Q1. Reverse a String

```
def reverse_string(s: str) -> str:
    return s[::-1]

print(reverse_string("hello"))  # 'olleh'
```

Complexity: Time: O(n), Space: O(n)

Q2. Check if a String is Palindrome

```
def is_palindrome(s: str) -> bool:
    return s == s[::-1]

print(is_palindrome("madam"))  # True
```

Complexity: Time: O(n), Space: O(n)

Q3. Find Maximum Element in Array

```
def find_max(arr):
    return max(arr)

print(find_max([1,2,3,4,5]))  # 5
```

Complexity: Time: O(n), Space: O(1)

Q4. Find Minimum Element in Array

```
def find_min(arr):
    return min(arr)

print(find_min([1,2,3,4,5]))  # 1
```

Complexity: Time: O(n), Space: O(1)

Q5. Find Second Largest Element

```
def second_largest(arr):
    arr = list(set(arr))
    arr.sort()
    return arr[-2]

print(second_largest([10,20,4,45,99]))  # 45
```

Complexity: Time: O(n log n), Space: O(n)

Q6. Sum of Elements in Array

```
def sum_array(arr):
    return sum(arr)

print(sum_array([1,2,3,4]))  # 10
```

Complexity: Time: O(n), Space: O(1)

Q7. Binary Search

```
def binary_search(arr, target):
    l, r = 0, len(arr)-1
    while l <= r:
        mid = (l+r)//2
```

```
        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            l = mid+1
        else:
            r = mid-1
    return -1

print(binary_search([1,2,3,4,5], 4))  # 3
```

Complexity: Time: O(log n), Space: O(1)

Q8. Linear Search

```
def linear_search(arr, target):
    for i, val in enumerate(arr):
        if val == target:
            return i
    return -1

print(linear_search([1,2,3], 2))  # 1
```

Complexity: Time: O(n), Space: O(1)

Q9. Check if Array is Sorted

```
def is_sorted(arr):
    return all(arr[i] <= arr[i+1] for i in range(len(arr)-1))

print(is_sorted([1,2,3,4]))  # True
```

Complexity: Time: O(n), Space: O(1)

Q10. Find Factorial

```
def factorial(n):
    return 1 if n==0 else n*factorial(n-1)

print(factorial(5))  # 120
```

Complexity: Time: O(n), Space: O(n)