1

PythonNotes

BasicsToAdvanced

UNSOLVED

References for all codes and CW notes : [Python Notes – OkItsNotOk (unaux.com)]

February 8, 2023

1.

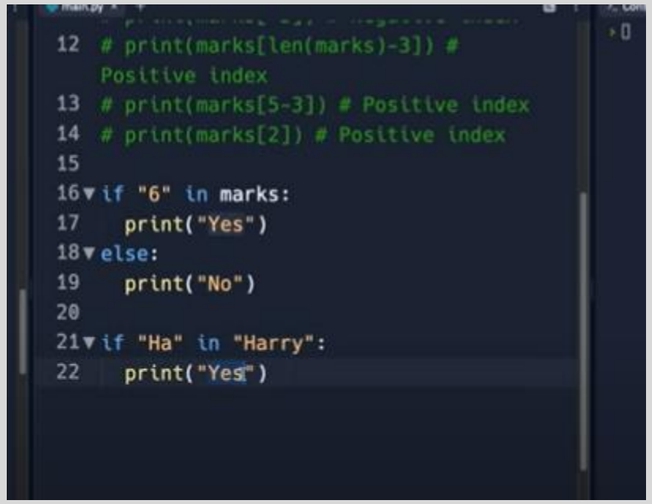
UNSOLVED — 02/08/2023 1:46 PM

length of list is simply no of elements in the list. elements can be numbers , strings, boolean or another list too

February 9, 2023

2.

UNSOLVED — 02/09/2023 1:35 AM



```
# print(marks[len(marks)-3]) # Positive index
# print(marks[5-3]) # Positive index
# print(marks[2]) # Positive index

if "6" in marks:
    print("Yes")
else:
    print("No")

if "Ha" in "Harry":
    print("Yes")
```

3.

UNSOLVED — 02/09/2023 7:40 PM

if m = l if we do m becames a reference of l i.e. its just like a name if we change elements of m , it gets changed in l too if we do m = l.copy() then it won't effect l and only the changes happen in m

```
l = [11, 45, 1, 2, 4, 6, 1, 1]
print(l)
# l.append(7)
# l.sort(reverse=True)
# l.reverse()
# print(l.index(1))
# print(l.count(1))
m = l.copy()
m[0] = 0
print(l)
```

4. [7:41 PM]

Output:

```
['voilet', 'indigo', 'blue', 'green']
```

insert():

This method inserts an item at the given index. User has to specify index and the item to be inserted within the insert() method.

Example:

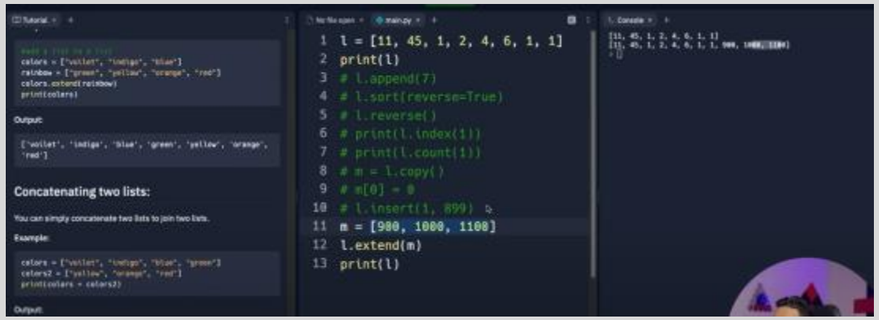
```
colors = ["voilet", "indigo", "blue"]
#          [0]       [1]      [2]

colors.insert(1, "green")   #inserts item at index 1
# updated list: colors = ["voilet", "green", "indigo", "blue"]
#       indexs              [0]      [1]      [2]
[3]

print(colors)
```

Output:

```
['voilet', 'green', 'indigo', 'blue']
```

5. [7:42 PM]

extends means add the list to list of the specified in the end



6. [7:44 PM]

if we want to create a list with l and m we can do : l.extend(m) this changes the list l and extends i.e. l gets effected whereas if we do k = l + m it does the same thing but doesn't effect the list l

7.

