i < n2; i++)
        rightArr[i] = arr[mid + 1 + i];
    int i = 0, j = 0, k = left;
    while (i < n1 && j < n2) {
        if (leftArr[i] <= rightArr[j]) {
            arr[k] = leftArr[i];
            i++;
        } else {
            arr[k] = rightArr[j];
```

```
            j++;
        }
        k++;
    }
    while (i < n1) {
        arr[k] = leftArr[i];
        i++;
        k++;
    }
    while (j < n2) {
        arr[k] = rightArr[j];
        j++;
        k++;
    }
}
void mergeSort(int arr[], int left, int right) {
    if (left < right) {
        int mid = left + (right - left) / 2;
        mergeSort(arr, left, mid);
        mergeSort(arr, mid + 1, right);
        merge(arr, left, mid, right);
    }
}
void printArray(int arr[], int size) {
    for (int i = 0; i < size; i++)
        printf("%d ", arr[i]);
    printf("\n");
}
int main() {
```

```
    int arr[] = {12, 11, 13, 5, 6, 7};
    int arrSize = sizeof(arr) / sizeof(arr[0]);
    printf("Original array: ");
    printArray(arr, arrSize);
    mergeSort(arr, 0, arrSize - 1);
    printf("Sorted array: ");
    printArray(arr, arrSize);

    return 0;
}
```

SEAT BOOKING USING 2D ARRAY

```
#include<stdio.h>
int arr[3][4]={0,0,0,0,0,0,0,0,0,0,0,0},column,row,choice;
char second_choice;
void print_seats()
{
    for(int i=0;i<3;i++)
    {
        for(int j=0;j<4;j++)
        {
            printf("%d ",arr[i][j]);
        }
        printf("\n");
    }
}
void book_seats()
{
    printf("Enter the row from (0-3)\n");
    scanf("%d",&row);
```

```
    printf("Enter the column from (0-4)\n");
    scanf("%d",&column);
    if((column<4) && (row < 3))
    {
        arr[row][column] = 1;
    }
    else{
        printf("Invalid row or column number:\n");
    }
    printf("Your seat is successfully Booked\n");
     for(int i=0;i<3;i++)
    {
        for(int j=0;j<4;j++)
        {
            printf("%d ",arr[i][j]);
        }
        printf("\n");
    }
}
void main()
{
    do{
        printf("Press 1 to book the seat\n");
        printf("Press 2 to dispaly all the seats\n");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
                book_seats();
```

```
            break;
        case 2:
            print_seats();
            break;
        default:
            printf("Invalid choice");
      }
      printf("Do you want to continue booking(y/n)\n");
      scanf("%s",&second_choice);
   }while(second_choice == 'y');
}
```