Practical no -1

Aim: Write a program to sort the array element using bubble sort.

Bubble sort:

Bubble sort is one of the sorting method .the method is to arrange the element in ascending or descending order. for to arrange the element in ascending order ,to begin with the 0th element is compared with the 1st element .if it is found to be greater than the 1st element then they are interchanged. Then the 1 st element is compared with the 2nd element if it is found to be greater, then they are interchanged. In this same way all the elements (excluding last) are compared with their next element and are interchanged if required. This is the first iteration and on completing this iteration the largest element gets placed at the last location or last position. Similarly, in the second iteration the second largest element get placed at the second last position in the list. As a result after all the iteration the list becomes a sorted list.

Analysis

Each of these algorithms requires ***n***-1 passes: each pass places one item in its correct place. (The ***n^th^*** is then in the correct place also.) The ***i^th^*** pass makes either ***i***or ***n - i*** comparisons and moves. So:

$$\begin{aligned} T(n) &= 1 + 2 + 3 + \ldots + (n-1) \\ &= \sum_{i=1}^{n-1} i \\ &= \frac{n}{2}(n-1) \end{aligned}$$

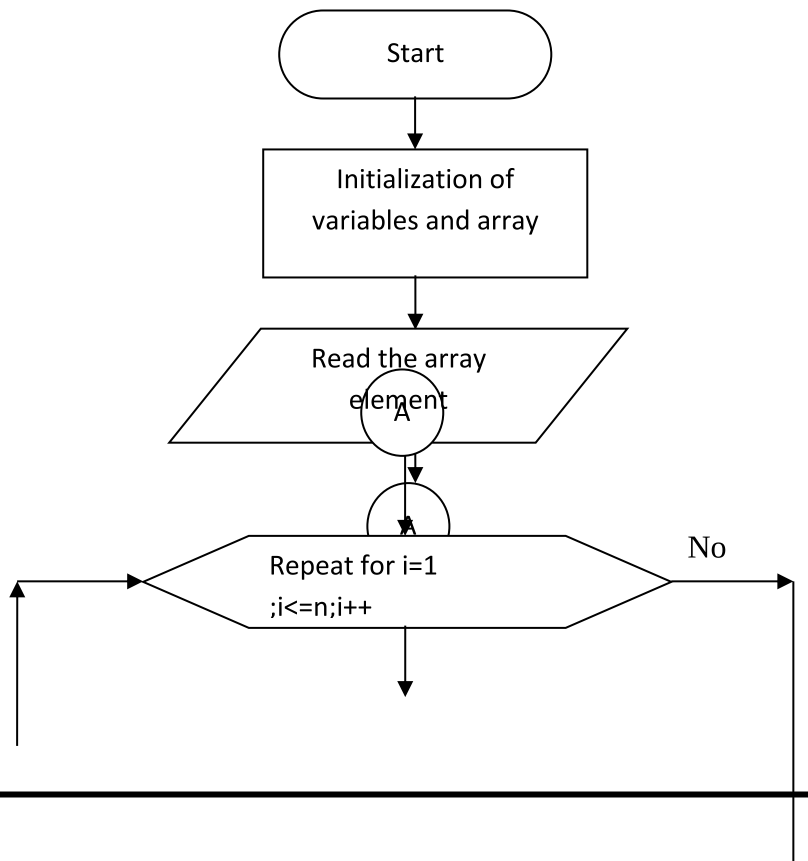
or **O(*n*^{2})** - but we already know we can use heaps to get an **O(*n* log*n*)** algorithm. Thus these algorithms are only suitable for small problems where their simple code makes them faster than the more complex code of the **O(*n* log*n*)** algorithm. As a rule of thumb, expect to find an **O(*n* log*n*)** algorithm faster for *n*>10 - *but the exact value depends very much on individual machines!*.