UNSOLVED — 02/09/2023 7:54 PM

tuple with one number as the element it says its class is integer to say it is a tuple write (1,)

8. [7:59 PM]

if we do slicing and assign it to something it gets created as a new tuple

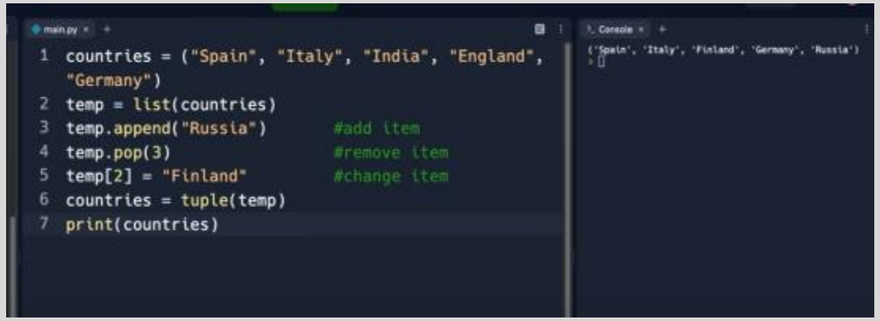
9.

UNSOLVED — 02/09/2023 8:03 PM

if we want to do anything to tuple we have to make a list make changes and make it tuple again

10. [8:04 PM]

like this



11. [8:09 PM]

count() in tuple in index() we can also specify the area to search (whattosearch, from , till) but it searches till till-1 only (like always) (edited)

12.

UNSOLVED — 02/09/2023 8:18 PM

if sathvik = {} and if I print type(sathvik) it says dictionary even if you thought it is going to be a set

13. [8:18 PM]

to create a set do sathvik = set() or sathvik = {some-elements}

14. [8:23 PM]

s1.union(s2) just prints the union set s1.update(s2) updates the set s1 with the union of s1,s2 (i.e. it gets changed)

15. [8:24 PM]

if we create a set = set1.intersection(set2) set becomes a set with only those intersection elements

16. [8:24 PM]

if we want to update also do set1.intersection_update(set2)

17.

UNSOLVED — 02/09/2023 8:27 PM

set1.isdisjoint(set2) disjoint meaning is the sets have no element in common the function prints True if there are no elements in common the function prints False if there are is one or more elements in common

18. [8:29 PM]

if set2 is a subset of set1 then set1 is called the superset of set2 set1.issuperset(set2) meaning is is set1 superset of set2 prints true or false

19. [8:29 PM]

the same way there is a function called issubset()

20.

UNSOLVED — 02/09/2023 9:11 PM

there is add() function if we want to add more than one item update() remove() / discard() pop() = usually removes the last element of the set since set is unordered we don't know what element it removes

21. [9:11 PM]

del set deletes the set and undefines it

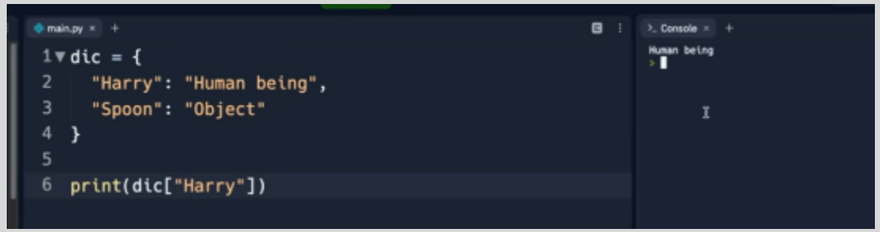
22. [9:12 PM]

if we don't want to delete the entire set but only want to remove all the elements we use clear()

23. [9:12 PM]

eg. set.clear()

24. [9:14 PM]



25. [9:16 PM]

in a way they are the same thing but if we do info["something"] since something is not there in the set it throws an error but if we do info.get("something") we get None as the output (edited)

26. [9:18 PM]

to access keys do print (info.key()) to access values do print(info.values()) or run a for loop like for key in info.keys() print (info[key]) (edited)

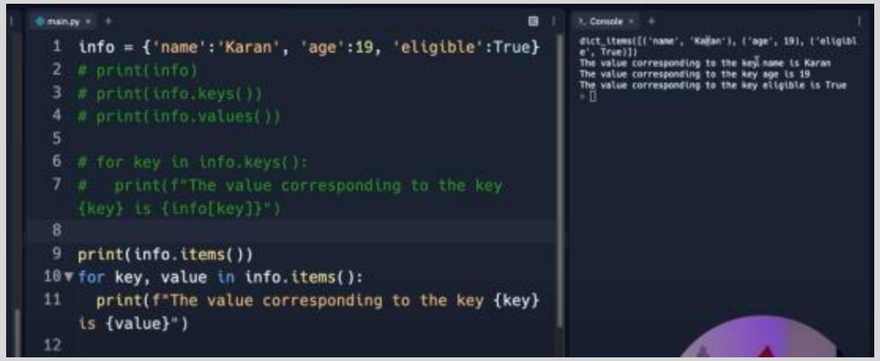
27.

