```
/*


Date                :July 24,2019

Author              :Mehul Y Khandhadiya

Problem statement   :Write a generalized function that takes a parameter to
indicate the mode(say 1 for decreasing order, 2 for increasing order, 3 for
increasing order with the Nth element out of order, 4 for a randomly
generated element values) to create a list of elements. The parameter
indicating the number of elements in a statically allocated array(the
maximum size is large enough to run possible iterations to test the time
complexity say 1000000)will be a multiple of 10. Also write appropriate
functions to create a copy of the list and to display the list contents.

Using above functions, write a menu-driven c program to order the list in
ascending sequence using - the selection sort, the insertion sort, the
counting sort and the shell sort.

*/

//header files

#include<stdio.h>

#include<time.h>

#define MX_SIZE 260000

//function prototypes

void selection_sort(int*,int);

void insertion_sort(int*,int);

void shell_sort(int*,int);

void counting_sort(int*,int,int);

int ascending_order(int*,int);

int descending_order(int*,int);

int asc_nearsort(int*,int);

int random_order(int*,int);

int counting_arr(int*,int);

void copy(int*,int*,int);

void display(int*,int);


//the driver function
```

```
int main()
{
    int a[MX_SIZE],b[MX_SIZE],num,choice,choice1,n1,n,n2,k;
    double elapsed_t;
    time_t t1,t2;
    do
    {
      printf("\nWant to create a list?\n Enter 1 for yes:");
      scanf("%d",&num);
      if(num==1)
      {
        do
        {
         printf("\nEnter no. of elements in multiple of 10:");
         scanf("%d",&n);
         }while(n%10!=0);
            do
            {
             printf("\nWhich type of list you want?\n");
             printf(" 1.ascending 2.descending 3.ascending near     sorted
            4.random 5.array for counting sort");
             printf("\nEnter choice:");
             scanf("%d",&choice);
             switch(choice)
             {
               case 1:ascending_order(a,n);
                      printf("\nDo you want to copy the generated  array?
                      Enter 1 for yes or any other no. to exit:");
                      scanf("%d",&n2);
                      if(n2==1) {
                        copy(a,b,n);
```

```
        }
        break;
case 2:descending_order(a,n);
       printf("\nDo you want to copy the generated array?
       Enter 1 for yes or any other no. to exit:");
       scanf("%d",&n2);
       if(n2==1) {
        copy(a,b,n);
        }
        break;
case 3:asc_nearsort(a,n);
       printf("\nDo you want to copy the   generated array?
       Enter 1 for yes or any other no. to exit:");
       scanf("%d",&n2);
       if(n2==1)
        {
        copy(a,b,n);
        }
        break;
case 4:random_order(a,n);
       printf("\nDo you want to copy the generated array?
       Enter 1 for yes or any other no. to exit:");
       scanf("%d",&n2);
       if(n2==1)
        {
        copy(a,b,n);
        }
        break;
case 5:printf("\n array for counting sort");
       counting_arr(a,n);
       printf("\nDo you want to copy the generated array?”);
       scanf("%d",&n2);
```

```
            if(n2==1)
              {
              copy(a,b,n);
              }
              break;
}

do
{
printf("\nWhich type of sorting procedure do you want?\n");
printf("1.insertion sort 2.selection sort 3.shell sort
4.counting sort 5.exit \n");
printf("\nEnter choice:");
scanf("%d",&choice1);
switch(choice1)
{
      case 1: t1=clock();
              insertion_sort(a,n);
              t2=clock();
              elapsed_t=((double)(t2-t1)/CLOCKS_PER_SEC);
              printf("\n\t Time required=%gseconds",elapsed_t);
              printf("\n Do you want to display sorted.enter 1
            for yes and any other no.to exit");
              scanf("%d",&n1);
              if(n1==1)
               {
               display(a,n);
               }
               break;
      case 2: t1=clock();
              selection_sort(a,n);
```

```
            t2=clock();
