Practice Problems
List Concepts
List Operation

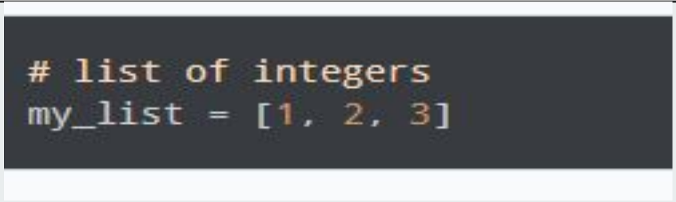
List Concept :

Python lists are one of the most versatile data types that allow us to work with multiple elements at once

- **Create Python Lists**
- **Accessing the List Elements**
- **Type Error and Index Error**
- **Slicing in List**
- **ADD, UPDATE and DELETE Operation on Lists**

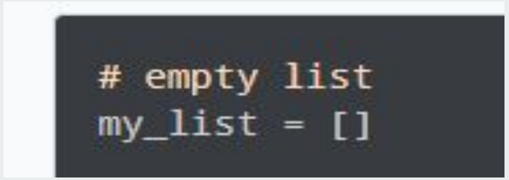
Create Python List :

Example of integer list:



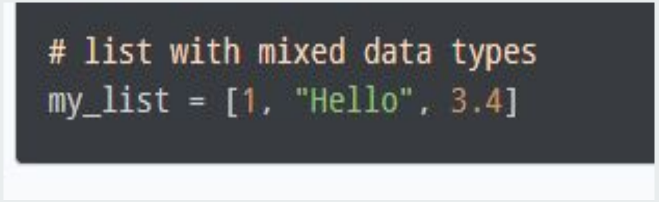
```
# list of integers
my_list = [1, 2, 3]
```

Empty List :



```
# empty list
my_list = []
```

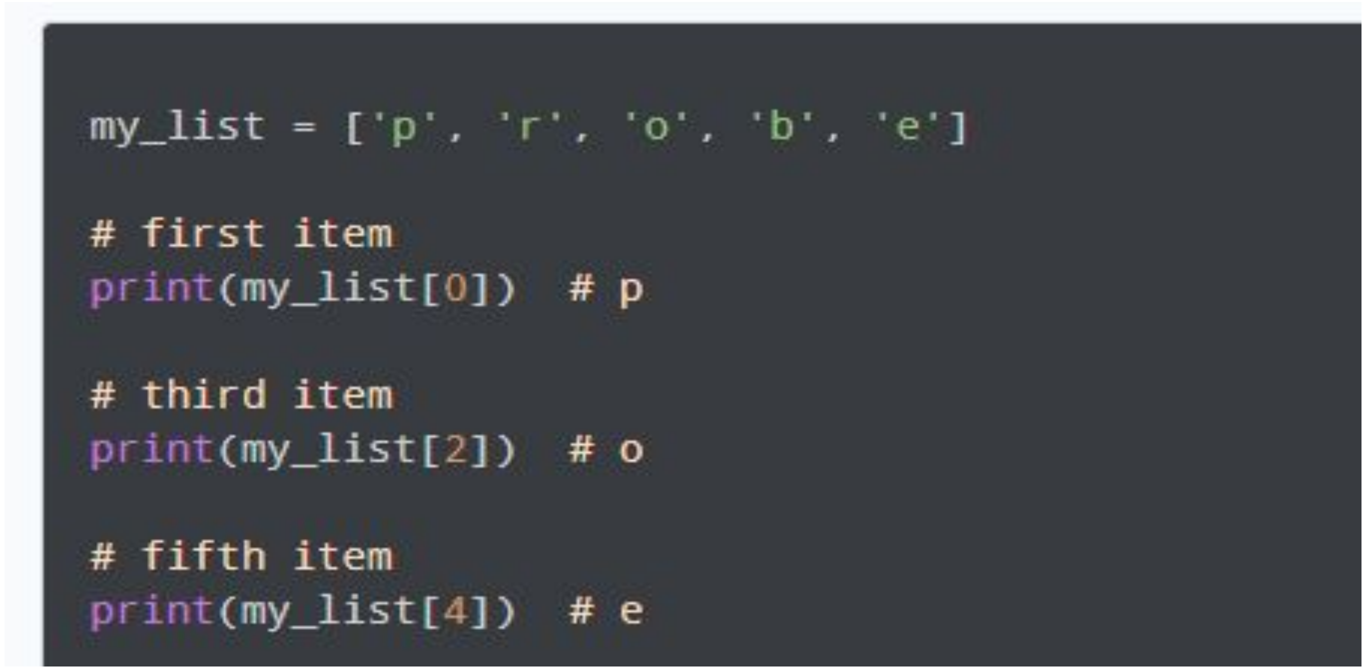
List with different data types:



```
# list with mixed data types
my_list = [1, "Hello", 3.4]
```

