Handling different list cases.

Python Code:-

```
def handle_list(arr):
if not arr:
    return arr
return sorted(arr)
Print(handle_list([]))
Print(handle_list([1]))
Print(handle_list([7,7,7,7]))
Print(handle_list([-5,-1,-3,-2,-4]))
```

Output:-

```
[]
[1]
[7,7,7,7]
[-5,-4,-3,-2,-1]
```

Selection Sort

Python Code:-

```
def selection_sort (arr):
n= len (arr)
for i in range (i+1, n):
if arr [j] < arr [min-index]:
 min_ index = j
arr[i], arr[min-index] = arr [min-index], arr[i]
return arr
Print (selection_sort([5, 2, 9, 1, 5, 6]))
Print (selection_sort([10, 8, 6, 4, 2]))
```

Output:-

[1, 2, 5, 5, 6, 9]

[2, 4, 6, 8, 10]

Bubble Sort:-

Python Code:-

```
def bubble_sort(arr):
n = len(arr)
for i in range(n):
 swapped = False
for j in range(0, n-i-1):
  if arr[j] > arr[j+1]:
   arr[j], arr[j+1] = arr[j+1], arr[j]
    swapped = True
 if not swapped:
  break
return arr
```

```
Print(bubble_sort([64, 25, 12, 22, 11]))
Print(bubble_sort([3, 5, 2, 1, 4]))
Print(bubble_sort([29, 10, 14, 37, 13]))
```

Output:-

[11, 12, 22, 25, 64]

[1, 2, 3, 4, 5]

[10, 13, 14, 29, 37]

Insertion Sort with duplicate handling

Python Code:-

```
def insertion_sort (arr):
for i in range (1, len(arr)):
  key = arr[i]
  j = i-1
while j >=0 and key < arr[j]:
  arr[j+1] = arr[j]
    j -=1
arr[j+1] = key
return arr
Print (insertion_sort ([3,1,4,1,5,9,2,6,5,3]))
Print (insertion_sort([5,5,5,5,5,5]))
```

Output:-

[1, 1, 2, 3, 3, 4, 5, 5, 6, 9]

[5, 5, 5, 5, 5, 5]

Kth missing Positive Integer

Python Code:-

```
def find_kth(arr,k):
missing = []
current = 1
for num in arr:
  while current < num:
    missing.append(current)
    current += 1
  current += 1
while len(missing)<k:
    missing.append(current)
    current += 1
return missing[k-1]
Print(find_kth([2,3,4,7,11],5))
Print(find_kth([1,2,3,4],2))
```

Output:-

9

6

Peak Element in O(log n)

Python Code:-

```
def find-peak (nums):
 left, right = 0, len (nums)-1
while left < right:
  mid = (left +right )//2
if nums [mid ] > nums[mid+1]:
 right = mid
else :
  left = mid+1
return left
Print (find-peak([1,2,3,1]))
Print (find-peak([1,2,1,3,5,6,4]))
```

Output:-

```
2
5
```

First Occurrence of substring

Python Code:-

```
def find-substring(hay-stack, needle):
return haystack.find(needle)

hay-stack= "sad but sad", "leetcode"
needle = "sad", "leeto"
Print(find-substring(hay_stack,needle).
```

Output:-

(0, -1)

strings that are substring of Another.

Python Code :-

```
def find-substring(words):
result = []
for i in range(len(words)):
    for j in range(len(words)):
        if i != j and words[i] in words[j]:
        result.append(words[i])
        break
return result.
Print(find-substring(["mass","as","hero","superhero"]))
Print(find-substring(["leetcode","et","code"]))
```

Output :-

["as","hero"]

["et","code"]