SORTING TECHNIQUES

Bubble Sort:

Bubble sort is a simple and intuitive sorting algorithm. It repeatedly swaps adjacent elements if they are in the wrong order until the array is sorted. In this algorithm, the largest element "bubbles up" to the end of the array in each iteration. Bubble sort algorithm

1. Start at index zero, compare the element with the next one (a[0] & a[1] (a is the name of the array)), and swap if a[0] > a[1]. Now compare a[1] & a[2] and swap if a[1] > a[2]. Repeat this process until the end of the array. After doing this, the largest element is present at the end. This whole thing is known as a pass. In the first pass, we process array elements from [0,n-1].
2. Repeat step one but process array elements [0, n-2] because the last one, i.e., a[n-1], is present at its correct position. After this step, the largest two elements are present at the end.
3. Repeat this process n-1 times.

Program:-

```
#include <stdio.h>
int main()
{
	int array[100], n, c, d, swap;
	printf("Enter number of elements\n");
	scanf("%d", &n);
	printf("Enter %d integers\n", n);
	for (c = 0; c < n; c++)
		scanf("%d", &array[c]);
	for (c = 0 ; c < n - 1; c++)
	{
		for (d = 0 ; d < n - c - 1; d++)
		{
			if (array[d] > array[d+1]) /* For decreasing order use '<' instead of '>' */
			{
				swap = array[d];
				array[d] = array[d+1];
				array[d+1] = swap;
			}
		}
	}
	printf("Sorted list in ascending order:\n");
	for (c = 0; c < n; c++)
		printf("%d\n", array[c]);
	return 0;
}
```

Output:-

```
Enter number of elements
5
Enter 5 integers
3
22
54
667
Sorted list in ascending order:
3
9
22
54
667
```

Insertion Sort

Insertion Sort is basically the insertion of an element from a random set of numbers, to its correct position where it should actually be, by shifting the other elements if required.

Insertion Sort Algorithm:

1. We will store the random set of numbers in an array.
2. We will traverse this array and insert each element of this array, to its correct position where it should actually be, by shifting the other elements on the left if required.
3. The first element in the array is considered as sorted, even if it is an unsorted array. The array is sub-divided into two parts, the first part holds the first element of the array which is considered to be sorted and second part contains all the remaining elements of array.
4. With each iteration one element from the second part is picked and inserted into the first part of array at its correct position by shifting the existing elements if required.
5. This goes until the last element in second part of array is placed in correct position in the output array.
6. Now, we have the array in sorted order.

Program:-

```
#include <stdio.h>
int main(void)
{
	int n, i, j, temp;
	int arr[64];
	printf("Enter number of elements\n");
	scanf("%d", &n);
	printf("Enter %d integers\n", n);
	for (i = 0; i < n; i++)
	{
		scanf("%d", &arr[i]);
```

```
    }
    for (i = 1; i < n; i++)
    {
        j = i;
        while (j > 0 && arr[j - 1] > arr[j])
        {
            temp = arr[j];
            arr[j] = arr[j - 1];
            arr[j - 1] = temp;
            j--;
        }
    }
    printf("Sorted list in ascending order:\n");
    for (i = 0; i < n; i++)
    {
        printf("%d\n", arr[i]);
    }
    return 0;
}
```

Output:-

```
Enter number of elements
5
Enter 5 integers
23
45
11
2
66
Sorted list in ascending order:
2
11
23
45
66
```

Selection sort

Selection sort in C to sort numbers of an array in ascending order. With a little modification, it arranges numbers in descending order.

Selection sort algorithm (for ascending order)

1.Find the minimum element in the array and swap it with the element in the 1st position.
2.Find the minimum element again in the remaining array[2, n] and swap it with the element at 2nd position, now we have two elements at their correct positions.
3.We have to do this n-1 times to sort the array.

Program:-

```
#include <stdio.h>
int main()
{
	int array[100], n, c, d, position, t;
	printf("Enter number of elements\n");
	scanf("%d", &n);
	printf("Enter %d integers\n", n);
	for (c = 0; c < n; c++)
		scanf("%d", &array[c]);
	for (c = 0; c < (n - 1); c++) // finding minimum element (n-1) times
	{
		position = c;
		for (d = c + 1; d < n; d++)
		{
			if (array[position] > array[d])
				position = d;
		}
		if (position != c)
		{
			t = array[c];
			array[c] = array[position];
			array[position] = t;
		}
	}
	printf("Sorted list in ascending order:\n");
	for (c = 0; c < n; c++)
		printf("%d ", array[c]);
	return 0;
}
```

Output:-

```
Enter number of elements
6
Enter 6 integers:23 11 5 55 76 8
Sorted list in ascending order:
5
8
11
23
55
76
```