RADIX SORT

Radix sort is a non-comparative integer sorting algorithm that sorts data with integer keys by grouping keys by the individual digits which share the same significant position and value.

Basic Steps to Be Performed:

Each key is first figuratively dropped into one level of buckets corresponding to the value of the rightmost digit. Each bucket preserves the original order of the keys as the keys are dropped into the bucket. There is a one-to-one correspondence between the number of buckets and the number of values that can be represented by a digit. Then, the process repeats with the next neighbouring digit until there are no more digits to process. In other words:

1. Take the least significant digit of each key.
2. Group the keys based on that digit, but otherwise keep the original order of keys.
3. Repeat the grouping process with each more significant digit.

The sort in step 2 is usually done using bucket sort or counting sort, which are efficient in this case since there are usually only a small number of digits.

Step-by-step example:

Original, unsorted list:

170, 45, 75, 90, 802, 24, 2, 66

Sorting by least significant digit (1s place) gives:

17<u>0</u>, 9<u>0</u>, 80<u>2</u>, <u>2</u>, 2<u>4</u>, 4<u>5</u>, 7<u>5</u>, 6<u>6</u>

Sorting by next digit (10s place) gives:

8<u>0</u>2, 2, <u>2</u>4, <u>4</u>5, <u>6</u>6, 1<u>7</u>0, <u>7</u>5, <u>9</u>0

Sorting by most significant digit (100s place) gives:

2, 24, 45, 66, 75, 90, <u>1</u>70, <u>8</u>02

It is important to realize that each of the above steps requires just a single pass over the data, since each item can be placed in its correct bucket without having to be compared with other items.

C program for Radix Sort:

```
#include<stdio.h>
#include<conio.h>
radix_sort(int array[], int n);
void main()
```

```
{
 int array[100],n,i;
 clrscr();
 printf("Enter the number of elements to be sorted: ");
 scanf("%d",&n);
 printf("\nEnter the elements to be sorted: \n");
 for(i = 0 ; i < n ; i++ )
 {
  printf("\tArray[%d] = ",i);
  scanf("%d",&array[i]);
 }
 printf("\nArray Before Radix Sort:");  //Array Before Radix Sort
 for(i = 0; i < n; i++)
 {
  printf("%8d", array[i]);
 }
 printf("\n");
 radix_sort(array,n);
 printf("\nArray After Radix Sort: ");  //Array After Radix Sort
 for(i = 0; i < n; i++)
 {
  printf("%8d", array[i]);
 }
 printf("\n");
 getch();
}
radix_sort(int arr[], int n)
{
 int bucket[10][5],buck[10],b[10];
 int i,j,k,l,num,div,large,passes;
 div=1;
 num=0;
 large=arr[0];
 for(i=0 ; i<n ; i++)
 {
```

```
 if(arr[i] > large)
  {
   large = arr[i];
  }
 while(large > 0)
 {
  num++;
  large = large/10;
 }
 for(passes=0 ; passes<num ; passes++)
 {
  for(k=0 ; k<10 ; k++)
  {
   buck[k] = 0;
  }
  for(i=0 ; i<n  ;i++)
  {
   l = ((arr[i]/div)%10);
   bucket[l][buck[l]++] = arr[i];
  }
  i=0;
  for(k=0 ; k<10 ; k++)
  {
   for(j=0 ; j<buck[k] ; j++)
   {
    arr[i++] = bucket[k][j];
   }
  }
  div*=10;
 }
}
```