Recursive Bubble Sort

- In recursive approach we will just select the range recursively.

// Steps

1. call recursive function: bubble(arr, n)
2. inside function, repeatedly swap adjacent elements, if(arr[i] > arr[i+1])
   - max element of unsorted array reaches to the end of the unsorted array after each call.
3. each time after step2, call the recursion again, decreasing range by 1.
4. Base Case: if(n == 1) return;

// Pseudocode:

```
bubble_sort(arr[], int n){
    if (n == 1)
        return;
    for (i = 0; i <= n-2){
        if(arr[i] > arr[i+1])
            swap();
    }
}
bubble_sort(arr, n-1)
}
```

time complexity: O(n²)

space complexity: O(n) (stack space)

#Recursive Insertion Sort

- Call the recursive function with the given array, size and index of selected array.
- Inside function, take the element at i, then place it in its corresponding position. (using swapping)
- after that shift the elements accordingly.
- finally, call the recursive function by increasing i by 1.
- Base case: if (n == 1) return.

// Pseudocode

```
insertion_Sort(arr[], n, i){
  if(i == n){ return;}
  j = i;
  while(j > 0 && arr[j-1] > arr[j]){
        swap();
        j--;
  }
insertion_Sort(arr, n, i+1); //recursive call
}
```

T.C = O(n²)

S.C = O(1)