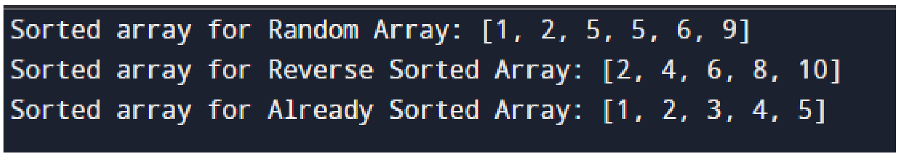
143.Describe the Selection Sort algorithm's process of sorting an array. Selection Sort works by dividing the array into a sorted and an unsorted region. Initially, the sorted region is empty, and the unsorted region contains all elements. The algorithm repeatedly selects the smallest element from the unsorted region and swaps it with the leftmost unsorted element, then moves the boundary of the sorted region one element to the right. Explain why Selection Sort is simple to understand and implement but is inefficient for large datasets. Provide examples to illustrate step-by-step how Selection Sort rearranges the elements into ascending order, ensuring clarity in your explanation of the algorithm's mechanics and effectiveness.
Sorting a Random Array:
Input: [5, 2, 9, 1, 5, 6]
Output: [1, 2, 5, 5, 6, 9]
Sorting a Reverse Sorted Array:
Input: [10, 8, 6, 4, 2]
Output: [2, 4, 6, 8, 10]
Sorting an Already Sorted Array:
Input: [1, 2, 3, 4, 5]
Output: [1, 2, 3, 4, 5]

AIM: To solve the selection sort

PROGRAM:

```
def selectionSort(nums):
    n = len(nums)
    for i in range(n):
        min_idx = i
        for j in range(i+1, n):
            if nums[j] < nums[min_idx]:
                min_idx = j
                nums[i], nums[min_idx] = nums[min_idx], nums[i]

# Examples to test the selectionSort function
if __name__ == "__main__":
    arr1 = [5, 2, 9, 1, 5, 6]
    selectionSort(arr1)
    print("Sorted array for Random Array:", arr1)
    arr2 = [10, 8, 6, 4, 2]
    selectionSort(arr2)
    print("Sorted array for Reverse Sorted Array:", arr2)
    arr3 = [1, 2, 3, 4, 5]
    selectionSort(arr3)
    print("Sorted array for Already Sorted Array:", arr3)
```

OUTPUT: 

```
Sorted array for Random Array: [1, 2, 5, 5, 6, 9]
Sorted array for Reverse Sorted Array: [2, 4, 6, 8, 10]
Sorted array for Already Sorted Array: [1, 2, 3, 4, 5]
```

TIME COMPLEXITY: O(n^2)