Accessing the List Elements

Use the index operator [ ] to access an item in a list.



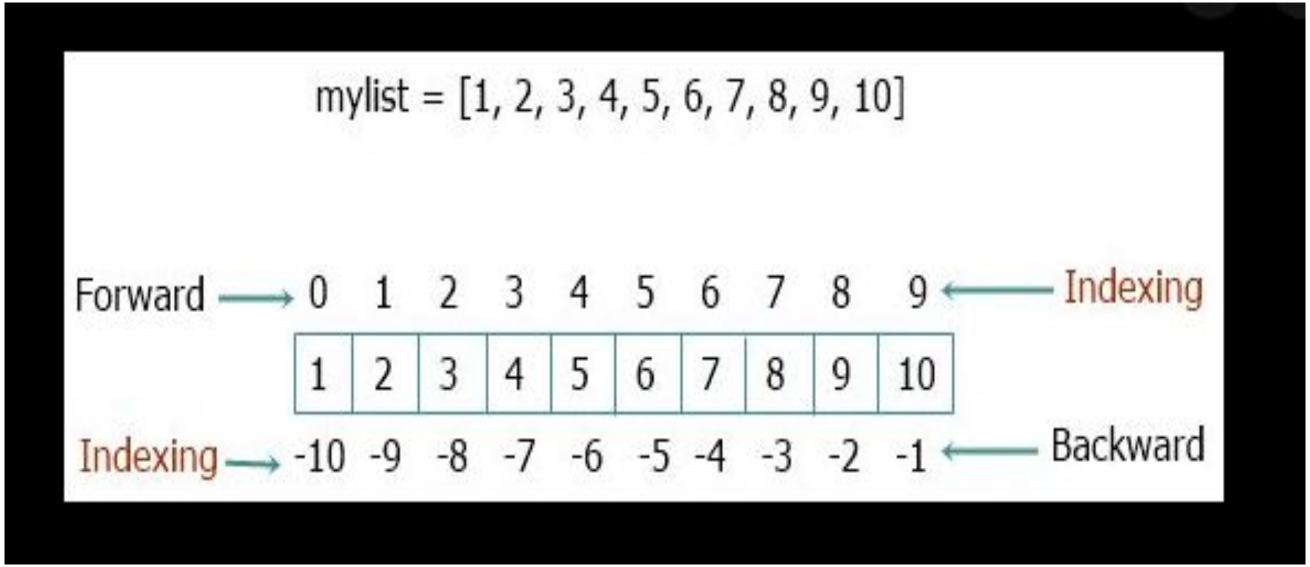
```
my_list = ['p', 'r', 'o', 'b', 'e']

# first item
print(my_list[0])  # p

# third item
print(my_list[2])  # o

# fifth item
print(my_list[4])  # e
```

