Experiment No.: - 2.2

**Student Name: Vivek** **UID:**

**Branch:** CSE **Section/Group:** 604

**Semester:** 3rd **Date of Performance**: 29-09-2022

**Subject Name:** Data Structures **Subject Code:** 21CSH-211

1. **Aim/Overview of the practical:** WAP to sort an array of integers in ascending order using Insertion Sort.

## Theory

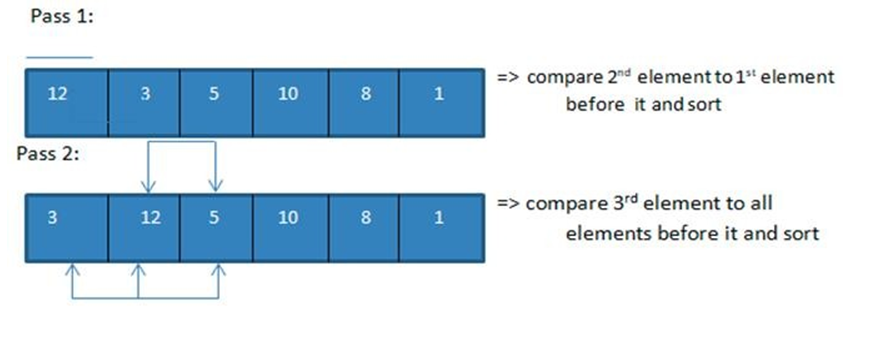
**Insertion sort** is a simple sorting algorithm that works similar to the way you sort playing cards in your hands. The array is virtually split into a sorted and an unsorted part. Values from the unsorted part are picked and placed at the correct position in the sorted part.

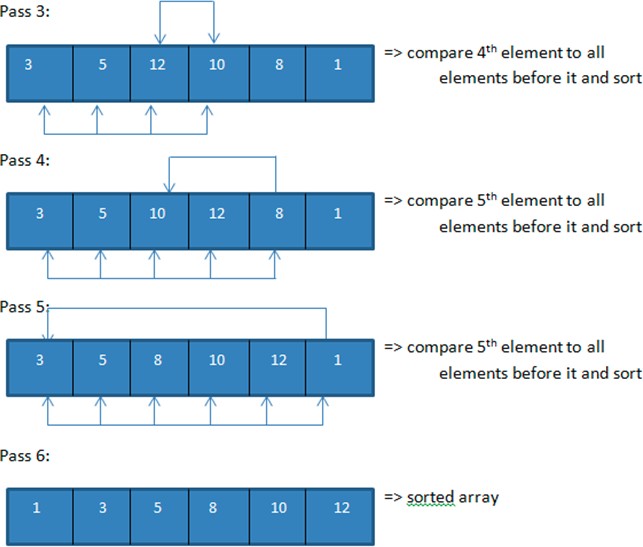
**Example**:

Consider an array



Now for each pass, we compare the current element to all its previous elements. So in the first pass, we start with the second element.





Thus, we require N number of passes to completely sort an array containing N number of elements.

## Algorithms: -

Step 1: If it is the first element it is already sorted, return 1. Step 2: pick the next element.

Step 3: Compare with elements in a sorted sub list.

Step 4: Shift all the elements in the sorted sub list that is greater than value to be sorted.

Step 5: Insert the value.

Step 6: Repeat until the list is sorted.

## 3:- Source Code:

#include <iostream> using namespace std;

void insertionSort(int arr[], int n)

{

int i, key, j;

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

{

key = arr[i]; j = i - 1;

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

{

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

}

arr[j + 1] = key;

}

}

void printArray(int arr[], int n)

{

int i;

for (i = 0; i < n; i++) cout << arr[i] << " ";

cout << endl;

}

int main()

{

int n;

cout << "Enter the size of the array: "; cin >> n; int arr[n];

cout << "Enter the elements of the array: "; for (int i = 0; i < n; i++)

{

cin >> arr[i];

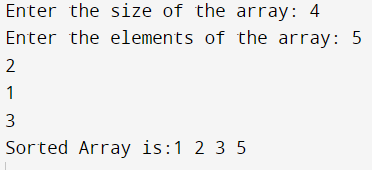
}

cout<<"Sorted Array is:";

insertionSort(arr, n); printArray(arr, n); return 0;

}

## Result/Output



1. **Learning Outcomes**
   * Insertion Sort is one of the simplest algorithm with simple implementation.
   * Basically, Insertion sort is efficient for small data values.
   * Insertion sort is adaptive in nature, i.e. it is appropriate for data sets which are already partially sorted.