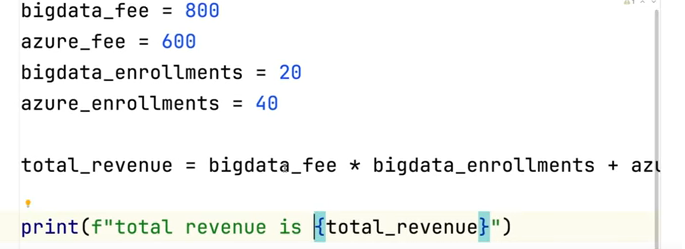
**Python by Sumit Sir**



+

\*

-

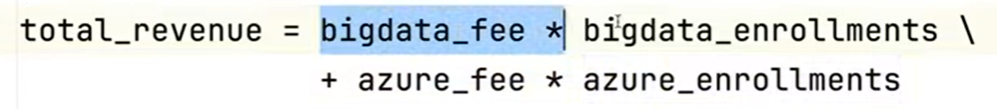
/

%

//

If we write the code and go to next line, we need to add **“\”**

**Ex:**

****

**To print in format**

**print(f“ total revenue is {variable}”)**

**Average\_price = (bigdata\_fee\*bigdata\_Enroll + azure\_fee\*azure\_enroll)/(bigdata\_enroll+azure\_enroll)**

**print(5+3\*8) 🡪 29**

**print( (5+3)\*8) 🡪 64**

**print(15/4) 🡪 3.75**

**print(15//4) 🡪 3**

**print(15%4) 🡪 3 (remainder)**

**Power**

**print( 2\*\*3) 🡪 8**

**Right side binding**

**print( 2\*\*2\*\*3) 🡪**

**(2\*\*3) 🡪 8**

**(2\*\*8) 🡪 256**

****

**2 is an operand and \*\* is the operator**

**Increment**

**fee = fee + 50**

**fee += 50**

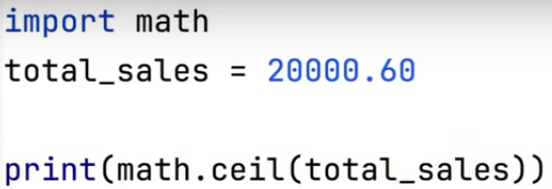
**Multiplication**

**fee \*= 2**

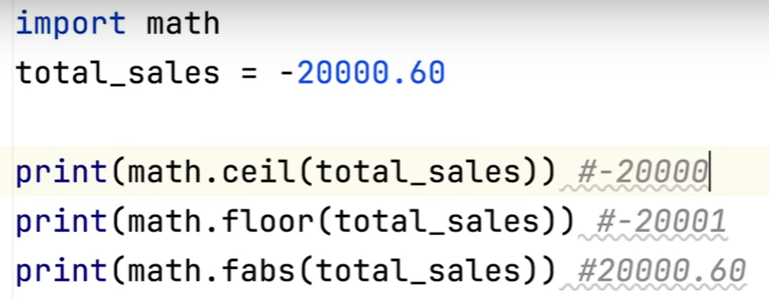
**To use round off/ round off to the value by applying the rule of 5 or greater number after decimal**

