Experiment No.1
Insertion Sort
Date of Performance:
Date of Submission:

Title: Insertion Sort

Aim: To implement Selection Comparative analysis for large values of 'n'

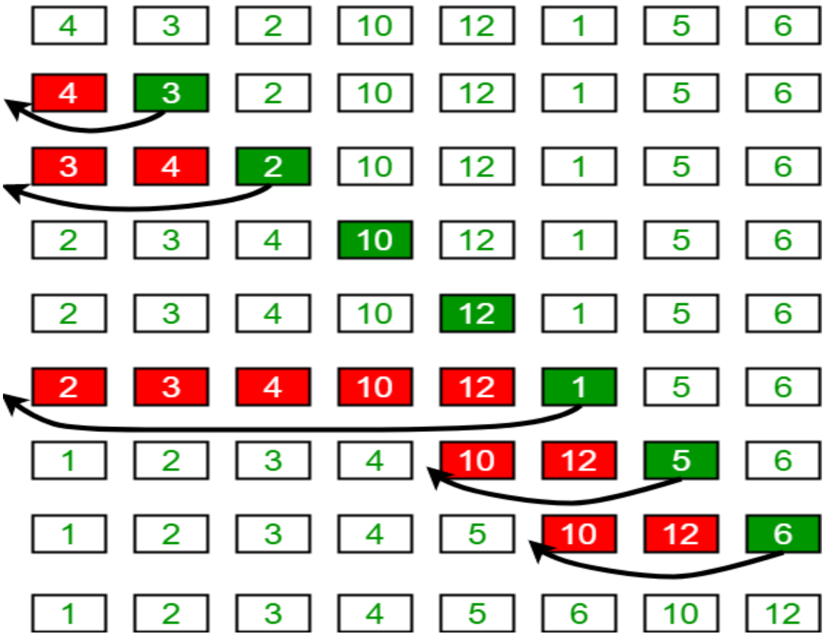
Objective: To introduce the methods of designing and analysing algorithms

Theory:

Insertion sort is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.

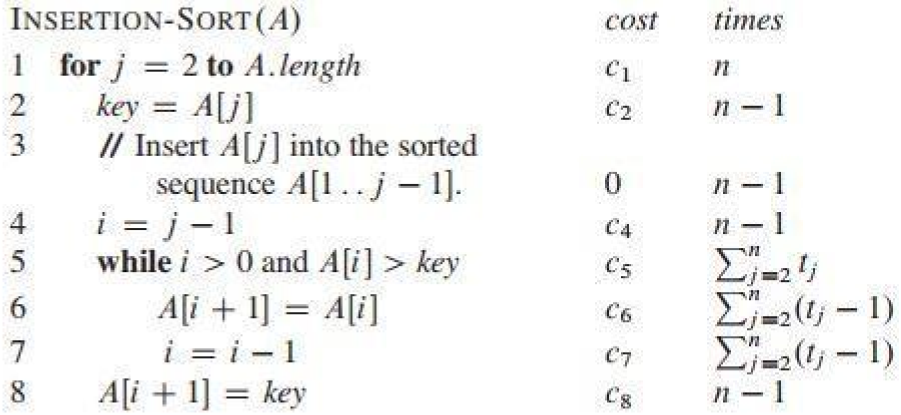
Example:

Insertion Sort Execution Example



4	3	2	10	12	1	5	6
4	3	2	10	12	1	5	6
3	4	2	10	12	1	5	6
2	3	4	10	12	1	5	6
2	3	4	10	12	1	5	6
2	3	4	10	12	1	5	6
1	2	3	4	10	12	5	6
1	2	3	4	5	10	12	6
1	2	3	4	5	6	10	12

Algorithm and Complexity:



```
INSERTION-SORT($A$)                                  cost     times
for $j = 2$ to $A.length$                            $c_1$    $n$
    $key = A[j]$                                     $c_2$    $n-1$
    // Insert $A[j]$ into the sorted
        sequence $A[1..j-1]$.                        0        $n-1$
    $i = j - 1$                                      $c_4$    $n-1$
    while $i > 0$ and $A[i] > key$                   $c_5$    $\sum_{j=2}^{n} t_j$
        $A[i+1] = A[i]$                              $c_6$    $\sum_{j=2}^{n} (t_j - 1)$
        $i = i - 1$                                  $c_7$    $\sum_{j=2}^{n} (t_j - 1)$
    $A[i+1] = key$                                   $c_8$    $n-1$
```

Implementation:

```
#include <stdio.h>



void insertionSort(int arr[], int n) {

   int i, key, j;

   for (i = 1; i < n; i++) {

      key = arr[i];

      j = i - 1;



      /* Move elements of arr[0..i-1], that are greater than key,

         to one position ahead of their current position */

      while (j >= 0 && arr[j] > key) {

         arr[j + 1] = arr[j];
```

```
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}


int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Array before sorting:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");


    insertionSort(arr, n);


    printf("Array after sorting:\n");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
    return 0;

}
```

Conclusion: Merge Sort algorithm has been successfully implemented.