1. Reverse a String

python

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

# Example usage
print(reverse_string("hello"))  # Output: "olleh"
```

2. Count the Number of Vowels in a String

python

```
def count_vowels(s):
    vowels = "aeiouAEIOU"
    return sum(1 for char in s if char in vowels)

# Example usage
print(count_vowels("hello world"))  # Output: 3
```

3. Check if a Given String is a Palindrome

python

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

# Example usage
print(is_palindrome("radar"))  # Output: True
print(is_palindrome("hello"))  # Output: False
```

4. Check if Two Given Strings are Anagrams

python

```
def are_anagrams(s1, s2):
    return sorted(s1) == sorted(s2)

# Example usage
print(are_anagrams("listen", "silent"))  # Output: True
print(are_anagrams("hello", "world"))    # Output: False
```

5. Find All Occurrences of a Given Substring

python

```
def find_substring_occurrences(s, substring):
    return [i for i in range(len(s)) if s.startswith(substring, i)]

# Example usage
print(find_substring_occurrences("banana", "an"))  # Output: [1, 3]
```

6. Basic String Compression

python

```
def compress_string(s):
    compressed = []
    count = 1
    for i in range(1, len(s)):
        if s[i] == s[i - 1]:
            count += 1
        else:
            compressed.append(s[i - 1] + str(count))
            count = 1
    compressed.append(s[-1] + str(count))
    return ''.join(compressed)

# Example usage
print(compress_string("aaabbbbccda"))  # Output: "a3b4c2d1a1"
```

7. Determine if a String Has All Unique Characters

python

```
def has_unique_characters(s):
    return len(s) == len(set(s))

# Example usage
print(has_unique_characters("abcdef"))  # Output: True
print(has_unique_characters("hello"))   # Output: False
```

8. Convert a String to Uppercase or Lowercase

python

```
def convert_case(s, to_upper=True):
    return s.upper() if to_upper else s.lower()

# Example usage
print(convert_case("Hello World"))  # Output: "HELLO WORLD"
print(convert_case("Hello World", to_upper=False))  # Output: "hello world"
```

9. Count the Number of Words in a String

python

```
def count_words(s):
    return len(s.split())

# Example usage
print(count_words("Hello world, this is a test"))  # Output: 6
```

10. Concatenate Two Strings Without Using +

python

```
def concatenate_strings(s1, s2):
    return ''.join([s1, s2])
```

```
# Example usage
print(concatenate_strings("Hello", "World"))  # Output: "HelloWorld"
```

11. Remove All Occurrences of a Specific Element from a List

python

```
def remove_element(lst, element):
    return [x for x in lst if x != element]

# Example usage
print(remove_element([1, 2, 3, 4, 2, 2], 2))  # Output: [1, 3, 4]
```

12. Find the Second Largest Number in a List

python

```
def find_second_largest(lst):
    unique_lst = list(set(lst))
    unique_lst.sort()
    return unique_lst[-2] if len(unique_lst) > 1 else None

# Example usage
print(find_second_largest([10, 20, 4, 45, 99]))  # Output: 45
```

13. Count Occurrences of Each Element in a List

python

```
def count_occurrences(lst):
    return {x: lst.count(x) for x in set(lst)}

# Example usage
print(count_occurrences([1, 2, 2, 3, 4, 4, 4]))  # Output: {1: 1, 2: 2, 3: 1, 4: 3}
```

14. Reverse a List In-Place

python

```
def reverse_list(lst):
    start, end = 0, len(lst) - 1
    while start < end:
        lst[start], lst[end] = lst[end], lst[start]
        start += 1
        end -= 1
    return lst

# Example usage
print(reverse_list([1, 2, 3, 4, 5]))  # Output: [5, 4, 3, 2, 1]
```

15. Remove Duplicates from a List While Preserving Order

python

```python
def remove_duplicates(lst):
    seen = set()
    return [x for x in lst if not (x in seen or seen.add(x))]

# Example usage
print(remove_duplicates([1, 2, 2, 3, 4, 1]))  # Output: [1, 2, 3, 4]
```