**fee = 200.60**

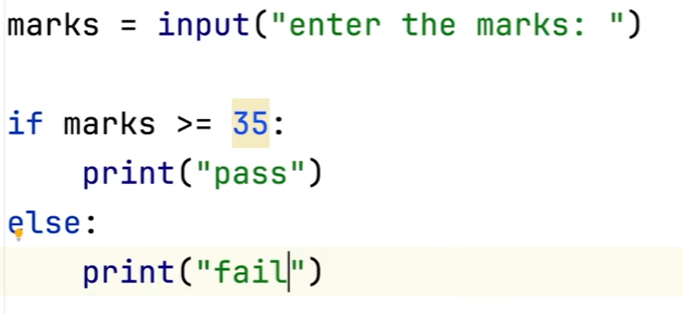
**print( math.ceil(fee)) 🡪 201**

****

**Print(math.floor(fee)) 🡪 200**

****

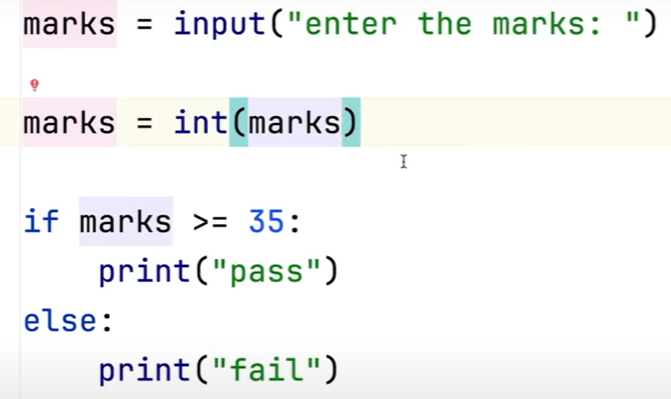
**Conditional statements**

****

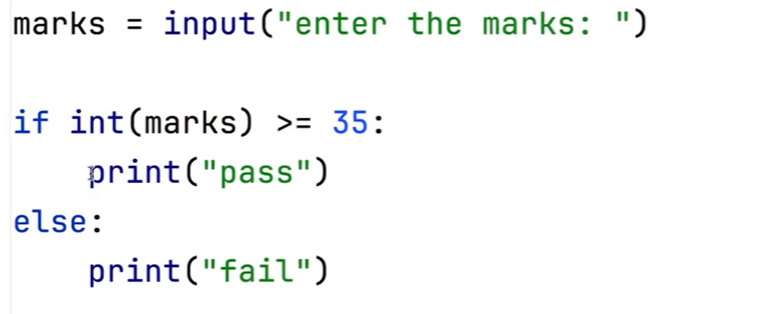
**Error**

**We need to convert marks into int**

**marks = int(marks)**

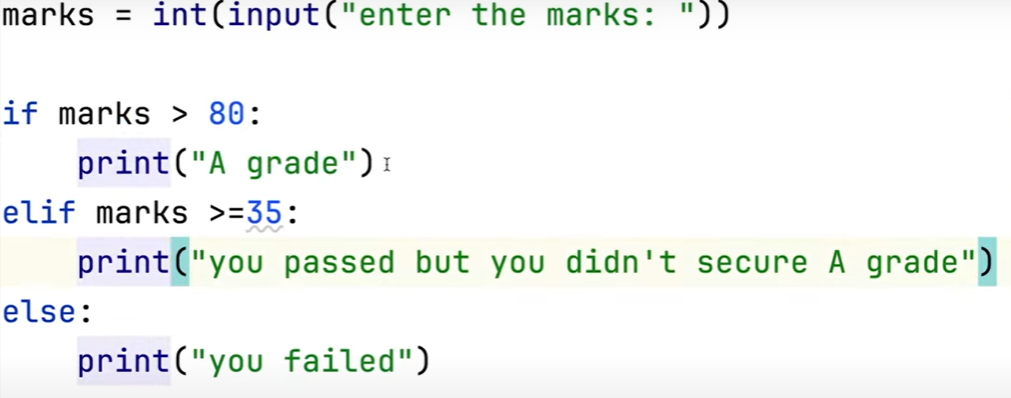
****

****

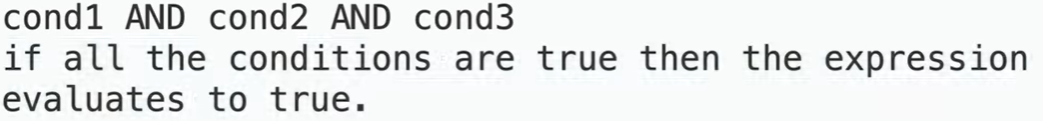
****

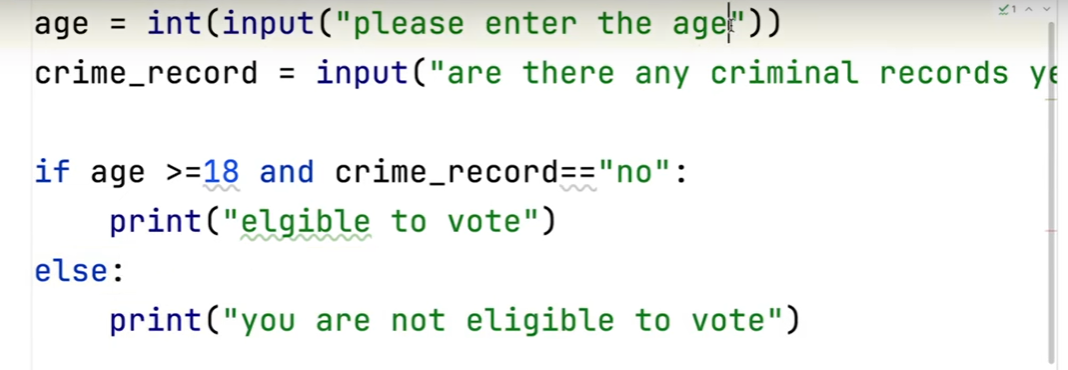
**Nested if statements**

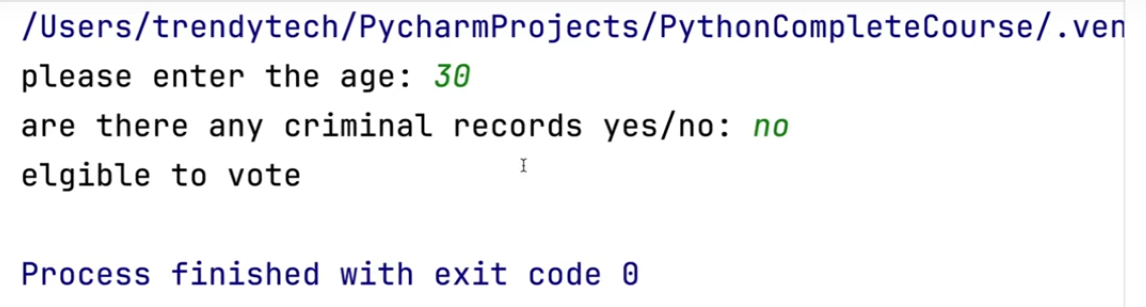
