ARRAYS

Sorting the elements of an array

- ✓ Sorting is one of the applications of the single dimensional array.
- ✓ Sorting is the technique of arranging elements of an array either in ascending or in descending order.
- ✓ Simple methods of searching are:
  - ➢ Bubble Sort.
  - ➢ Selection Sort.

Bubble Sort

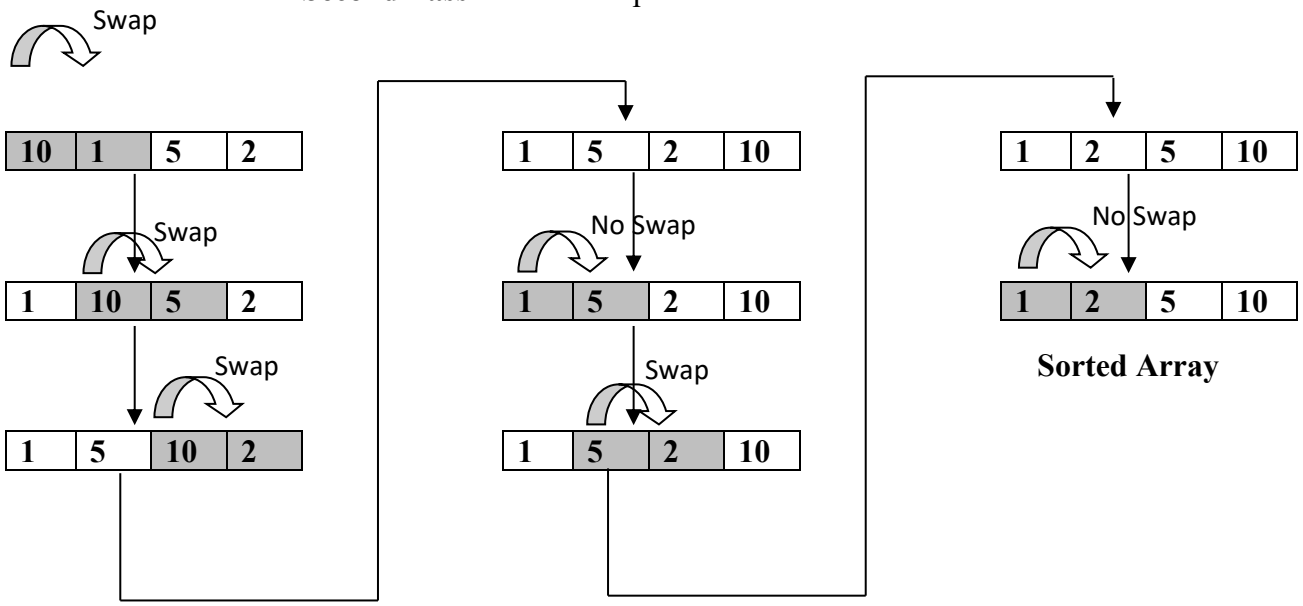
- ✓ Bubble sort sometimes referred to as sinking sort.
- ✓ *Procedure for Bubble sort to arrange the elements in ascending order:*
  - ➢ In Bubble sort each element is compared with the adjacent element.
  - ➢ If the first element is larger than the second then the elements are interchanged, otherwise it is not changed.
  - ➢ Then next element is compared with the adjacent element and same process is repeated for all the elements in the array.
  - ➢ During the first pass, the largest element occupies the last position.
  - ➢ During the next pass the same process is repeated leaving the largest element.

Example: No of passes will be always **number of elements -1 (n-1), here 4-1=3 passes**

First pass – n-i-1 comparisons 4-0-1=3

Third Pass – n-i-1 Comparisons 4-2-1 =1

Second Pass – n-i-1 Comparisons 4-1-1 =2



Logic:

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

Program:

