Dictionary

Program 1: constructing set and dict comprehensions as well.

```
print({x for x in 'abcddeef' if x not in 'abc'})
print({x: x**2 for x in range(5)})
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Documents
{'f', 'd', 'e'}
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16}
```

Program 2: merge dictionary and square of no. in dict

```
dict1 = {'a': 1, 'b': 2}
dict2 = {'c': 3, 'd': 4}
merged_dict = {**dict1, **dict2}
print("Merged dictionary:", merged_dict)

n = 5
squares = {x: x**2 for x in range(1, n+1)}
print("Squares up to", n, ":", squares)
```

OUTPUT

```
❯ /bin/python /home/sridhar/Documents/7thsem/Pondic
Merged dictionary: {'a': 1, 'b': 2, 'c': 3, 'd': 4}
Squares up to 5 : {1: 1, 2: 4, 3: 9, 4: 16, 5: 25}
```

Program 3:

```
dict1 = {"a": 1, "b": 2}
dict2 = {"c": 3}
dict1.update(dict2)
print(dict1)

# If they have same keys:
dict1.update({"c": 4})
print(dict1)
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Docume
{'a': 1, 'b': 2, 'c': 3}
{'a': 1, 'b': 2, 'c': 4}
```

Lists

Program 1: Program to check if a string is a palindrome using list slicing

```
string = "radar"
if string == string[::-1]:
print("Palindrome")
else:
print("Not a palindrome")
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Documents
Palindrome
```

Program 2: progam for fibonacci sequence using list

```
n = 10
a, b = 0, 1
fibonacci_sequence = [a, b]
for _ in range(n - 2):
a, b = b, a + b
fibonacci_sequence.append(b)
print("Fibonacci sequence:", fibonacci_sequence)
```

OUTPUT:

```
Fibonacci sequence: [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

Program 3: program using list in-build functions

```
numbers = [1, 2, 3, 4, 2, 5, 2, 6, 2, 7]
count_of_2 = numbers.count(2)
print("Number of occurrences of 2:", count_of_2)

values = (-x for x in [1,2,3,4,5])
gen_to_list = list(values)
print(gen_to_list)
```

OUTPUT:

```
Number of occurrences of 2: 4
[-1, -2, -3, -4, -5]
```

Sets

Program 1: To add or pop values from a set

```
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
intersection = set1.intersection(set2)
print("Intersection of sets:", intersection)

set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
union = set1.union(set2)
print("Union of sets:", union)
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Documents/7tl
Intersection of sets: {4, 5}
Union of sets: {1, 2, 3, 4, 5, 6, 7, 8}
```

Program 2: Program to find the difference between two sets:

```
set1 = {1, 2, 3, 4, 5}

set2 = {4, 5, 6, 7, 8}
difference = set1.difference(set2)
print("Difference of sets (set1 - set2):", difference)
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Documents/7thsem/
Difference of sets (set1 - set2): {1, 2, 3}
```

Program 3:

```
set1 = {1, 2, 3}
set2 = {1, 2, 3, 4, 5}
is_subset = set1.issubset(set2)
print("Is set1 a subset of set2?", is_subset)
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Docume
Is set1 a subset of set2? True
```

Tuple

Program 1: Program to check if an element exists in a tuple

```
my_tuple = (10, 20, 30, 40, 50)
element = 30
if element in my_tuple:
    print(element, "exists in the tuple")
else:
    print(element, "does not exist in the tuple")
```

OUTPUT:

```
❯ /bin/python /home/sridh
30 exists in the tuple
```

Program 2: Program to find min and max using tuple

```
my_tuple = (10, 20, 30, 40, 50)
maximum = max(my_tuple)
minimum = min(my_tuple)
print("Maximum element:", maximum)
print("Minimum element:", minimum)
```

OUTPUT:

```
❯ /bin/python /home/sri
Maximum element: 50
Minimum element: 10
```

Program 3: Swaping numbers using tuple

```
a = 5
b = 10
print("Before swapping: a =", a, ", b =", b)
a, b = b, a
print("After swapping: a =", a, ", b =", b)
```

OUTPUT:

```
❯ /bin/python /home/sridhar/Docume
Before swapping: a = 5 , b = 10
After swapping: a = 10 , b = 5
```