****

****

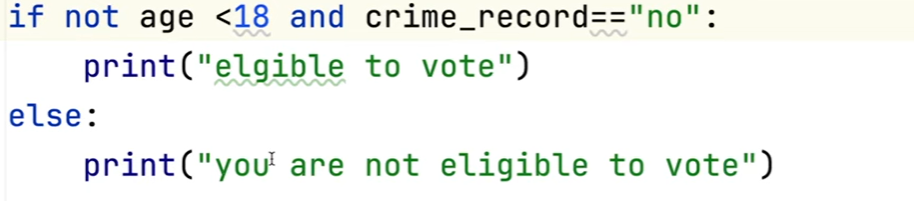
**Print(True and False) 🡪 False**

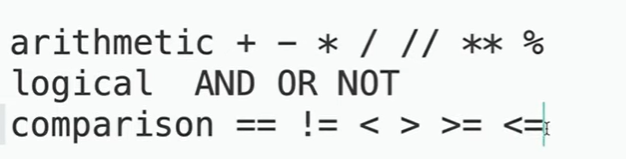
****

****

****

**Not operator**

****

****

**If any string is present in the variable, use “in” keyword**

name = "Sumit Mittal"  
print("Sumit" in name) *#True*print("sumit" in name) *#False*

**Working with Strings**

‘ ’

“ ”

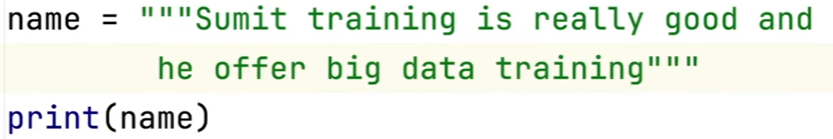
‘’’ ’’’

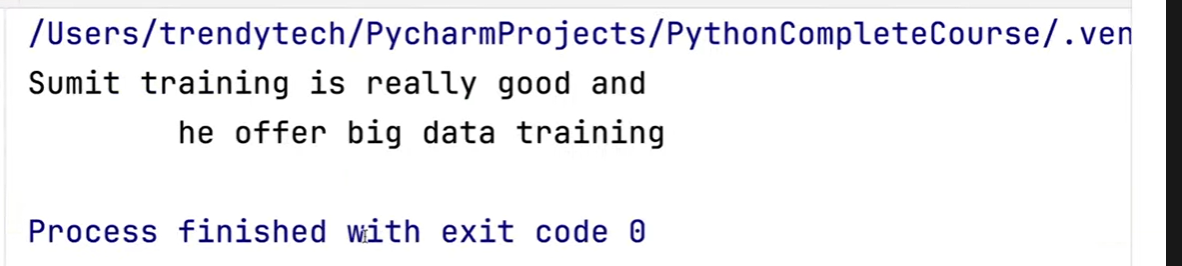
Sentence1 = “Jack’s batting is very good”

