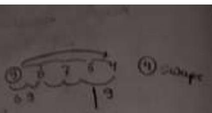
Time Complexity Analysis :



n elements (n-1) swaps
n-2
n-3
⋮
1

Swaps : n-1
n-2
n-3
⋮
1

Let, $S = 1 \quad 2 \quad 3 \; . \; . \; . \; . \; . \; n-2 \quad n-1$

$S = n-1 \quad n-2 \quad n-3 \; . \; . \; . \; . \; . \; 2 \quad 1$

+

$2S = n \quad n \quad n \; . \; . \; . \; . \; . \; n \quad n$

$2S = n(n-1)$

$\therefore S = \frac{n(n-1)}{2}$

$\therefore \frac{n \times (n-1)}{2} = \frac{n^2-n}{2} = O(n^2)$

The time complexity of the provided bubble sort code is $O(n^2)$ in the worst case and $O(n)$ in the best case.

When the array is in reverse order, and each element must be compared and swapped in each pass, resulting in quadratic time complexity : $O(n^2)$.

When the array is already sorted, and no swaps are needed inside the loop, the time complexity is reduced to $O(n)$.

The average case time complexity is also $O(n^2)$.

Experiment : 2b

Date : 02/02/2024

Title : Bubble Sort

Aim : to implement and analyze bubble sort algorithm

Algorithm :

Step 1 : Start

Step 2 : Read the size of the array (n) and its elements from the user.

Step 3 : Declare and populate an array with user-input elements.

Step 4 : Apply the bubble sort algorithm to arrange the elements in ascending order

Step 5 : Print the initially entered array.

Step 6 : Print the array after sorting using bubble sort

Step 7 : Stop

Program Implementation

```
#include <stdio.h>
#include <conio.h>
void printArray (int *A, int n){
    int i ;
    for (i=0; i<n ; i++){
        printf ("%d", A[i]);
    }
    printf ("\n");
}
void bubbleSort (int *A, int n){
    int temp ;
    int i,j ;
    for (i=0; i<n -1 ; i++){
        int flag =0 ;
```

Dry run with sample input and output :

Sample Input :

```
Enter the size of the array : 5
Enter 5 elements :
9
6
7
5
4
```

Sample Output :

```
Original Array : 9 6 7 5 4
Sorted Array : 4 5 6 7 9
```

```
        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;
                flag =1;
            }
        }
        if (flag = =0)
            break;
    }
}
int main(){    clrscr();
    int n;
    printf ("Enter the size of the array : ");
    scanf ("%d", &n);
    int i, A [100] ;
    printf ("Enter %d elements :\n", n);
    for (i=0; i<n;i++) {
        scanf ("%d", &A[i]);
    }
    printf ("Original Array : ");
    printArray (A,n) ;
    bubbleSort (A,n);
    printf("Sorted Array : ");
    printArray (A,n);   getch();
    return 0;
}
```

Result : Bubble sort algorithm was implemented and analyzed successfully.