Day 8:

Lists in Python:

- Ordered, mutable collections of items can hold different data types..
- defined using square brackets [] and elements are separated by commas.

1. creating a list in python:

Empty list = []

numbers = [1, 2, 3, 4, 5]

fruits = ["apple", "banana", "cherry"].

mixed_list = [10, "hello", 3.5, True].

print (fruits)

output: ['apple', 'banana', 'cherry'].

2. accessing list elements:

You can access list elements using indexing and negative indexing:

Example:

fruits = ["apple", "banana", "cherry"]

Positive indexing:

print (fruits [0]) : apple.

collections of items

...

brackets [] and

commas.

:

", "cherry"].

5, True].

']

uring index

"cherry"]

negative indexing :

print (fruits [-1]) : cherry.

3. Adding elements into a list :

- append () → adds an item at the end.
- insert (index, element) → adds at specific index
- extend (iterable) → adds multiple items.

4. updating elements in a list :

lists are mutable, - so you can modify elements after creation.

fruits = list = ["apple", 'banana', 'orange']

fruits - list . append ('grape')

fruits - list . insert (1, 'kiwi')

fruits - list . extend (['mango', 'peach'])

fruits - list [2] = 'strawberry'

5. Removing elements from a list :

1. remove (value)
2. ~~index~~ pop (index)
3. del list (index)
4. clear () → Empties the list

6. Iterating lists:

You can loop through a list using a for loop

```
ex. fruits = ["apple", "banana", "orange"]

    for fruits in fruits:
        print (fruit)
```

Output:

apple
banana
cherry.

7. Nested lists in Python:

- list inside another list.

```
nested_list = [[1, 2, 3], ["apple", "banana"], [True, False]]
e.g.
    print (nested_list[1][0])

# output : apple

    Print (nested_list[2][1])

# output : false.
```

Accessing an element inside a nested list.