Sentence2 = ‘Jacks” batting is very good’

**Jack\’s”**  🡪 \ is an escape character

If we have string in multiple lines then we need to use “”” “””





**\n ->** escape character for new line

Sentense3 = “Jack’s batting is awesome \n as he scored some valuable runs”

\

\n 🡪 new line

\t 🡪 tab

fname = input("Enter first name ")  
lname = input("Enter last name ")  
print(fname+" "+lname)

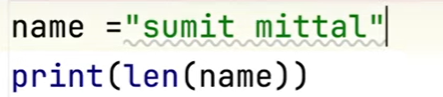
O/p:

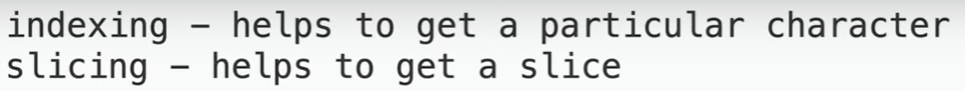
Enter first name Jack

Enter last name Will

Jack Will

To find the length of a string 🡪 len()





Index starts from 0 from left to right.

If we start indexing from right to left then it will start from -1.

Indexing

* Left to right 🡪 0
* Right to left 🡪 -1

String is immutable. We cannot modify the string.

**Slicing**

string1 = "test match"  
print(string1[0:2]) *#te*print(string1[5:10]) *#match*

print(string1[5:len(string1)]) *#match*

print(string1[5:]) *#match*

print(string1[:4]) *#test*

To print last 3 characters

print(string1[-3:len(string1)]) *#tch*

print(string1[-5:-2]) *#mat*

To print everything after the space

index = string1.find(" ")  
print(string1[index+1:len(string1)]) *#match*

Print the first word (it will be before first space)

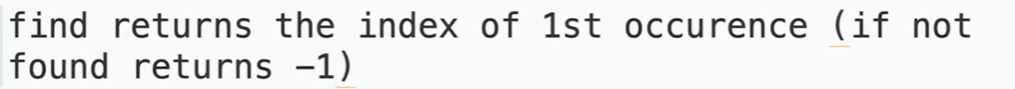
index = string1.find(" ")  
print(string1[0:index]) *#test*

Print the alternate character

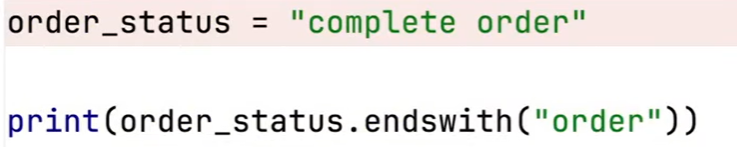
string2 = "abcdefghijk"  
print(string2) *#abcdefghijk*print(string2[1:4]) *#bcd*print(string2[1:9:2]) *#bdfh*print(string2[1::2]) *#bdfhj*

Print a string in reverse order

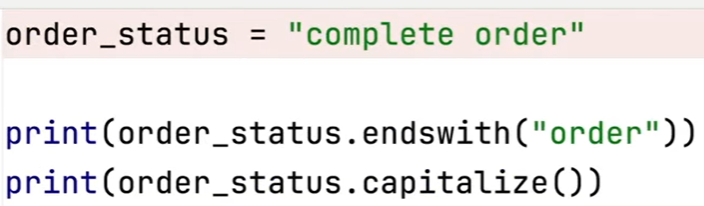
string3 = "Will jacks"  
print(string3[::-1]) *#skcaj lliW*

