1.INSERTION SORT:

```
#include <stdio.h>
void insertionSort(int arr[], int n)
{
   for (int i = 1; i < n; ++i) {
      int key = arr[i];
      int j = i - 1;

      while (j >= 0 && arr[j] > key) {
         arr[j + 1] = arr[j];
         j = j - 1;
      }
      arr[j + 1] = key;
   }
}

void printArray(int arr[], int n)
{
   for (int i = 0; i < n; ++i)
      printf("%d ", arr[i]);
   printf("\n");
}



int main()
{
   int arr[] = { 7,3,10,4,1,11 };
   int n = sizeof(arr) / sizeof(arr[0]);

   insertionSort(arr, n);
   printArray(arr, n);
```

```
   return 0;
}
```

Output:

1 3 4 7 10 11

2.MERGE SORT:

```
#include <stdio.h>

#define max 8

int a[8] = { 16,9,2,20,14,3,10,7};
int b[8];

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);
  } else {
    return;
  }
}

int main() {
  int i;

  printf("List before sorting\n");

  for(i = 0; i <= max; i++)
    printf("%d ", a[i]);

  sort(0, max);

  printf("\nList after sorting\n");

  for(i = 0; i <= max; i++)
    printf("%d ", a[i]);
}
```

Output:

List before sorting

16 9 2 20 14 3 10 7 0

List after sorting

0 2 3 7 9 10 14 16 20

3.RADIX SORT:

```
#include <stdio.h>



int getMax(int arr[], int n)
{
   int mx = arr[0];
   for (int i = 1; i < n; i++)
      if (arr[i] > mx)
         mx = arr[i];
   return mx;
}



void countSort(int arr[], int n, int exp)
{

   int output[n];
   int i, count[10] = { 0 };



   for (i = 0; i < n; i++)
      count[(arr[i] / exp) % 10]++;
```

```
    for (i = 1; i < 10; i++)
        count[i] += count[i - 1];
 // Build the output array
    for (i = n - 1; i >= 0; i--) {
        output[count[(arr[i] / exp) % 10] - 1] = arr[i];
        count[(arr[i] / exp) % 10]--;
    }


    for (i = 0; i < n; i++)
        arr[i] = output[i];
}

void radixsort(int arr[], int n)
{

    int m = getMax(arr, n);


    for (int exp = 1; m / exp > 0; exp *= 10)
        countSort(arr, n, exp);
}


void print(int arr[], int n)
{
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
}
```

```
int main()
{
    int arr[] = { 170, 45, 75, 90, 802, 24, 2, 66 };
    int n = sizeof(arr) / sizeof(arr[0]);



    radixsort(arr, n);


    print(arr, n);
    return 0;
}
```