Day 01

1. Create a list of 5 numbers. Print the first and last elements using indexing.

```
In [7]: numbers = [15,25,64,48,98]
        print("First element of the list is:",numbers[0])
        print("Last element of the list is:",numbers[4])
```

```
First element of the list is: 15
Last element of the list is: 98
```

2. Print the list in reverse using slicing.

```
In [8]: number = [15,25,64,48,98]
        print("Reverse of the list is :",numbers[::-1])
```

```
Reverse of the list is : [98, 48, 64, 25, 15]
```

3. Print the elements from index 1 to 3 (inclusive).

```
In [15]: numbers = [15,"55",46,28,445,698]
         print(numbers[1:4])
```

```
['55', 46, 28]
```

4. Find the sum of all elements in the list using a for loop

```
In [25]: number = [15,25,45,55,50]
         sumation = 0
         for summing in number:
             sumation += summing
         print("Sum of all elements in the list is:",sumation)
```

```
Sum of all elements in the list is: 190
```

5. Count how many even numbers are present in the list.

```
In [20]: numbers = [12,25,32,15,47,87,98]
         count = 0
         for i in numbers:
             if i % 2 == 0:
                 count += 1
         print("Even numbers in list are: ",count)
```

```
Even numbers in list are:  3
```

6. Print only the elements at even index positions.

```
In [31]: number = [15,24,36,25,78,98,58]
         y = 1
         for i in range(0,len(number),2):
             print("elements at even index (i) are:",number[i])
             if i % 2 == 0:
                 y += 2
         print("count of elements present at even possision:",y)
```

```
elements at even index 0 are: 15
elements at even index 2 are: 36
elements at even index 4 are: 78
elements at even index 6 are: 58
count of elements present at even possision: 4
```

7. Create a list of 5 names. Print each name in uppercase using a for loop.

```
In [42]: names = ["karan","mahesh","nagur","shiva","ramu"]
         for i in names:
             print("names in upper case:",i.upper())
```

```
names in upper case: KARAN
names in upper case: MAHESH
names in upper case: NAGUR
names in upper case: SHIVA
names in upper case: RAMU
```

8. Print the length of the list using len() function.

```
In [43]: names = ["karan","mahesh","nagur","shiva","ramu"]
         print(len(names))
```

```
5
```

9. Replace the 3rd element of the list with a new value

```
In [51]: names = ["karan","mahesh","nagur","shiva","ramu"]
         names[2] = "amar"
         print(names)
```

```
['karan', 'mahesh', 'amar', 'shiva', 'ramu']
```

Day 2 and 3

1. What is a list in Python? How does it differ from a tuple?

```
In [ ]: >> list in python are mutable and ordered collection of elements. list are writen using []
```

2. Explain the concept of mutability in the context of Python lists.

```
In [1]: # >>we can change the elements in the list so the list is mutable
          # Ex:
        num = [1,2,3,4,5,6,7]
        num[2] = "nam"
        num
```

```
Out[1]: [1, 2, 'nam', 4, 5, 6, 7]
```

3. How do you access elements in a list? What is negative indexing?

```
In [ ]: >> we can access elements in a list by using index number
            ex:
                list1 = [1,2,3,4,5,6]
                print(list1[1])          # it will return the element in the list present at index number 1
        >> negative indexing means it will start counting index from the last to first in list (it starts from -1)
```

4. Explain list slicing with an example. How can you reverse a list using slicing?

```
In [105]: list1 = [11,22,33,44,55,66]
          print(list1[1:5])
          print(list1[::-1])
```

```
[11, 22, 33, 44, 55, 66]
[66, 55, 44, 33, 22, 11]
```

5. Describe the purpose and behavior of the append() and insert() list methods. What are their differences?

```
In [53]: # >> append() used to add element to the end of list
         # >> insert() used to add element to list using index
         list01 = [1,3,4,5,6,7]
         list01.append(8)
         print("Using append: ",list01)
         list01.insert(1,2)    # syntax : list.insert(index,element)
         print("Using insert: ",list01)
```

```
Using append:  [1, 3, 4, 5, 6, 7, 8]
Using insert:  [1, 2, 3, 4, 5, 6, 7, 8]
```

6. How do the remove() and pop() list methods work? What value does pop() return?

```
In [62]: # >> remove() used to remove first occurance in the list
         # >> pop() used to remove the element from list using index
         list02 = [11,22,33,44,55,66,77,88,99]
         list02.remove(22)
         print("Using remove():",list02)
         list02.pop(4)
         print("Using pop(): ",list02)
```

```
Using remove():  [11, 33, 44, 55, 66, 77, 88, 99]
Using pop():  [11, 33, 44, 55, 77, 88, 99]
```

What happens if you try to remove an element that doesn't exist?

```
In [24]: # it will show error as item is not in list
         list02.remove(100)
         print(list02)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[24], line 1
----> 1 list02.remove(100)
      2 print(list02)

ValueError: list.remove(x): x not in list
```

7. What is the purpose of the index() and count() list methods?

```
In [51]: # >> index() returns the index of a item in a list
         # >> count() returns the number of time the item appears in list
         list01 = [1,2,11,11,22,3,4,5,6,4]
         print("Possition of item1 :",list01.index(11))
         print("Number of time item appears: ",list02.count(11))
```

```
Possition of item1  2
Number of time item appears:  2
```