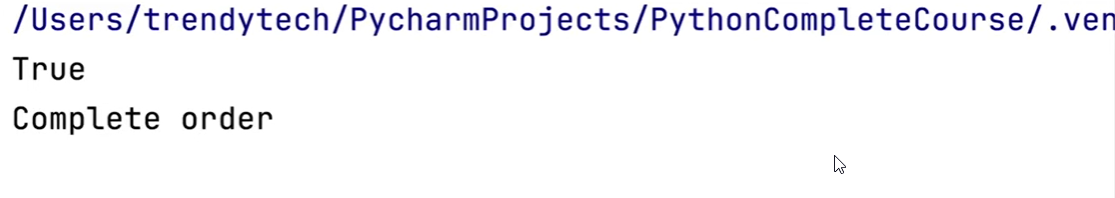






True



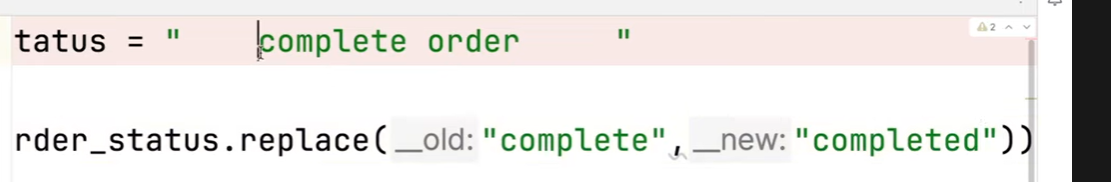


Strip() 🡪 it will remove all the left side and right side spaces of the string

lStrip() 🡪 it will remove all the left side spaces of the string

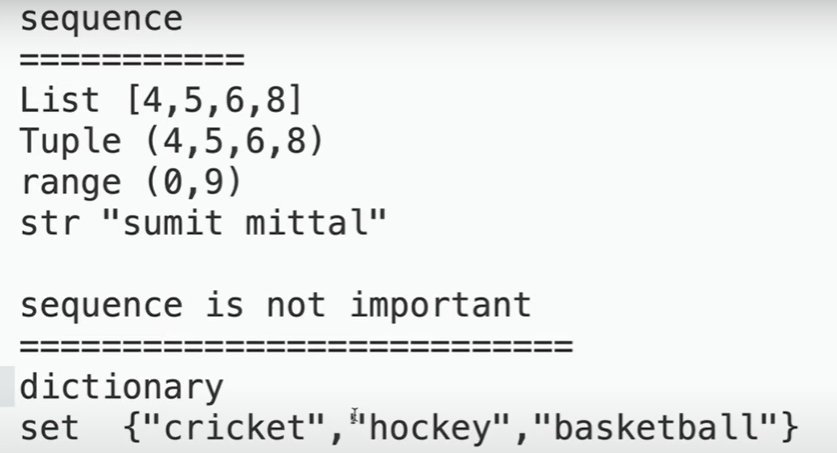
rStrip() 🡪 it will remove all the left side spaces of the string

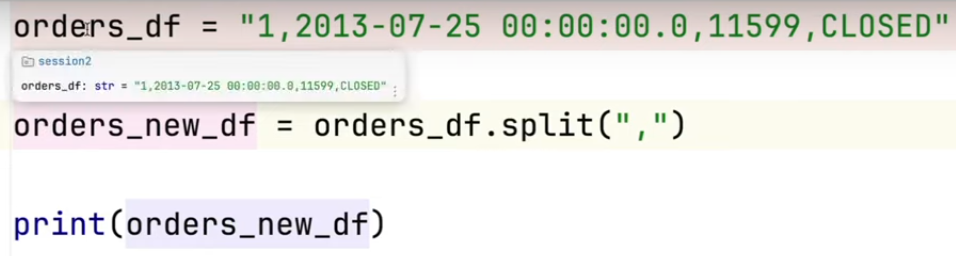
replace() 🡪 will replace a substring from the string