Type Error and Index Error

Trying to access indexes other than these will raise an `IndexError.`

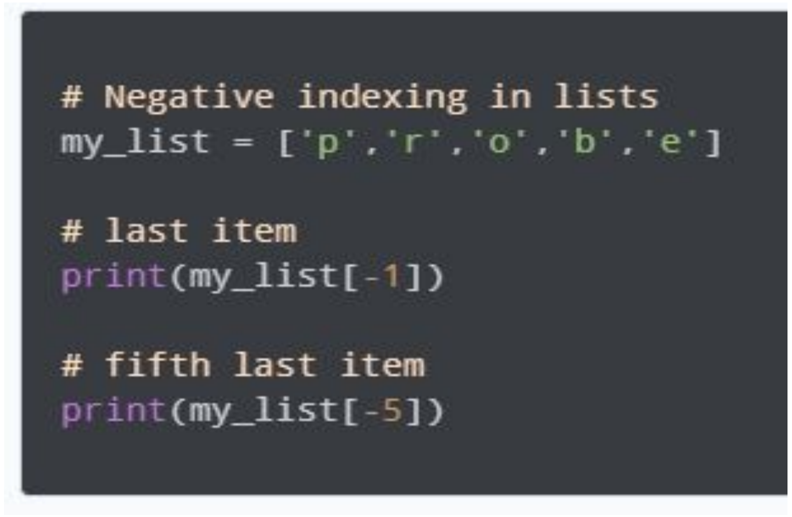
The index must be an integer. We can't use float or other types, this will result in `TypeError`

List Indexing



Negative indexing

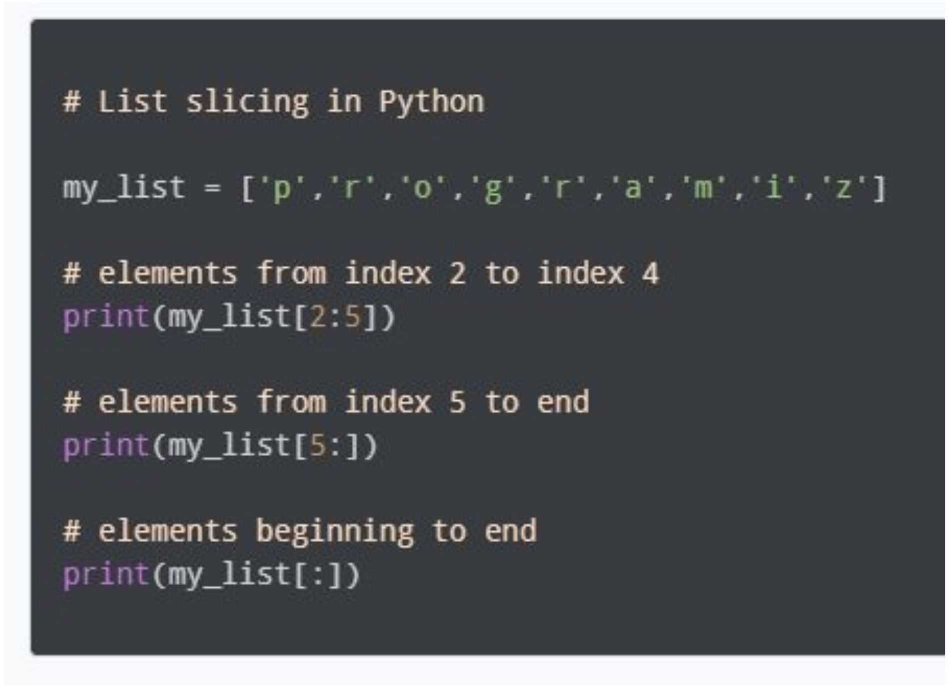
Python allows negative indexing for its sequences. The index of -1 refers to the last item, -2 to the second last item and so on



```
# Negative indexing in lists
my_list = ['p','r','o','b','e']

# last item
print(my_list[-1])

# fifth last item
print(my_list[-5])
```

Slicing in List

We can access a range of items in a list by using the slicing operator



```
# List slicing in Python

my_list = ['p','r','o','g','r','a','m','i','z']

# elements from index 2 to index 4
print(my_list[2:5])

# elements from index 5 to end
print(my_list[5:])

# elements beginning to end
print(my_list[:])
```

