//Created By Ritwik Chandra Pandey on 6 Nov' 21
//Bucket Sort: Max element method

/* In the Bucket Sort algorithm, all the elements to be sorted are initially distributed into number of buckets. Then each bucket is individually sorted and then they are merged back to get the sorted order.

There are two ways of taking the buckets:

1. Taking the number of buckets equal to the maximum element in array to be sorted. For example for sorting 1,3,2,4,5,6,2, we would take 6 buckets as 6 is the maximum of all the elements in the list.
2. Taking a fixed number of buckets and distributing the given elements into the buckets based on a logic.

This is method 2. Here, we distribute the given elements into different buckets and then perform any sorting technique on each bucket and then merge all the elements from the bucket. */

```
#include <stdio.h>
#include <conio.h>

struct bucket {
        int count;
        int* value;
};
void InsertionSort(int a[], int n) {
        int j, p;
        int tmp;
        for(p = 1; p < n; p++) {
                tmp = a[p];
                for(j = p; j > 0 && a[j-1] > tmp; j--)
                        a[j] = a[j-1];
                a[j] = tmp;
        }
}

void BucketSort(int array[],int n, int nbuckets) {
        struct bucket buckets[10];
        int i,j,k;
        for(i=0;i<nbuckets;i++){
                buckets[i].count = 0;
                buckets[i].value = malloc(sizeof(struct bucket)*n);
        }
```

```
        for(i=0;i<n;i++){
                if(array[i]<0){
                        buckets[0].value[buckets[0].count++] = array[i];

                }else if(0<array[i] && array[i]<10){
                        buckets[1].value[buckets[1].count++] = array[i];

                }else{
                        buckets[2].value[buckets[2].count++] = array[i];

                }
        }
        k = 0;
        for(i=0;i<nbuckets;i++){
                InsertionSort(buckets[i].value,buckets[i].count);
                for(j=0;j<buckets[i].count;j++){
                        array[k++] = buckets[i].value[j];
                }
        free(buckets[i].value);
        }



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

int main()  {
        int size,buckets=3;
        int *arr, i;
        printf("Enter array size : ");
   scanf("%d",&size);
        arr = (int*) malloc(size * sizeof(int));
        printf("Enter %d elements : ",size);
   for (i = 0; i < size; i++) {
                scanf("%d", &arr[i]);
   }
```

```
    printf("Before sorting the elements are : ");
        printArray(arr,size);
        BucketSort(arr,size,buckets);
        printf("After sorting the elements are : ");
        printArray(arr,size);
        return 0;
}
```