            elapsed_t=((double)(t2-t1))/CLOCKS_PER_SEC;
            printf("\n\t Time required=%gseconds",elapsed_t);
            printf("\n Do you want to display sorted array?\n
          enter 1 for yes and any other no.to exit");
            scanf("%d",&n1);
            if(n1==1)
            {
            display(a,n);
            }
            break;
    case 3: t1=clock();
            shell_sort(a,n);
            t2=clock();
            elapsed_t=((double)(t2-t1))/CLOCKS_PER_SEC;
            printf("\n\t Time required=%gseconds",elapsed_t);
            printf("\n Do you want to display sorted array?\n
            enter 1 for yes and any other no.to exit");
            scanf("%d",&n1);
            if(n1==1)
            {
            display(a,n);
            }
            break;
     case4: t1=clock();
            counting_sort(a,n,k);
            t2=clock();
            elapsed_t=((double)(t2-t1))/CLOCKS_PER_SEC;
            printf("\n\t Time required=%gseconds",elapsed_t);
            printf("\n Do you want to display sorted array?\n
            enter 1 for yes and any other no.to exit");
            scanf("%d",&n1);
```

```
                        if(n1==1)
                        {
                        display(sorted,n);
                        }
                        break;
                case 5:printf("terminating condition entered”);
                        exit(0);
        }
      }while(choice1!=0);
    }while(choice!=0);
}
   else
   printf("\nTerminating\n ");
}while(num!=0);
return 0;
}
void copy(int a[],int b[],int n)
{
    int i,n2;
    for(i=0;i<n;i++)
    {
        b[i]=a[i];
    }
    printf("\nEnter 1 if you want to display copied array otherwise 0:");
    scanf("%d",&n2);
    if(n2==1)
    {
        display(a,n);
    }
}
```

```
void display(int a[],int n)
{
     int i;
     for(i=0;i<n;i++)
     {
        printf("%d\n",a[i]);
     }
}
int ascending_order(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        a[i]=987698/7+5*i;
    }
    return;
}
int descending_order(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        a[i]=987698/7-5*i;
    }
    return;
}
int asc_nearsort(int a[],int n)
{
    int i;
    for(i=0;i<n;i++)
```

```
{
        if(i%10==0)
            a[i]=987698/7+5*i;
        else
            a[i]=987698/7+10*i;
    }
return;
}
int random_order(int a[],int n)
{
    int i,offset=987698;
    rand(offset);
    for(i=0;i<n;i++)
    {
        a[i]=rand()/11;
    }
return;
}
int counting_arr(int a[],int n)
{
    int i,k;
    printf("\nEnter the range:");
    scanf("%d",&k);
    for(i=0;i<n;i++)
    {
        if(n>0)
        {
            for(i=0;i<n;i++)
            {
                a[i]=rand()%k;
```

```
            }
        }
    }
return;
}
void insertion_sort(int a[],int n)
{
    int i,j,temp;
    printf("sorting of array\n");
    for(i=1;i<n;i++)
     {
        j=i;
        while(j>0 && a[j]<a[j-1])
        {
            temp=a[j];
            a[j]=a[j-1];
            a[j-1]=temp;
         j--;
         }
      }

}
void selection_sort(int a[],int n)
{
     int i,j,temp;
     for(i=0;i<=n-2;i++)
     {
        for(j=i+1;j<=n-1;j++)
        {
            if(a[i]>a[j])
```

```
            {
                temp=a[i];
                a[i]=a[j];
                a[j]=temp;
                                                }
            }
        }
     }

 void shell_sort(int a[],int n)
 {
    int i,key,gap,j;
    for(gap=n/2;gap>=1;gap=(gap/2))
    {
        for(i=gap;i<n;i++)
        {
            key=a[i];
            for(j=i;j>=gap && a[j-gap]>key;j=j-gap)
            {
                a[j]=a[j-gap];
            }
            a[j]=key;
        }
    }
 }

void counting_sort(int a[],int n,int k) {
    int i,kount[k+1],sorted[MX_SIZE],j;
    for(i=0;i<=k;i++)
    {
```

```
        kount[i]=0;
    }
        for(j=1;j<=n;j++)
        {
            kount[a[j]]=kount[a[j]]+1;
        }
        for(i=1;i<=k;i++)
        {
            kount[i]=kount[i]+kount[i-1];
        }
             for(j=n;j>=1;j--){
             sorted[kount[a[j]]-1]=a[j];
             kount[a[j]]=kount[a[j]]-1;
            }
}
```