UNSOLVED — 02/09/2023 9:19 PM

or the ultimate

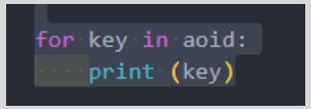
28. [9:21 PM]

sets



29. [9:24 PM]

to get key names to get keys/values or keys and values do key,value in set.items() and run a f string



30. [9:26 PM]

- - - - - - - - - - F U N C T I O N S - - - - - - - - - - -

31.

UNSOLVED — 02/09/2023 9:28 PM

FOR LIST :

- sort(): Sorts the list in ascending order.
- type(list): It returns the class type of an object.
- append(): Adds one element to a list.
- extend(): Adds multiple elements to a list.
- index(): Returns the first appearance of a particular value.
- max(list): It returns an item from the list with a max value.
- min(list): It returns an item from the list with a min value.
- len(list): It gives the overall length of the list.
- clear(): Removes all the elements from the list.
- insert(): Adds a component at the required position.
- count(): Returns the number of elements with the required value.
- pop(): Removes the element at the required position.
- remove(): Removes the primary item with the desired value.
- reverse(): Reverses the order of the list.
- copy(): Returns a duplicate of the list.

32. [9:29 PM]

FOR TUPLE

tuple. Therefore, we can call them tuple functions. Furthermore, these tuple functions make our work easy and efficient. Besides, there are a number of functions such as cmp(), len(), max(), min(), tuple(), index(), count(), sum(), any(), all(), sorted(), reversed().

33. [9:29 PM]

FOR SET

tuple. Therefore, we can call them tuple functions. Furthermore, these tuple functions make our work easy and efficient. Besides, there are a number of functions such as cmp(), len(), max(), min(), tuple(), index(), count(), sum(), any(), all(), sorted(), reversed().

34. [9:29 PM]

FOR DICTIONARY

Functions Name	Description
clear()	Removes all items from the dictionary
copy()	Returns a shallow copy of the dictionary
fromkeys()	Creates a dictionary from the given sequence
get()	Returns the value for the given key
items()	Return the list with all dictionary keys with values
keys()	Returns a view object that displays a list of all the keys in the dictionary in order of insertion
pop()	Returns and removes the element with the given key
popitem()	Returns and removes the key-value pair from the dictionary
setdefault()	Returns the value of a key if the key is in the dictionary else inserts the key with a value to the dictionary
update()	Updates the dictionary with the elements from another dictionary
values()	Returns a list of all the values available in a given dictionary

35.

UNSOLVED — 02/09/2023 10:19 PM

we can convert numbers, strings , tuples to list using list() function list(123) makes a list with 1, 2 and 3 as the elements

February 11, 2023

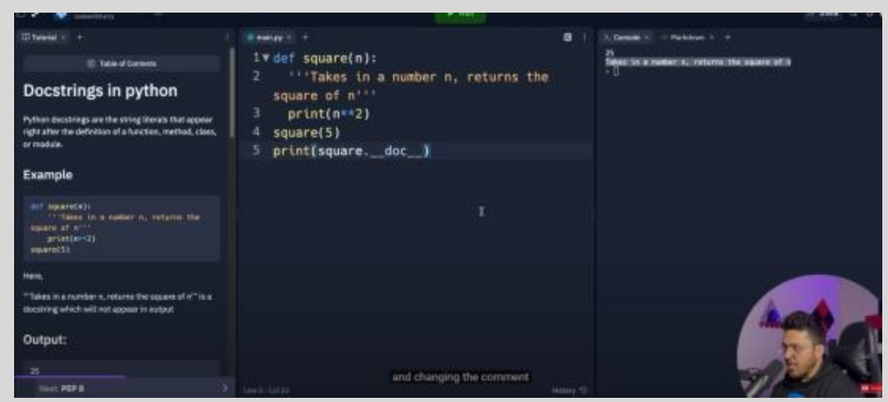
36.

