**1) Bubble Sort**

#include <stdio.h>

void main()

{

int array[100], n, c, d, swap;

clrscr();

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 ; c++)

{

for (d = 0 ; d < n - c - 1; d++)

{

if (array[d] > array[d+1])

{

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]);

getch();

}

**2) Insertion Sort**

#include<stdio.h>

void main()

{

int i, j, count, temp, number[25];

clrscr();

printf("How many numbers u are going to enter?: ");

scanf("%d",&count);

printf("Enter %d elements: ", count);

for(i=0;i<count;i++)

scanf("%d",&number[i]);

for(i=1;i<count;i++){

temp=number[i];

j=i-1;

while((temp<number[j])&&(j>=0)){

number[j+1]=number[j];

j=j-1;

}

number[j+1]=temp;

}

printf("Order of Sorted elements: ");

for(i=0;i<count;i++)

printf(" %d",number[i]);

getch();

}

**3) Selection Sort**

#include <stdio.h>

void selection\_sort();

int a[30], n;

void main()

{

int i;

clrscr();

printf("\nEnter size of an array: ");

scanf("%d", &n);

printf("\nEnter elements of an array:\n");

for(i=0; i<n; i++)

scanf("%d", &a[i]);

selection\_sort();

printf("\n\nAfter sorting:\n");

for(i=0; i<n; i++)

printf("\n%d", a[i]);

getch();

}

void selection\_sort()

{

int i, j, min, temp;

for (i=0; i<n; i++)

{

min = i;

for (j=i+1; j<n; j++)

{

if (a[j] < a[min])

min = j;

}

temp = a[i];

a[i] = a[min];

a[min] = temp;

}

}

**4) Merge Sort**

#include <stdio.h>

#include<conio.h>

void mergeSort(int [], int, int, int);

void partition(int [],int, int);

void main()

{

int list[50];

int i, size;

clrscr();

printf("Enter total number of elements:");

scanf("%d", &size);

printf("Enter the elements:\n");

for(i = 0; i < size; i++)

{

scanf("%d", &list[i]);

}

partition(list, 0, size - 1);

printf("After merge sort:\n");

for(i = 0;i < size; i++)

{

printf("%d ",list[i]);

}

getch();

}

void partition(int list[],int low,int high)

{

int mid;

if(low < high)

{

mid = (low + high) / 2;

partition(list, low, mid);

partition(list, mid + 1, high);

mergeSort(list, low, mid, high);

}

}

void mergeSort(int list[],int low,int mid,int high)

{

int i, mi, k, lo, temp[50];

lo = low;

i = low;

mi = mid + 1;

while ((lo <= mid) && (mi <= high))

{

if (list[lo] <= list[mi])

{

temp[i] = list[lo];

lo++;

}

else

{

temp[i] = list[mi];

mi++;

}

i++;

}

if (lo > mid)

{

for (k = mi; k <= high; k++)

{

temp[i] = list[k];

i++;

}

}

else

{

for (k = lo; k <= mid; k++)

{

temp[i] = list[k];

i++;

}

}

for (k = low; k <= high; k++)

{

list[k] = temp[k];

}

}