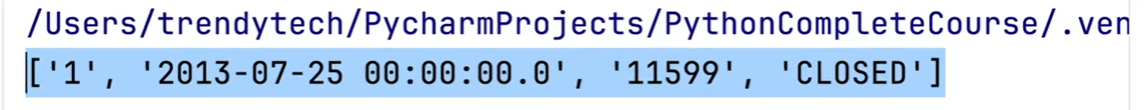


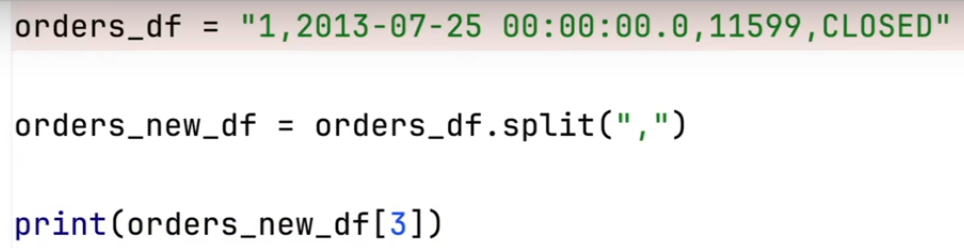
Data strutures

* List
* Tuple
* range
* str
* dictionary
* set









* CLOSED

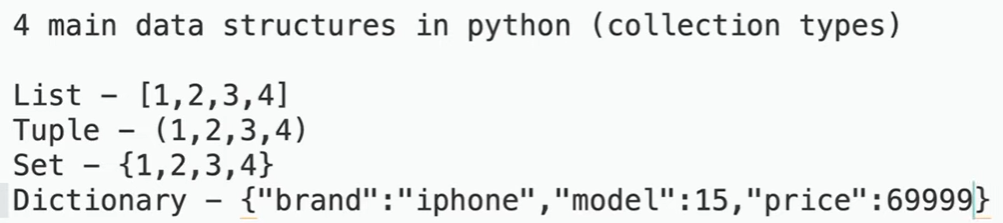
If the string is having extra spaces added like this

“\*\*\*\*\*, CLOSED “

We can use strip() function

Print(orders\_new\_df[3].strip()) 🡪 CLOSED

**List and Tuples**



For the Set and Dictionary, the sequence doesn’t matter

Sequence type – List, Tuple, String

Mutable vs Immutable

Mutable – value can be changed

Immutable – value cannot be changed

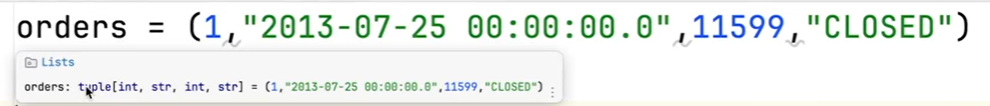
String - Immutable

Tuple - Immutable

Set – Mutable

Dictionary – Mutable

orders = [1, "2024-02-02",112233 ,"CLOSED"]  
print(type(orders))  
*# O/p - <class 'list'>*



orders = (1, "2024-02-02",112233 ,"CLOSED")  
print(type(orders))  
*# O/p - <class 'tuple'>*

List is mutable and Tuple is immutable

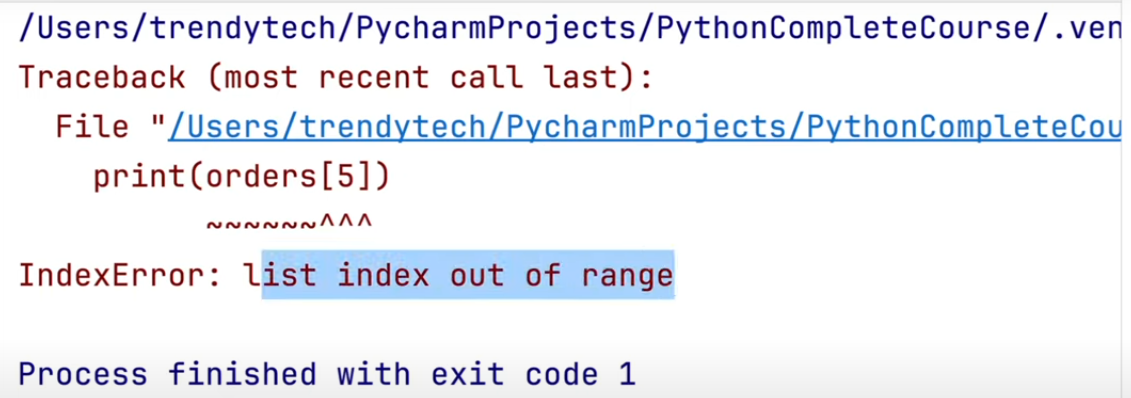
**To get element at 2nd index**

orders = [1, "2024-02-02",112233 ,"CLOSED"]

print(orders[2]) # 112233

**Element at 5th index**

orders = [1, "2024-02-02",112233 ,"CLOSED"]  
print(orders[5])