UNSOLVED — 02/11/2023 6:47 PM

multi line comment ''' comment ''' or """ comment """

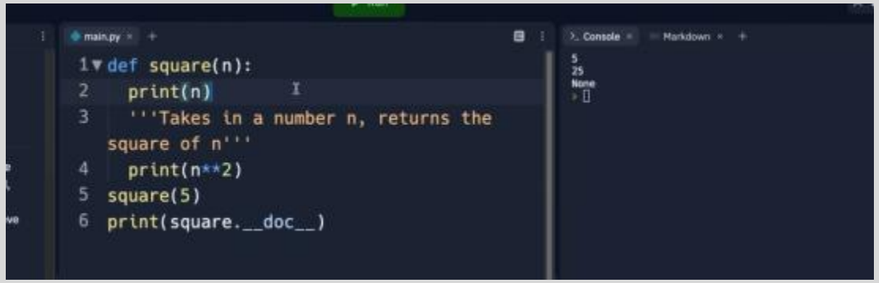
37. [6:52 PM]

The comments which are written in a function which give us a brief explanation of how the function works {{{{ T H E Y H A V E T O B E W R I T T E N A B O V E T H E F U N C T I O N B O D Y A N D R I G H T B E L O W T H E F U N C T I O N N A M E }}}}



38. [6:53 PM]

and this doesn't print docstring of the function



39.

UNSOLVED — 02/11/2023 7:25 PM

dictionary if you del {setname} it removes the set from variable space

40. [7:26 PM]

WE CAN USE ELSE WITH FOR-LOOP for i in range(5): print ("SathviK") else: print ("If else executed that means the for loop has run successfully till its last iteration and didn't break anywheere in between")

41.

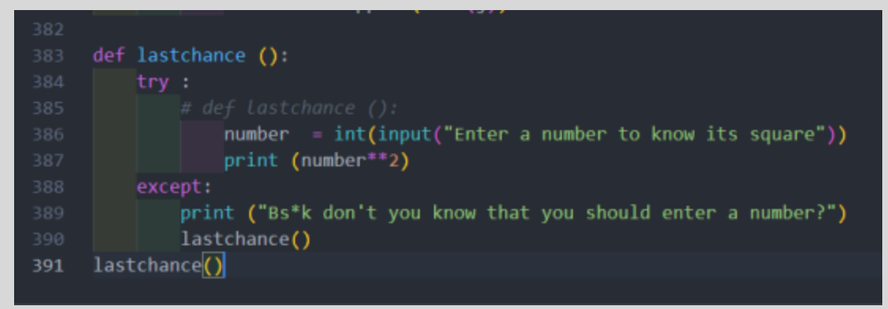
UNSOLVED — 02/11/2023 7:34 PM

try : #here write that code which could give errors the name itself says it just tries the lines of the code written in try and if it gets any error it goes to except part IF EXCEPT PART IS : 1) except Exception as e: print (e) This prints the Exception which usually shows in the terminal in normal lines. (doesn't end the program and doesn't show it in the red lines) 2) except : print ("Your random words which you want to tell the user") (edited)

42.

UNSOLVED — 02/11/2023 7:53 PM

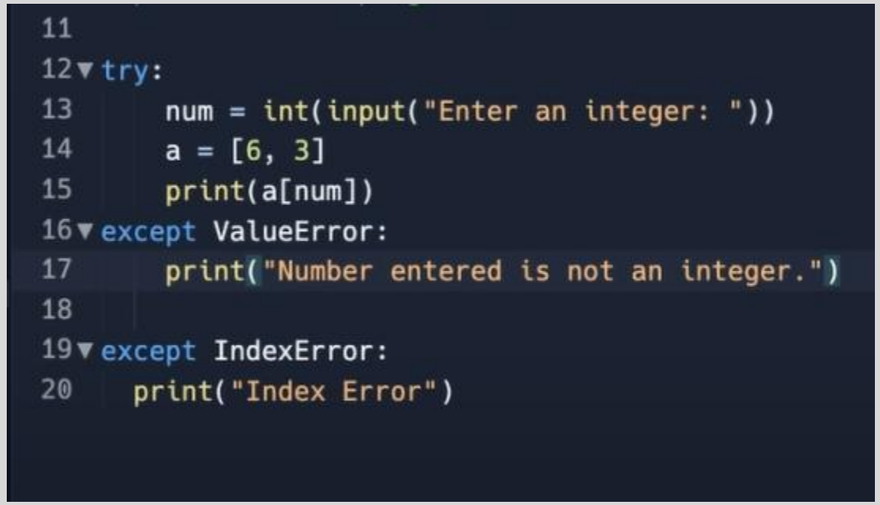
if you want to give another chance so that they input the things without the error you can do this



