**ASSIGNMENT - 4**

**1. PROBLEM STATEMENT**

Write a program in C to implement insertion sort

**2. ALGORITHM**

**Algorithm Insertion\_Sort**

**Input:** A pointer to an integer array named arr[1…n] with size n

**Output:** The array arr in sorted order

**Remarks:** The array is sorted in ascending order

**Steps:**

1. **For**(j=2 to n) **do** // traversing the unsorted list
2. temp = arr[i] // storing the leftmost element of unsorted list
3. **For** j = i-1 to 1 **AND** arr[j]>temp **do** // finding valid position
4. arr[j+1] = arr[j] // shifting elements to the right
5. **EndFor**
6. arr[j+1] = temp // placing temp at valid position
7. **Stop**

**3. SOURCE CODE**

#include<stdio.h>

#include<stdlib.h>

// function to display an array

void disp(int\* arr,int num)

{

int i;

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

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

}

// function to take input in an array

void getarr(int\*arr,int num)

{

int i;

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

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

}

void insertionsort(int\* arr,int num)

{

int i,j,temp;

for(i=1;i<num;i++)

{

temp=arr[i]; // storing first element of unsorted list

for(j=i-1; j>=0 && arr[j]>temp;j--) // finding valid position

arr[j+1]=arr[j]; // shifting elements to the right

arr[j+1]=temp; // inserting temp at valid position

printf("\n\nPASS %d: ",i);

disp(arr,num);

}

}

int main(void)

{

int \*arr,num,i,j;

//taking the length of the array from the user

printf("Enter the number of elements needed: ");

scanf("%d",&num);

arr = (int\*)calloc(num,sizeof(int)); // creating array in heap

// checking if at least 2 elements are present

if(num<2)

{

printf("Invalid Array Length\nPlease Retry");

return 0;

}

//taking the elements of the array from user

printf("Enter %d elements of the Array: ",num);

getarr(arr,num);

//displaying the user entered array

printf("Entered Array: ");

disp(arr,num);

//calling insertion sort function

insertionsort(arr,num);

//displaying the sorted array

printf("\n\nSorted Array: ");

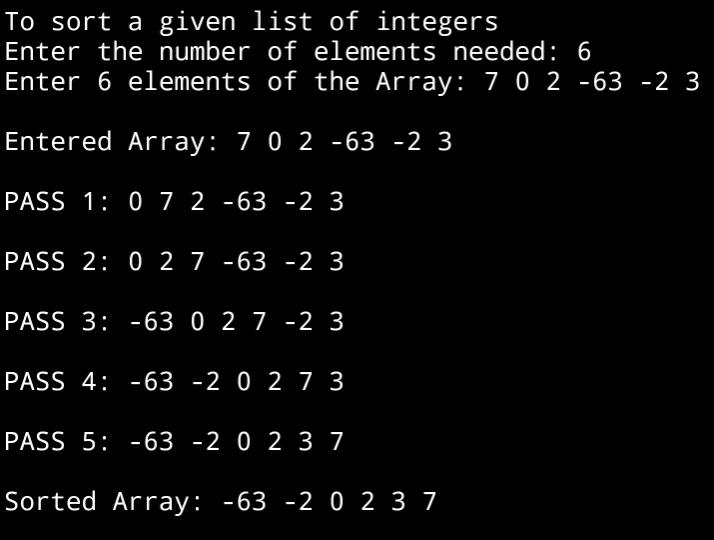
disp(arr,num);

return 0 ;

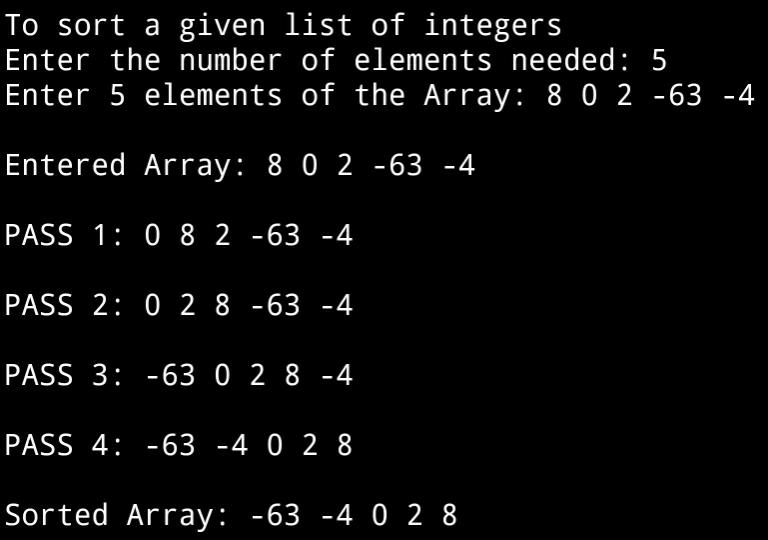
}

**4. OUTPUT**

**SET 1 :** Even Sized Input Array



**SET 2 :** Odd Sized Input Array



**5. DISCUSSIONS**

1. **Variable Description:**

**In insertionsort function**

* **i,j:** loop counters to access the array
* **temp:** to store the leftmost element of unsorted sublist

**In main function**

* **i,j:** loop counters to access the array
* **num:** size of the array entered by the user

**In getarr and disp function**

* **i:** loop counter

1. **Limitations**

* The program uses an integer array to hold the list of integers entered by the user, since arrays are static data structures ,their size cannot be manipulated once it is allocated in the memory.

1. **Uses**

* The above program can be used to sort any list of integers in ascending order. It can be used by educational institutions to arrange a list of students in ascending order of their roll numbers

1. **Future Scope**

* The list of integers can be stored in a linked list , enabling more elasticity in manipulation of size of the list.