LAB - 3:

QUICK SORT:

PROGRAM:

```
//Sorted Array
#include <stdio.h>
void swap(int a[], int i, int j)
{
   int temp = a[i];
   a[i] = a[j];
   a[j] = temp;
}

int partition(int arr[], int low, int high)
{
   int pivot = arr[high];
   int i = (low - 1);

   for (int j = low; j <= high - 1; j++)
   {
      if (arr[j] <= pivot)
      {
         i++;
         swap(arr, i, j);
      }
   }
   swap(arr, i + 1, high);
   return (i + 1);
}

void quickSort(int arr[], int low, int high)
{
   if (low < high)
   {
      int pi = partition(arr, low, high);
      quickSort(arr, low, pi - 1);
      quickSort(arr, pi + 1, high);
   }
}

void printArray(int arr[], int size)
{
   for (int i = 0; i < size; i++)
```

```
    {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

void main()
{
    int size, arr[100];
    printf("\nEnter the number of elements in the array: ");
    scanf("%d",&size);
    for (int i = 0 ; i < size ; i++)
    {
        printf("\nEnter the element %d: ",i+1);
        scanf("%d",&arr[i]);
    }
    printArray(arr, size);
    quickSort(arr, 0, size - 1);
    printf("After doing Quick sort, Sorted array: \n");
    printArray(arr, size);
}
```

OUTPUT:

```
Enter the number of elements in the array: 4

Enter the element 1: 100

Enter the element 2: 77

Enter the element 3: 73

Enter the element 4: 97
100 77 73 97
After doing Quick sort, Sorted array:
73 77 97 100


=== Code Exited With Errors ===
```

MERGE SORT:

PROGRAM:

```
#include <stdio.h>
#define MAX_SIZE 100
int a[MAX_SIZE], b[MAX_SIZE];
int size;
void merging(int low, int mid, int high)
{
   int l1, l2, i;
   for(l1 = low, l2 = mid + 1, i = low; l1 <= mid && l2 <= high; i++)
   {
      if(a[l1] <= a[l2])
      {
         b[i] = a[l1++];
      }
      else
      {
         b[i] = a[l2++];
      }
   }
   while (l1 <= mid)
   {
      b[i++] = a[l1++];
   }
   while (l2 <= high)
   {
      b[i++] = a[l2++];
   }
   for(i = low; i <= high; i++)
   {
      a[i] = b[i];
   }
}

void sort(int low, int high)
{
   int mid;
   if(low < high)
   {
      mid = (low + high) / 2;
      sort(low, mid);
      sort(mid + 1, high);
      merging(low, mid, high);
```

```
    }
}

int main()
{
    printf("Enter the number of elements in the array: ");
    scanf("%d", &size);
    if(size > MAX_SIZE)
    {
        printf("Size exceeds the maximum allowed value of %d\n", MAX_SIZE);
        return 1;
    }
    for (int i = 0; i < size; i++)
    {
        printf("Enter the element %d: ", i + 1);
        scanf("%d", &a[i]);
    }
    printf("Before sorting: ");
    for(int i = 0; i < size; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
    sort(0, size - 1);
    printf("After doing Merge sort, Sorted array is: ");
    for(int i = 0; i < size; i++)
    {
        printf("%d ", a[i]);
    }
    printf("\n");
    return 0;
}
```

OUTPUT:

```
Enter the number of elements in the array: 4
Enter the element 1: 100
Enter the element 2: 77
Enter the element 3: 73
Enter the element 4: 97
Before sorting: 100 77 73 97
After doing Merge sort, Sorted array is: 73 77 97 100


=== Code Execution Successful ===
```

SHELL SORT:

PROGRAM:

```
//Shell sort
#include <stdio.h>

void shellsort(int array[], int n)
{
   for (int inc = n / 2; inc > 0; inc /= 2)
   {
      for (int i = inc; i < n; i++)
      {
         int temp = array[i];
         int j;
         for (j = i; j >= inc && array[j - inc] > temp; j -= inc)
         {
            array[j] = array[j - inc];
         }
         array[j] = temp;
      }
   }
}

void printarray(int array[], int size)
{
   for (int i = 0; i < size; ++i)
```

```
    {
        printf("\n%d ", array[i]);
    }
    printf("\n");
}

void main()
{
    int size, data[100];
    printf("\nEnter the number of elements in the array: ");
    scanf("%d",&size);
    for (int i = 0 ; i < size ; i++)
    {
        printf("\nEnter the element %d: ",i+1);
        scanf("%d",&data[i]);
    }
    printarray(data, size);
    shellsort(data, size);
    printf("After doing shell sorting, Sorted array is: ");
    printarray(data, size);

}
```

OUTPUT:

```
Enter the number of elements in the array: 4

Enter the element 1: 100

Enter the element 2: 77

Enter the element 3: 73

Enter the element 4: 97

100
77
73
97
After doing shell sorting, Sorted array is:
73
77
97
100


=== Code Exited With Errors ===
```