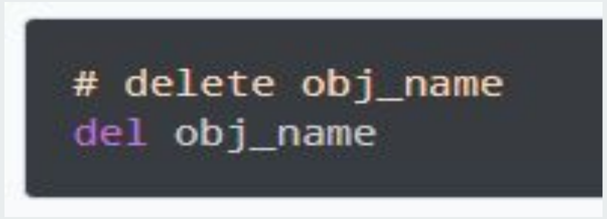
Add , Update and Delete

Lists are mutable, meaning their elements can be changed

- To add one element in list use **append()**
- To add more than one element in list use **extend()**
- To delete use **remove() or pop()**
- To Update use the assignment operator = to change an item or a range of items.

Python del Statement

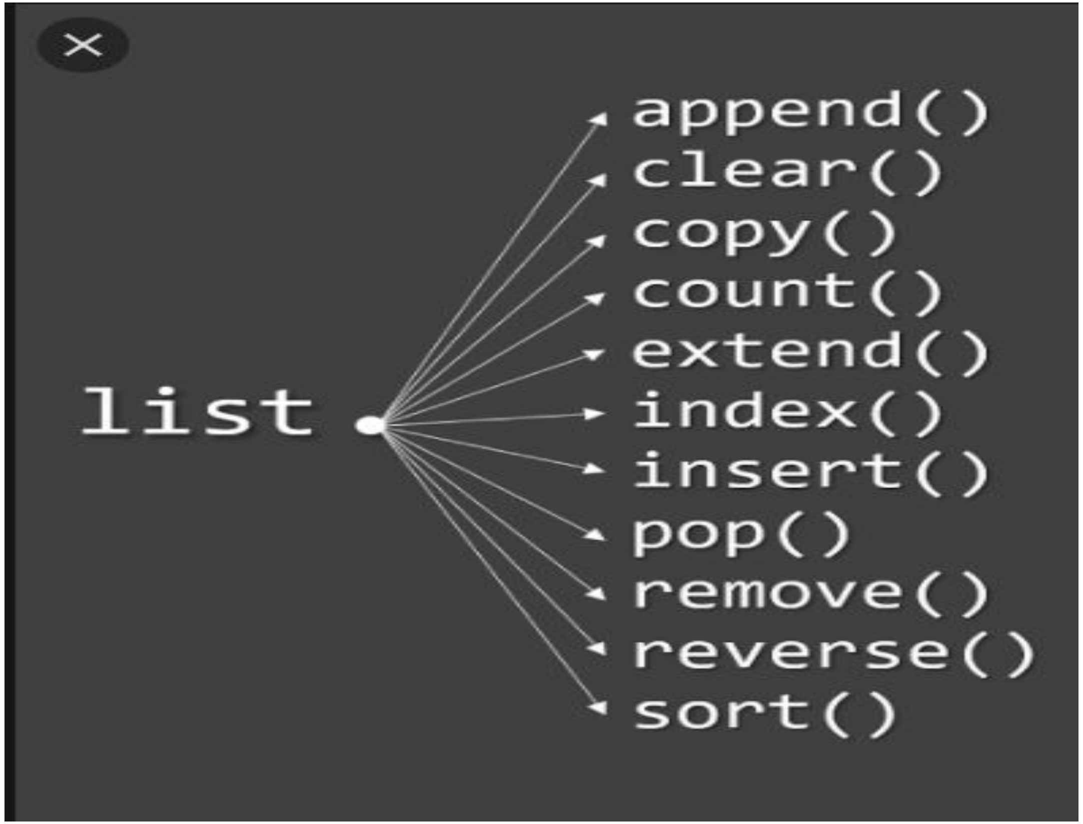
The Python **`del`** keyword is used to delete objects. Its syntax is:



```
# delete obj_name
del obj_name
```

Everything is in Python treated as an object, including variable, function, list, tuple, dictionary, set, etc. Every object belongs to its class. For example - **An integer variable belongs to integer class**.

List Methods


list
append()
clear()
copy()
count()
extend()
index()
insert()
pop()
remove()
reverse()
sort()