OUTPUT!

```
Enter no of elements for the array : 5
Enter elements of the array
 80
 50
 23
 -5
 1
List after sorting
-5   1   23   50   80
```


Experiment No. : 1 Date :

Aim Write a C program to implement bubble sort

description traverse from left and compare adjacent elements and the higher one is place at right it has average and worst case time complexity as $O(n^2)$

Source code

```
# include <stdio.h>
# include <conio.h>
void display_arr (int arr[], int n);
void bubble_sort (int arr[], int n);
void main () {
 int temp, n, i;
 int arr [50];
 clrscr ();
printf (" Enter number of element for
        the array: ");
scanf ("%d", &n);
printf (" Enter the elements of the array");
for (i = 0; i < n; i++) {
    scanf ("%d", & arr[i] ); }
```

Experiment No. : Date :

```
bubble_sort(arr, n);
printf("List after sorting \n");
display_arr(arr, n);
getch(); }
void display_arr(int arr[], int n) {
int i;
for(i=0; i<n; i++) {
printf("%d", arr[i]); }}
void bubble_sort(int arr[], int n) {
int i, j, temp;
for(i=0; i<n; i++) {
for(j=0; j<n-i-1; j++) {
if(arr[j] > arr[j+1]) {
temp = arr[j];
arr[j] = arr[j+1];
arr[j+1] = temp; }
}}}
```

Result Bubble sort is implemented and given list of array is sorted

OUTPUT2

```
Enter element count for array : 5
Enter elements of the array
82
50
-4
2
10
List after sorting
-4   2   10   50   82
```


Experiment No. : Date :

Aim Write a C program to implement insertion sort

description The algorithm takes one value at a time from unsorted part of the array and puts it into right place of sorted part of array and has average and worst case time complexity $O(n^2)$

Source code

```
# include <stdio.h>
# include <conio.h>
void display_arr (int arr[], int n);
void insertion_sort (int arr[], int n);
void main () {
int temp, n, i ;
int arr[50];
clrscr ();
printf (" Enter element count for array ");
scanf ("%d" &n);
printf (" Enter elements of the array \n");
for (i = 0; i < n ; i ++) {
  scanf ("%d", & arr[i]); }
  insertion_sort (arr, n);
```

```
printf ("List after sorting \n");
display_arr (arr, n);
getch (); }
void display_arr (int arr[], int n){
int i;
for (i=0; i<n; i++) {
printf ("%d", arr[i]); }}
void insertion_sort (int arr[], int n){
int temp, j, i;
for (i=1; i<n; i++) {
temp = arr[i];
j = j-1;
while (j>=0 && arr[j]>temp)
{ arr[j+1] = arr[j];
arr[j] = temp;
j = j-1; }}
}
```

Result Insertion sort is implemented and given list of array is sorted.