16. Check if a List is Sorted

```python
def is_sorted(lst):
    return lst == sorted(lst) or lst == sorted(lst, reverse=True)

# Example usage
print(is_sorted([1, 2, 3, 4]))  # Output: True
print(is_sorted([4, 3, 2, 1]))  # Output: True
print(is_sorted([1, 3, 2, 4]))  # Output: False
```

17. Merge Two Sorted Lists into a Single Sorted List

```python
def merge_sorted_lists(lst1, lst2):
    sorted_list = []
    i, j = 0, 0

    while i < len(lst1) and j < len(lst2):
        if lst1[i] < lst2[j]:
            sorted_list.append(lst1[i])
            i += 1
        else:
            sorted_list.append(lst2[j])
            j += 1

    # Append remaining elements
    sorted_list.extend(lst1[i:])
    sorted_list.extend(lst2[j:])

    return sorted_list

# Example usage
print(merge_sorted_lists([1, 3, 5], [2, 4, 6]))  # Output: [1, 2, 3, 4, 5, 6]
```

18. Find the Intersection of Two Given Lists

```python
def intersection_of_lists(lst1, lst2):
    return list(set(lst1) & set(lst2))

# Example usage
print(intersection_of_lists([1, 2, 3, 4], [3, 4, 5, 6]))  # Output: [3, 4]
```

19. Find the Union of Two Lists Without Duplicates

python

```
def union_of_lists(lst1, lst2):
    return list(set(lst1) | set(lst2))

# Example usage
print(union_of_lists([1, 2, 3, 4], [3, 4, 5, 6]))  # Output: [1, 2, 3, 4, 5, 6]
```

20. Shuffle a Given List Randomly Without Using Built-In Shuffle Functions

python

```
import random

def shuffle_list(lst):
    lst_copy = lst[:]
    n = len(lst_copy)
    for i in range(n):
        j = random.randint(0, n - 1)
        lst_copy[i], lst_copy[j] = lst_copy[j], lst_copy[i]
    return lst_copy

# Example usage
print(shuffle_list([1, 2, 3, 4, 5]))  # Output: Randomly shuffled list
```

21. Find Common Elements in Two Tuples

python

```
def common_elements_in_tuples(tpl1, tpl2):
    return tuple(set(tpl1) & set(tpl2))

# Example usage
print(common_elements_in_tuples((1, 2, 3), (2, 3, 4)))  # Output: (2, 3)
```

22. Prompt User to Enter Two Sets of Integers and Print Their Intersection

python

```
def get_set_intersection():
    set1 = set(map(int, input("Enter the first set of integers separated by commas: ").split(',')))
    set2 = set(map(int, input("Enter the second set of integers separated by commas: ").split(',')))
    intersection = set1 & set2
    print("Intersection:", intersection)

# Call the function
get_set_intersection()
```

23. Concatenate Two Tuples

python

```
def concatenate_tuples(tpl1, tpl2):
```

```
    return tpl1 + tpl2

# Example usage
print(concatenate_tuples((1, 2, 3), (4, 5, 6)))  # Output: (1, 2, 3, 4, 5, 6)
```

24. Print Elements Present in the First Set But Not in the Second Set

```
python

def difference_of_sets():
    set1 = set(input("Enter the first set of strings separated by commas: ").split(','))
    set2 = set(input("Enter the second set of strings separated by commas: ").split(','))
    difference = set1 - set2
    print("Elements in the first set but not in the second set:", difference)

# Call the function
difference_of_sets()
```

25. Return Elements from Tuple Within Specified Range of Indices

```
python

def elements_in_range(tpl, start, end):
    return tpl[start:end]

# Example usage
print(elements_in_range((1, 2, 3, 4, 5, 6, 7), 2, 5))  # Output: (3, 4, 5)
```