43. [7:57 PM]

Types of errors: **Value error** arises when you type string as a input in place of integer or something and so on **Index error** arises when you type an index as an input and it is out of range

44. [7:57 PM]

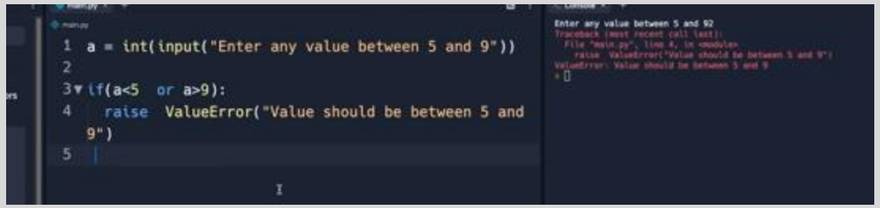


45.

UNSOLVED — 02/11/2023 8:05 PM

DIFFERENCE BETWEEN FINALLY AND NORMAL PRINT STATEMENT **If function got returned** it goes to exception if there is a exception occured **if function got returned without an exception** then it doesn't check remaining lines of statement in the function (here print is not even touched) BUT **even if function got returned without an exception** then also it goes to finally (after returning , since finally means finally)

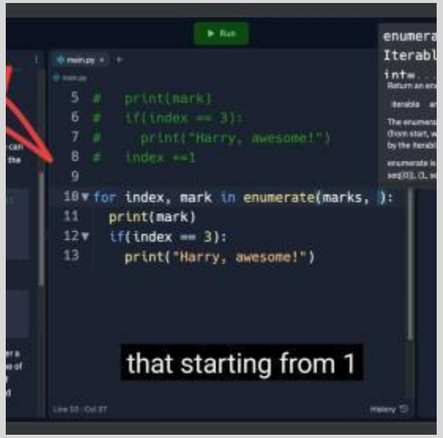
46. [8:09 PM]



February 12, 2023

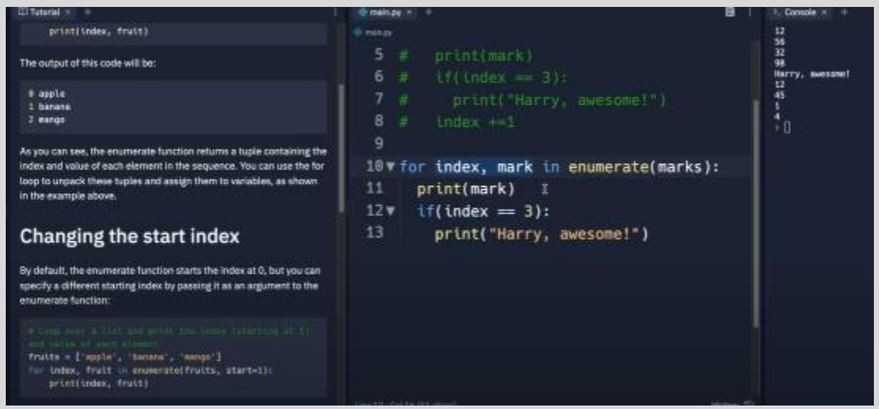
47.

UNSOLVED — 02/12/2023 12:26 PM



48.

