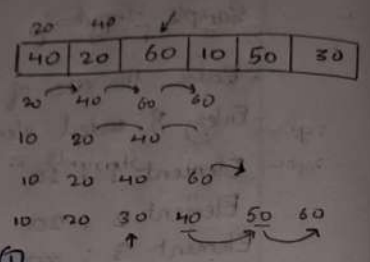
Time Complexity Analysis :



Here, we get recurrence relation as :

$$T(n) = \begin{cases} T(n-1) + n-1 & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$$

$T(n) = T(n-1) + n - 1$ ——— (1)

$T(n-1) = T(n-1-1) + n-1-1$

$= T(n-2) + n - 2$ ——— (2)

$T(n-2) = T(n-2-1) + n-2-1$

$= T(n-3) + n - 3$ ——— (3)

Then, from (1), (2), & (3),

$\therefore T(n) = T(n-2) + n - 2 + n - 1$

$= T(n-3) + n-3 + n-2 + n-1$

$\vdots$

$(n-1)$ times

$= T(n-(n-1)) + \ldots\ldots + n-3 + n-2 + n-1$

$= T(1) + \ldots\ldots + n-3 + n-2 + n-1$

$= 1 + 2 + 3 + \ldots\ldots + n-3 + n-2 + n-1$

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

So, $O(n^2)$

Therefore, the time complexity of the provided insertion sort is $O(n^2)$ in the worst case and $O(n)$ in the best case. The average case time complexity is also $O(n^2)$.

Experiment : 2a

Date : 02/02/2024

Title : Simple Algorithm - Insertion Sort

Aim : To implement and analyze insertion 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 insertion sort algorithm to arrange the elements in ascending order.

Step 5 : Print the initially entered array.

Step 6 : Print the array after sorting using insertion 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]);
    }
    print("\n");
}
```

Dry run with sample input and output

Sample Input :

```
Enter the size of the array : 6
Enter 6 elements :
40
20
60
10
50
30
```

Sample Output :

```
Original Array : 40 20 60 10 50 30
Sorted Array : 10 20 30 40 50 60
```

```
void insertionSort (int *A, int n) {
    int key, i, j;
    for (i=1; i<=n-1; i++) { // Loop for passes
        key = A[i];
        j = i-1;
        while (j>=0 && A[j]>key) { // Loop for each pass
            A[j+1] = A[j];
            j--;
        }
        A[j+1] = key;
    }
}

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

Result : Insertion Sort algorithm was implemented and analyzed successfully.