26. Print Union of Two Sets of Characters

```
python

def union_of_character_sets():
    set1 = set(input("Enter the first set of characters separated by commas: ").split(','))
    set2 = set(input("Enter the second set of characters separated by commas: ").split(','))
    union = set1 | set2
    print("Union of the two sets:", union)

# Call the function
union_of_character_sets()
```

27. Return Maximum and Minimum Values from a Tuple

```
python

def max_min_from_tuple(tpl):
    max_val, min_val = max(tpl), min(tpl)
    return max_val, min_val

# Example usage
print(max_min_from_tuple((3, 1, 4, 1, 5, 9, 2)))  # Output: (9, 1)
```

28. Print Union, Intersection, and Difference of Two Sets

python

```
def set_operations():
    set1 = set([1, 2, 3, 4, 5])
    set2 = set([4, 5, 6, 7, 8])

    union = set1 | set2
    intersection = set1 & set2
    difference = set1 - set2

    print("Union:", union)
    print("Intersection:", intersection)
    print("Difference:", difference)

# Call the function
set_operations()
```

29. Count Occurrences of an Element in a Tuple

python

```
def count_occurrences(tpl, element):
    return tpl.count(element)

# Example usage
print(count_occurrences((1, 2, 3, 1, 4, 1), 1))  # Output: 3
```

30. Print Symmetric Difference of Two Sets of Strings

python

```
def symmetric_difference_of_sets():
    set1 = set(input("Enter the first set of strings separated by commas: ").split(','))
    set2 = set(input("Enter the second set of strings separated by commas: ").split(','))
    symmetric_difference = set1 ^ set2
    print("Symmetric difference:", symmetric_difference)

# Call the function
symmetric_difference_of_sets()
```

31. Return Dictionary of Word Frequencies

python

```
def word_frequencies(words):
    freq_dict = {}
    for word in words:
        freq_dict[word] = freq_dict.get(word, 0) + 1
    return freq_dict

# Example usage
print(word_frequencies(['apple', 'banana', 'apple', 'orange', 'banana', 'apple']))  # Output: {'apple': 3, 'banana': 2, 'orange': 1}
```

32. Merge Two Dictionaries and Add Values for Common Keys

python

```
def merge_dicts(dict1, dict2):
    merged_dict = dict1.copy()
    for key, value in dict2.items():
        if key in merged_dict:
            merged_dict[key] += value
        else:
            merged_dict[key] = value
    return merged_dict

# Example usage
print(merge_dicts({'a': 1, 'b': 2}, {'b': 3, 'c': 4}))  # Output: {'a': 1, 'b': 5, 'c': 4}
```

33. Access Value in a Nested Dictionary Using a List of Keys

python

```
def get_nested_value(dictionary, keys):
    for key in keys:
        if key in dictionary:
            dictionary = dictionary[key]
        else:
            return None
    return dictionary

# Example usage
nested_dict = {'a': {'b': {'c': 1}}}
print(get_nested_value(nested_dict, ['a', 'b', 'c']))  # Output: 1
print(get_nested_value(nested_dict, ['a', 'b', 'd']))  # Output: None
```

34. Return Sorted Dictionary Based on Values

python

```
def sort_dict_by_values(d, reverse=False):
    return dict(sorted(d.items(), key=lambda item: item[1], reverse=reverse))

# Example usage
print(sort_dict_by_values({'a': 3, 'b': 1, 'c': 2}))  # Output: {'b': 1, 'c': 2, 'a': 3}
```

35. Invert a Dictionary Swapping Keys and Values

python

```
def invert_dict(d):
    inverted_dict = {}
    for key, value in d.items():
        if value in inverted_dict:
            inverted_dict[value].append(key)
        else:
            inverted_dict[value] = [key]
    return inverted_dict
```

```
# Example usage
print(invert_dict({’a’: 1, ’b’: 2, ’c’: 1}))  # Output: {1: [’a’, ’c’], 2: [’b’]}
```