UNSOLVED — 02/12/2023 5:34 PM

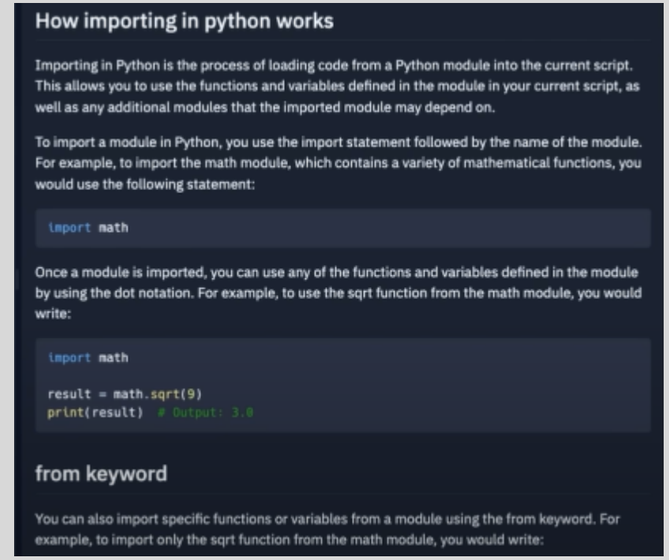


49. [5:37 PM]

actually enumerate(marks) has two values and they come as a pack of tuple (value1, value2) by writing index, mark will unpack the values to the corresponding first and second

50.

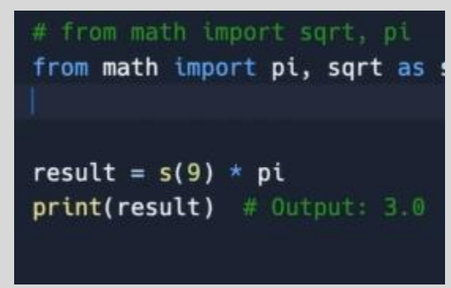
UNSOLVED — 02/12/2023 5:45 PM

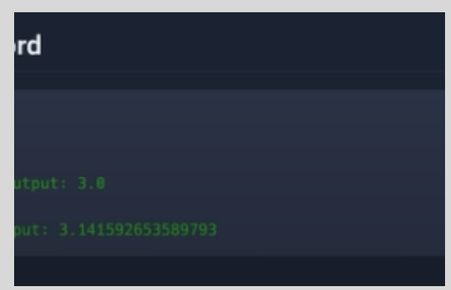


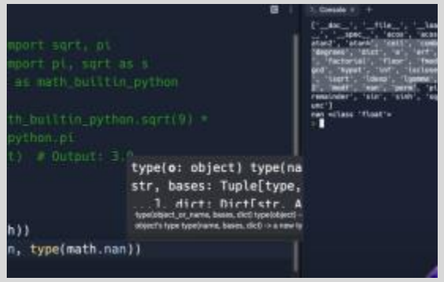
51.

UNSOLVED — 02/12/2023 5:58 PM

ALL THOSE ABOUT IMPORTS





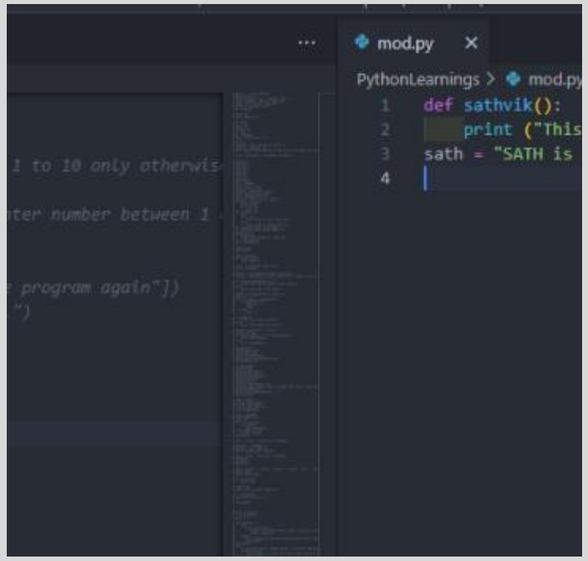


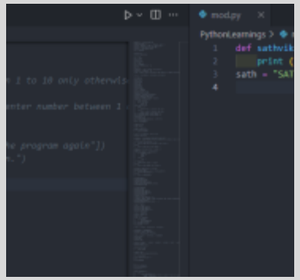
52. [5:58 PM]

Practice of modules

53. [6:00 PM]

These should of same folder All these works from mod import * is not advised to use





54.

