**ASSIGNMENT - 3**

**1.PROBLEM STATEMENT**

Write a program in C to implement bubble sort

**2.ALGORITHMS**

**Algorithm Bubble\_Sort**

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

**Output:** The same input array in sorted order

**Remarks:** Elements are sorted in ascending order and it is assumed that the array is not empty

**Steps:**

1. **For** i=1 to (n-1) **do** // bubble sort needs n-1 iterations to complete
2. **For** j=1 to (n-i) **do** // traversing the unsorted array
3. **If**(arr[j]>arr[j+1] **then** // if previous is greater
4. **Swap**(arr[j],arr[j+1]) // swap the two elements
5. **EndIf**
6. **EndFor**
7. **Endfor**
8. **Stop**

**Algorithm Swap**

**Input:** The two variables named a and b whose data is to be swapped

**Output:** The two variables a and b with interchanged data

**Remarks:** The variables must be passed as pointers

**Steps:**

1. a = a + b
2. b = a - b
3. a = a - b
4. **Stop**

**3. Source Code**

#include<stdio.h>

#include<stdlib.h>

// function to swap the contents of two variables

void swap(int \*a,int \*b)

{

\*a = \*a+\*b;

\*b = \*a-\*b;

\*a = \*a-\*b;

}

// function to display an array

void disparr(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 bubblesort(int \*arr,int arrlen)

{

int i,j;

for(i=0;i<arrlen-1;i++)// loop to perform n-iterations

{

for(j=0;j<arrlen-1-i;j++) // access unsorted part

{

if(arr[j]>arr[j+1]) // if the order is wrong

swap(&arr[j],&arr[j+1]); //interchange the elements

}

printf("\n\nPASS %d: ",i+1);

disparr(arr,arrlen);

}

}

int main(void)

{

int \*arr,num,i;

printf("To Sort An Array");

printf("\nEnter The Length Of The Array: ");

scanf("%d",&num);

//checking if at least two elements are present

if(num<2)

{

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

return 0;

}

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

printf("\nEnter %d Elements Of The Array: ",num);

getarr(arr,num); // taking input in array

printf("\nEntered Array: ");

disparr(arr,num); // displaying the entered array

bubblesort(arr,num);

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

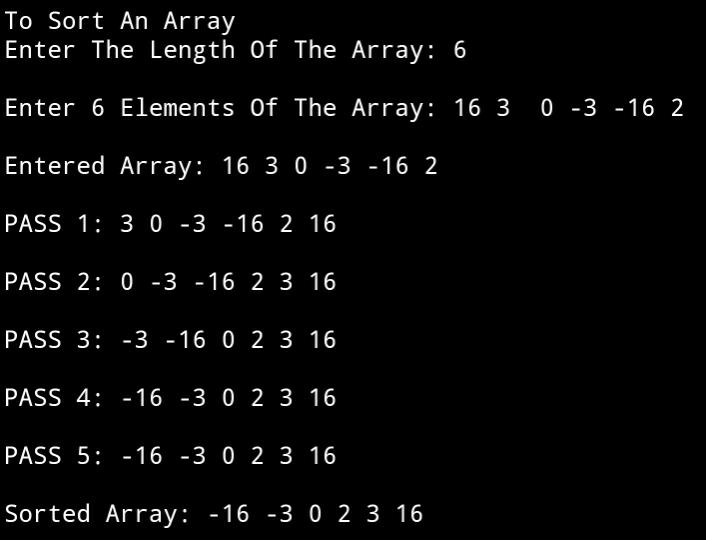
disparr(arr,num);

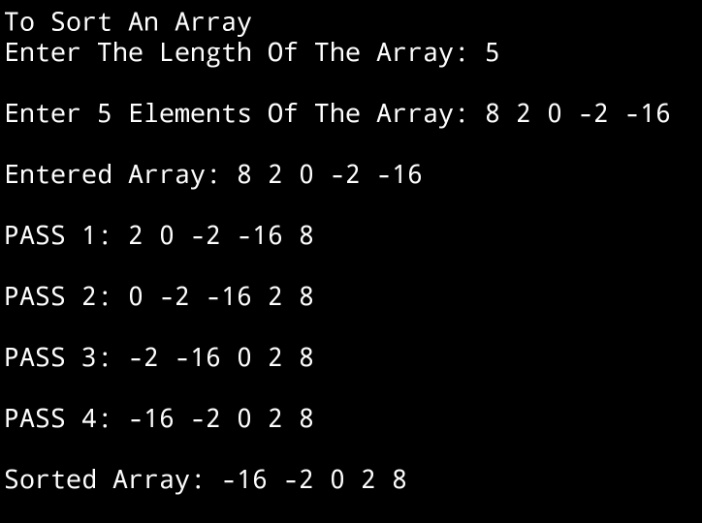
return 0;

}

**4. OUTPUT**

**SET 1:** Even Sized Input Array



**SET 2:** Odd Sized Input Array

1. **DISCUSSIONS**
2. **Variable Description**

**In main function**

* **num:** size of the array entered by the user
* **i:** loop counter to access the array
* **\*arr:** pointer to an array

**In bubblesort function**

* **i,j:** loop counters to access the array

**In disparr and getarr functions**

* **i**: loop counter

1. **Limitations**

* The time complexity for bubble sort algorithm is O(n2) , therefore, for a very large list of integers , the algorithm is less efficient
* 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 population survey groups to sort a list of people in ascending order of their age.

1. **Future Scope**

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