1.

```
def perform_on_list(lst):
    return sorted(lst)
print(perform_on_list([7,7,7,7]))
```

2.

```
def selection_sort(arr):
    for i in range(len(arr)):
        min_idx = i
        for j in range(i + 1, len(arr)):
            if arr[min_idx] > arr[j]:
                min_idx = j
                arr[i], arr[min_idx] = arr[min_idx], arr[i]
                return arr
                print(selection_sort([5, 2, 9, 1, 5, 6]))
```

3.

```
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]))
```

4

```
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]))
```

5

```
def find_kth_missing_positive(arr, k):
    missing = []
    current = 1
    i = 0
    while len(missing) < k:
        if i < len(arr) and arr[i] == current:
            i += 1
            else:
                missing.append(current)
                current += 1
                return missing[-1]
                print(find_kth_missing_positive([2, 3, 4, 7, 11], 5)
```

6.

```
def str_str(haystack, needle):
    return haystack.find(needle)
    print(str_str("sadbutsad", "sad"))
```

7.

```python
def find_substrings(words):
    result = []
    for i, word in enumerate(words):
        for j, other in enumerate(words):
            if i != j and word in other:
                result.append(word)
            break
            return result
            print(find_substrings(["mass", "as", "hero", "superhero"]))
```