UNSOLVED — 02/12/2023 6:06 PM

if there are some executionable functions in a file and if there is a call of that function in the file on importing the file they run the calling automatically so it runs once on importing the file (edited)

55. [6:11 PM]

just on importing it ran the file since there is a function call in the file itself

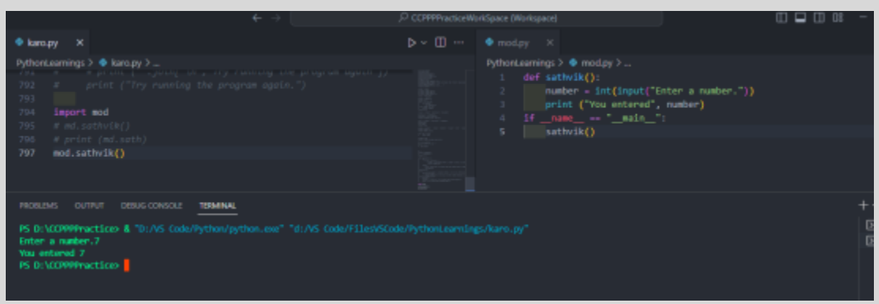
56. [6:12 PM]

if I WRITE function call in the file where we are importing the module it ran the program twice in total

57.

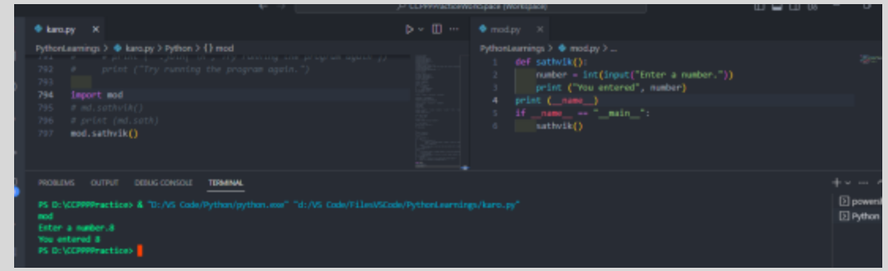
UNSOLVED — 02/12/2023 6:18 PM

here mod is the main function for its own code written in its file if mod.py is being executed then only the flow goes into if condition otherwise it doesn't since we are running it from karo and since it is the main function / main file of that code **only the calling of sathvik() in karo works and imports doesn't run the function** main simply means it is running / executed from the same file (edited)

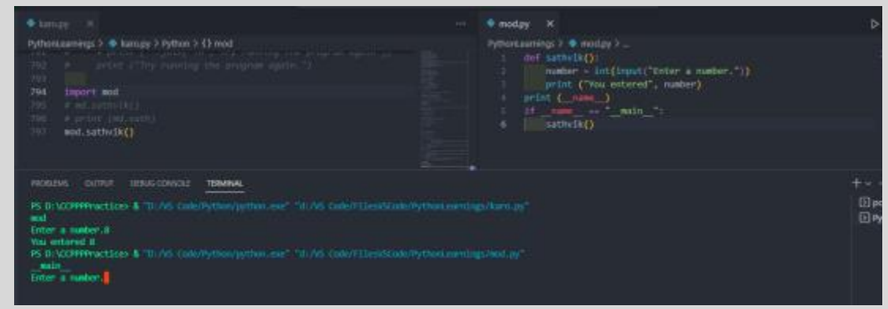


58. [6:23 PM]

name tells how it is being executed/ran in the figure I ran the mod file from karo therefore it said it is being ran as **mod** that is the file is ran somewhere else if it ran in the same place the name would be printed as **main** (edited)



59. [6:24 PM]



60.

UNSOLVED — 02/12/2023 6:27 PM

This can be useful if you have code that you want to reuse in multiple scripts, but you only want it to run when the script is run directly and not when it's imported as a module.