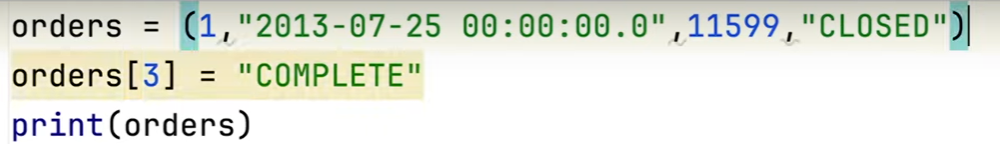


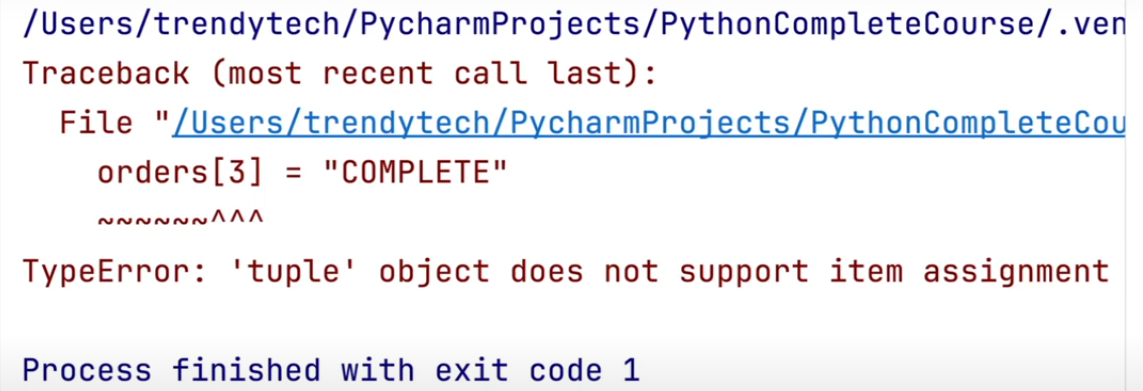
**Modify the string at 3rd index for List**

orders = [1, "2024-02-02",112233 ,"CLOSED"]  
orders[3] = 'COMPLETE'  
print(orders)

*#[1, '2024-02-02', 112233, 'COMPLETE']*

Modify the string at 3rd index for tuple



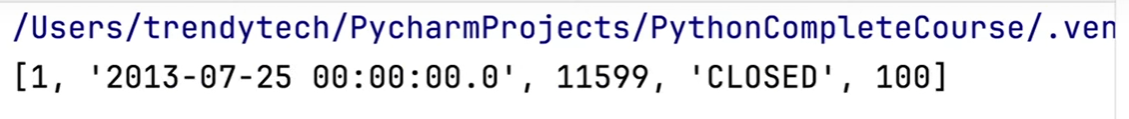


**For the list, to add element at the end 🡪 append()**

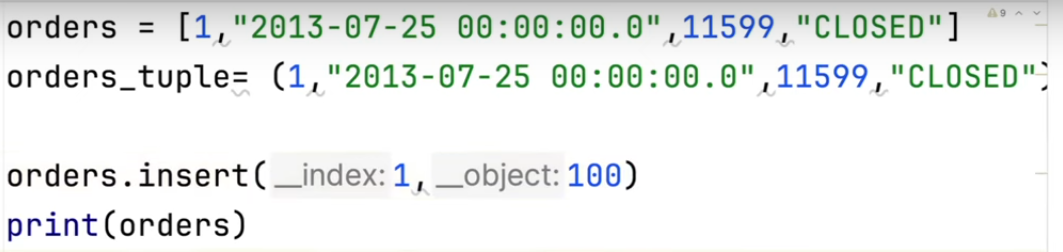
orders = [1, "2024-02-02",112233 ,"CLOSED"]  
orders.append(100)  
print(orders)  
*#[1, '2024-02-02', 112233, 'CLOSED', 100]*

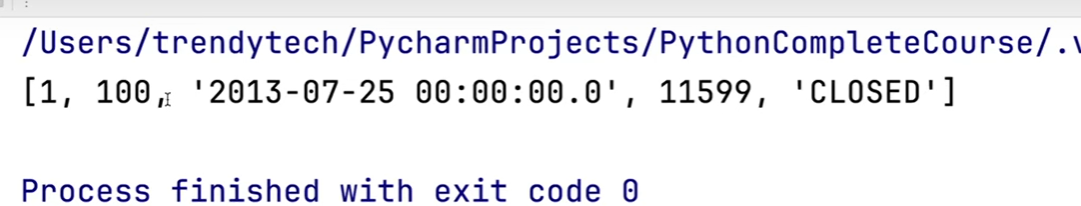
Append is applicable only for the List. For Tuple it won’t applicable.





To add element at particular index, **insert()** method is used.





To get total number of elements present in the List 🡪 **len()**

orders = [1, "2024-02-02",112233 ,"CLOSED"]  
orders.append(100)  
print(orders)  
*#[1, '2024-02-02', 112233, 'CLOSED', 100]*print(len(orders)) *#5*