Experiment 3 c)

Aim: Concatenation operation on Strings, List, Tuple, Set and Dictionary in Python.

Theory:

➢ **Concatenation:**

- Concatenation means **joining characters or items together end-to-end to create a new sequence**.
- Concatenation is done between the same data types only.
- The functions or operators used for concatenation are enlisted as:

S. No.	Built-in Data Types	Method
1	String Concatenation	'+' , '%' , ',' operator and join()
2	List Concatenation	'+' , '*' operator, append() and extend()
3	Tuple Concatenation	'+' operator and sum()
4	Set Concatenation	union()
5	Dictionary Concatenation	update()

```
#String Concatenation

var1 = "Hello "
var2 = "World"

# + Operator is used to combine strings
var3 = var1 + var2
print(var3)


# join() method is used to combine the strings
print("".join([var1, var2]))

# join() method combine the string with a separator
var4 = " ".join([var1, var2])
print(var4)

# % Operator is used here to combine the string
print("% s % s" % (var1, var2))

# , to combine data types with a single whitespace.
print(var1, var2)
```

```
#List Concatenation
a = [1, 4, 5, 6, 5]
b = [3, 5, 7, 2, 5]

for i in b:
    a.append(i)
# Printing concatenated list
print ("Concatenated list:", a)

c = [1, 4, 5, 6, 5]
d = [3, 5, 7, 2, 5]
# using + operator to concat
e = c + d
print ("Concatenated list using + : ", e)

# using * operator to concat
f = [*c, *d]
print ("Concatenated list using * operator : ", f)

# using list.extend() to concat
c.extend(d)
print ("Concatenated list using list.extend(): ", c)
```

```
#Tuple Concatenation

# using + operator
test_tup1 = (1, 3, 5)
test_tup2 = (4, 6)

# using + operator
res = test_tup1 + test_tup2
print("The tuple after concatenation is : ", res)


# using sum()
res = sum((test_tup1, test_tup2), ())
print("The tuple after concatenation is : ",res)
```

```
# Set Concatenation

a = {"a", "b", "c"}
b = {1, 2, 3}

c = a.union(b)
print(c)
```

```
# Dictionary Concatenation

a = {'a': 10, 'b': 8, 'c':10}

b = {'d': 6, 'e': 4, 'k' : "Hello"}

a.update(b)

print(a)
```

Output

String	List	Tuple	Set	Dictionary
Hello World Hello World Hello World Hello World Hello World	Concatenated list: [1, 4, 5, 6, 5, 3, 5, 7, 2, 5] Concatenated list using + : [1, 4, 5, 6, 5, 3, 5, 7, 2, 5] Concatenated list using * operator : [1, 4, 5, 6, 5, 3, 5, 7, 2, 5] Concatenated list using list.extend(): [1, 4, 5, 6, 5, 3, 5, 7, 2, 5]	The tuple after concatenation is : (1, 3, 5, 4, 6) The tuple after concatenation is : (1, 3, 5, 4, 6)	{1, 'b', 2, 'a', 3, 'c'}	{'a': 10, 'b': 8, 'c': 10, 'd': 6, 'e': 4, 'k': 'Hello'}

Conclusion:

Hence implemented Concatenation operation for Strings, List, Tuple, Set and Dictionary.