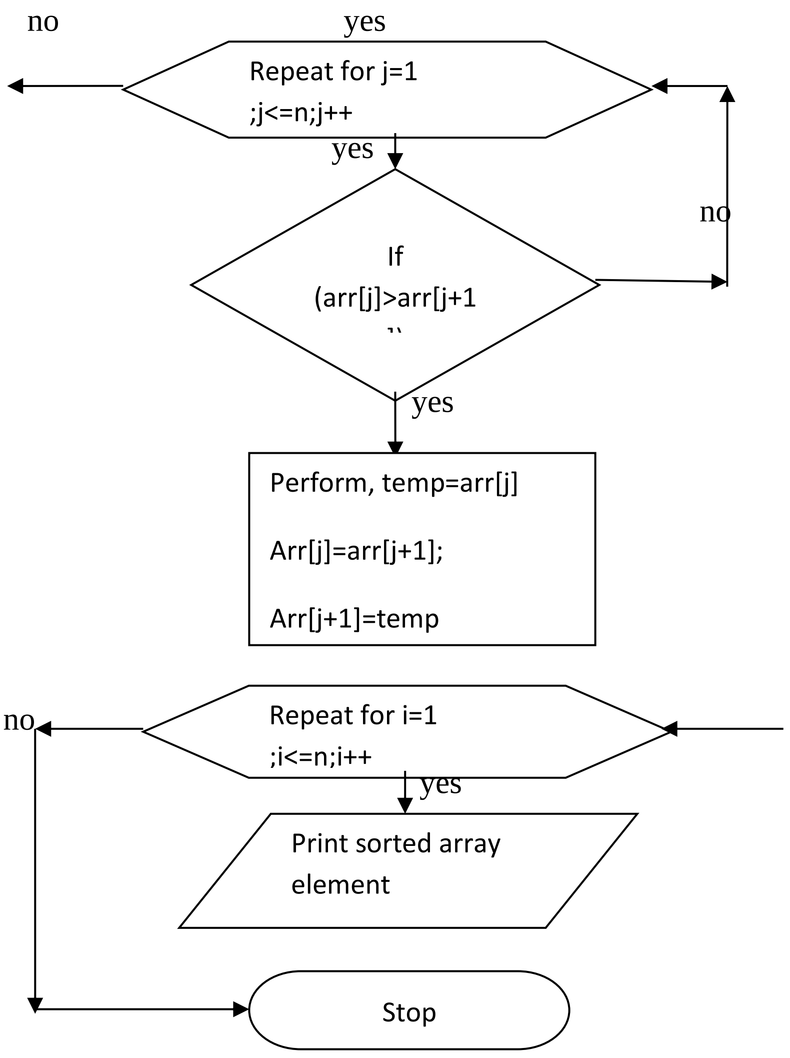
They can be used to squeeze a little bit more performance out of fast sort algorithms

Algorithm:

```
Bubble_sort (DATA,N)
Here DATA is an array with N elements.
This algorithm sort the element DATA.
STEPS:
1. Repeat step 2 and 3 for
        K=0 TO N-1
2. Set PTR=1
3. Repeat while PTR<=N-K
    [Execute pass]
a)  If DATA[PTR] >DATA [PTR+1]
    Then interchange DATA [PTR] and DATA [PTR+1]
b) Set PTR=PTR+1
        [End of inner loop]
[End of step 1 outer loop]
4. Exit
```

Flowchart-





Program:

```
#include<conio.h>
#include<stdio.h>
void main()
        {
        int arr[50],i,j,temp,dim;
        clrscr();
        printf("Enter the dimension");
        scanf("%d",&dim);
        printf("\nEnter the array element:\n");
        for(i=1;i<=dim;i++)
                {
```

```
                scanf("%d",&arr[i]);
                }
        for(i=1;i<=dim;i++)
        {
                for(j=1;j<=dim-i;j++)
                {
                        if(arr[j]>arr[j+1])
                        {
                                temp=arr[j];
                                arr[j]=arr[j+1];
                                arr[j+1]=temp;
                        }
                }
        }
                printf("\nSorted array is:-\n");
                for(i=1;i<=dim;i++)
                        printf("%d ",arr[i]);

        getch();
}
```

Output:

```
Enter the dimension 5
Enter the array element:
8
9
5
```

```
4
3
Sorted array is:-
3
4
5
8
9
```