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

void main()
{
int a[100],n,i,j,temp;
clrscr();
printf("\nEnter the number of elements in the array : ");
scanf("%d",&n);
printf("\nEnter %d elements of the array : \n",n);
      for(i=0;i<n;i++)
scanf("%d",&a[i]);
for(i=0;i<n-1;i++)
{
      for(j=0;j<n-i-1;j++)
      {
            if(a[j]>a[j+1])
            {
            temp=a[j];
            a[j]=a[j+1];
            a[j+1]=temp;
            }
      }
}
printf("\n The sorted Array is : \n");
for(i=0;i<n;i++)
printf("%d\t",a[i]);
getch();
}
```

Advantages of Bubble sort:

- Very simple and easy to write.
- Straight forward approach.

Disadvantages of bubble sort:

- It runs slowly and hence it is not efficient. More efficient sorting techniques are available.
- Even if the elements are sorted, n-1 passes are required to sort.

Selection Sort

- ✓ The selection sort is based on the minimum/maximum technique.
- ✓ By means of a nest of for loops; a pass through the array is made to locate the minimum value. Once this is found, it is placed in the first position of the array.
- ✓ Another pass through the remaining elements is made to the next smallest element, which is placed in the second position, and so on.
- ✓ Once the next-to-last element has been compared with the last one, all the elements of the array have been sorted into ascending order.
- ✓ If there are nelements to be sorted then, the process should be repeated **n-1** times to get required result.

Example:

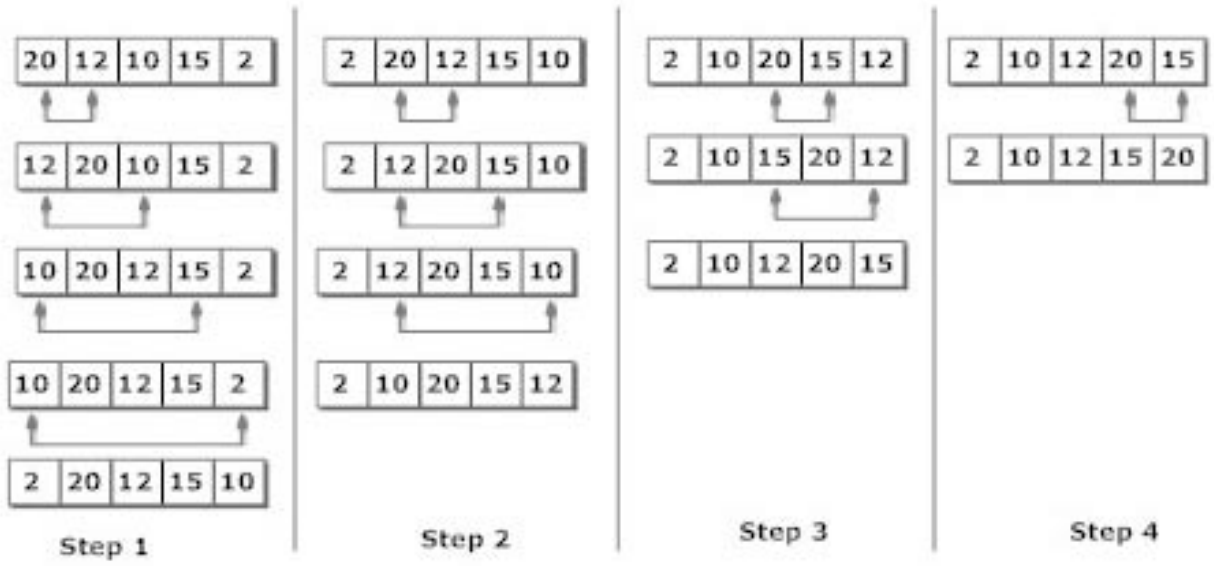


Figure: Selection Sort

Logic:

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

Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[100],n,i,j,temp;
clrscr();
printf("\nEnter the number of elements in the array : ");
scanf("%d",&n);
printf("\nEnter %d elements of the array : \n",n);
for(i=0;i<n;i++)
scanf("%d",&a[i]);

for(i=0;i<n-1;i++)
{
      for(j=i+1;j<n;j++)
      {
      if(a[i]>a[j])
      {
      temp=a[i];
      a[i]=a[j];
      a[j]=temp;
      }
      }
}
printf("\n The sorted Array is : \n");
for(i=0;i<n;i++)
printf("%d\t",a[i]);
getch();
}
```