Problem Statement:

Write a program to implement sieve sort with suitable elements.

Algorithm:

Input: Number of elements n and n elements stored in a data structure (here, integer type array a).
Output: Sorted array.
Data Structure: Array data structure.

Step 1: Declare an array a.
Step 2: Take the first element and find the corresponding numbers in the array which are smaller than that element and pass it in another array.
Step 3: Take the second element and find the corresponding numbers in the array which are smaller than that element and pass it in another array (excluding the ones already picked up in the first selection in step 2).
Step 4: In each pass, take the sub-lists (formed in steps 2 & 3) and arrange them in reverse order (un-sorted).
Step 5: Repeat steps 2, 3 & 4 (for each pass) until the array is sorted. If there is a single element left in any pass, put it in the highest position of its corresponding sub-list.

Source code:

```
#include<stdio.h>
#include<malloc.h>
int n;
int sieve(int **b, int a[],int y)
{
        int i,j,k=1,subtree=1,f,flag=1;
        b[0][0]=a[0];
        while(k<n)
        {
                i=0;
                f=0;
                while(i<subtree && f!=1)
                {
                        j=0;
                        if(b[i][0]>a[k])
                        {
                                while(b[i][j]!=' ')
                                        j++;
                                b[i][j]=a[k];
                                f=1;
                        }
                        i++;
                }
                if(f==0)
                {
                        subtree++;
                        b[i][0]=a[k];
                }
                k++;
        }
        f=0;
        for(i=0;i<subtree;i++)
        {
                k=0;
                while(b[i][k]!=' ')
                        k++;
                        for(j=k-1;j>=0;j--)
                                a[f++]=b[i][j];
                        printf("SUBLIST- %d : ",i+1);
                        for(j=0;j<k;j++)
                        {
                                if(y==0)
                                        printf("%d ",b[i][j]);
                                else
                                        printf("%d ",b[i][k-j-1]);
                        }
                        printf("\n");
        }
        printf("\n");
        for(i=0;i<n;i++)
                printf("%d ",a[i]);
        printf("\n");
        for(i=0;i<n-1;i++)
        {
                if(a[i]>a[i+1])
                {
                        flag=0;
                        break;
                }
        }
        return(flag);
}
main()
{
        int *a,**b,i,j,k,t,x=0;
        printf("Enter the number of elements : ");
        scanf("%d",&n);
        a=(int*)malloc(n * sizeof(int));
        b=(int**)malloc(n * sizeof(int *));
   for (i=0; i<n; i++)
      b[i]=(int*)malloc(n * sizeof(int));
        printf("Enter the elements :\n");
        for(i=0;i<n;i++)
        {
                printf("a[%d] : ",i+1);
                scanf("%d",&a[i]);
        }
        i=0;
        while(x!=1)
        {
                printf("\nPASS- %d\n",i+1);
                for(j=0;j<n;j++)
                {
                        for(k=0;k<n;k++)
```

```
                                b[j][k]=' ';
                }
                if(i%2==0)
                        x=sieve(b,a,0);
                else
                        x=sieve(b,a,1);
                i++;
        }
        printf("\nSORTED ELEMENTS ARE :\n");
        for(i=0;i<n;i++)
                printf("%d ",a[i]);
}
```

Output:

```
Enter the number of elements : 5
Enter the elements :
a[1] : 20
a[2] : 11
a[3] : 15
a[4] : 18
a[5] : 29

PASS- 1
SUBLIST- 1 : 20 11 15 18
SUBLIST- 2 : 29

18 15 11 20 29

PASS- 2
SUBLIST- 1 : 11 15 18
SUBLIST- 2 : 20
SUBLIST- 3 : 29

11 15 18 20 29

SORTED ELEMENTS ARE :
11 15 18 20 29
```