8. Explain how the sort() method works. How can you sort a list in descending order? Does sort() return a new list?

```
In [75]: # >> sort() is used to sort the items in list to ascending or descending
         # >> sort() will not return a new list it will update existing list
         list02 = [25,26,65,98,12,3,58,45]
         list02.sort()
         print(list02)
```

```
[3, 12, 25, 26, 45, 58, 65, 98]
```

9. What does the reverse() method do to a list? Does it return a new list?

```
In [91]: # >> reverse() will return the list making it reverse
         # >> it will not return new list it will update existing one
         list1 = [1,22,33,44,55]
         list1.reverse()
         list1
```

```
Out[91]: [55, 44, 33, 22, 11]
```

10. What is a nested list in Python? How do you access elements within a nested list? Provide an example.

```
In [111]: # >> list inside list is called nested list
          # >>ex: [1,2,5,6,78,15,9,89,57]
          # >> example to access element within nested list
          list12 = [1,2,5,6,78,[12,6,89,57]]
          list12[3][2]
```

```
Out[111]: 89
```

11. Explain list comprehension with its basic syntax. What are the advantages of using list comprehension over traditional for loops for list creation?

```
In [ ]: Syntax for list comprehension:
        list_name = []
        list_name2 = [expression for item in iterable condition]
          where
            > expression is operation that we wants to do on item or list
            > for item in refer name as in for loop
            > iterable means list or range from where item to be taken
            > condition is condition for the result

        Advantages of using list comprehension:
            > easy to write code
            > don't need to write multiple line of code
            > fast in execution
```

12. What is the purpose of the list() constructor? Give an example of how it can be used

```
In [ ]: # list() constructor:
            > each character is iterated
            > it iterates each character and gives the output as comma seperated characters
            > Syntax: string_name = ""
                      new_string = list(string_name)
```

```
In [8]: #Ex:
        string_1 = "mahesh"
        new_string = list(string_1)
        print(new_string)
```

```
['m', 'a', 'h', 'e', 's', 'h']
```

13. What does the del keyword do in Python when used with lists? How can you delete elements by index or slice? How can you delete the entire list?

```
In [ ]: # Del() function in list:
            > it deletes the item from a list base on index
        # Ex: deleting element by index:
        syntax: list_name = []
                del list_name[index]
                print(list_name)
        # Deleing element by index:
        >syntax: list_name = []
                del list_name[start_index:end_index]
                print(list_name)
```

```
In [16]: #EX for del() by index:
         list_name = [1,2,3,4]
         del list_name[1]
         print(list_name)
         #EX for del() by slicing:
         del list_name[0:1]
         print(list_name)
```

```
[1, 3, 4]
[3, 4]
```

Day 4

1. What is a set in Python? How does it differ from a list and a tuple? What are its key properties?

```
In [ ]: >> set is the unordered collection of unique elements in python
        >> in list and tuple we may use duplicate elements  but in set it automatically removes the duplicate elements
        >> key properties are:
            > we cant use list inside the set
            > removes duplicates
            > we can acces elements from set using indexing or slicing
            > we can acces elements from set using for loop and in operator
```

2. What is a frozenset? How does it differ from a regular set? When might you use a frozenset?

```
In [ ]: >> frozenset is the immutable version of set in python
            >list is mutable we can add,remove,update the elements
            >frozenset is immutable once created cant be changed
        >> we can use frozenset when we want to fix or dont want the changes in set
```

3. Explain the purpose and usage of the split() method for strings. Provide an example

```
In [8]: # >> splits the string based on delimeter
        string = "split the string"
        string.split(" ")
```

```
Out[8]: ['split', 'the', 'string']
```

4. Explain the purpose and usage of the join() method for strings. Provide an example

```
In [111]: string = ["Hi","!","I","am","ramesh"]
          join_string = " ".join(string)
          print(join_string)
```

```
Hi ! I am ramesh
```

5.Create two sets, set1 with elements [1, 2, 3, 4, 5] and set2 with elements [4, 5, 6, 7, 8]. Convert these lists to sets.

```
In [121]: set1 = [1,2,3,4,5]
          set2 = [4,5,6,7,8]
          new_set1 = set(set1)
          new_set2 = set(set2)
          print(new_set1)
          print(new_set2)
          print("Type of new_set1",type(new_set1))
          print("Type of new_set2",type(new_set2))
```

```
{1, 2, 3, 4, 5}
{4, 5, 6, 7, 8}
Type of new_set1 <class 'set'>
Type of new_set2 <class 'set'>
```

6. Find and print the common elements between set1 and set2 created in the previous question.

```
In [125]: set1 = [1,2,3,4,5]
          set2 = [4,5,6,7,8]
          new_set1 = set(set1)
          new_set2 = set(set2)
          new_set1.intersection(new_set2)
```

```
Out[125]: {4, 5}
```

7. Write a program that takes a sentence as input, splits it into words, and then prints the unique words in the sentence

```
In [137]: string = "my hometown is chikodi chikodi is in belagum karnataka chikodi is very beautiful village"
          string1 = string.split(" ")
          print("Unique words in string are:",set(string1))
```

```
Unique words in string are: {'is', 'belagum', 'hometown', 'in', 'my', 'chikodi', 'village', 'very', 'beautiful', 'karnataka'}
```

8. You have a list of words: ["Hello", "World", "Python"]. Use the join() method to create a single string with these words separated by a hyphen "-".

```
In [39]: list001 = ["Hello","World","Python"]
         string = "-".join(list001)
         print(string)
```

```
Hello-World-Python
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```