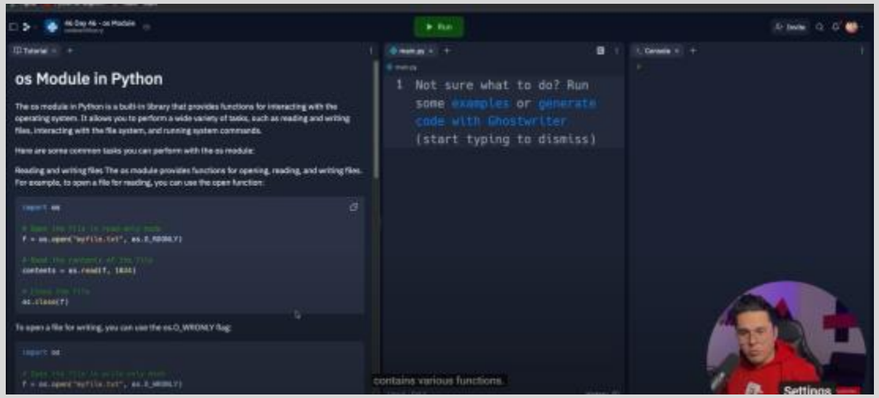
Is it a necessity?

It's important to note that the `if __name__ == "__main__"` idiom is not required to run a Python script. You can still run a script without it by simply calling the functions or running the code you want to execute directly. However, the `if __name__ == "__main__"` idiom can be a useful tool for organizing and separating code that should be run directly from code that should be imported and used as a module.

In summary, the `if __name__ == "__main__"` idiom is a common pattern used in Python scripts to determine whether the script is being run directly or being imported as a module into another script. It allows you to reuse code from a script by importing it as a module into another script, without running the code in the original script.

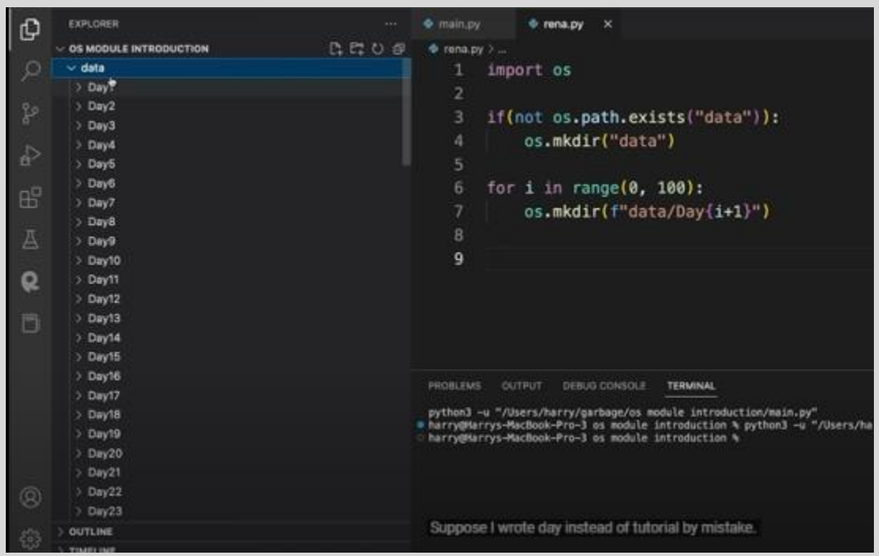
61.

UNSOLVED — 02/12/2023 7:40 PM

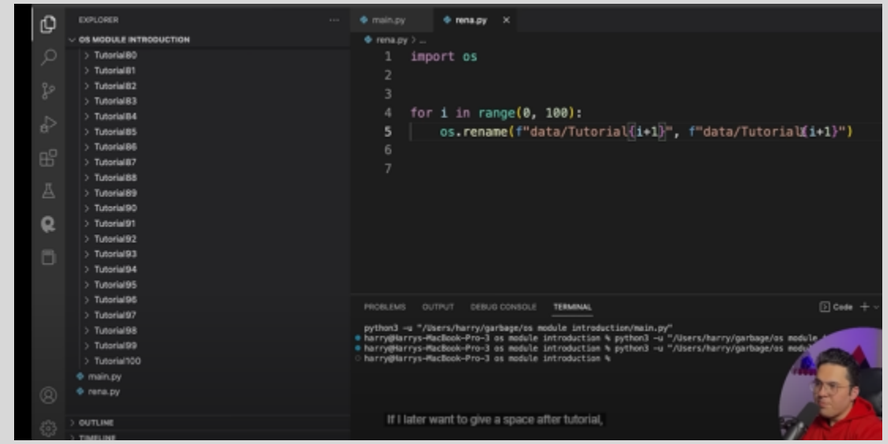


62.

UNSOLVED — 02/12/2023 9:33 PM



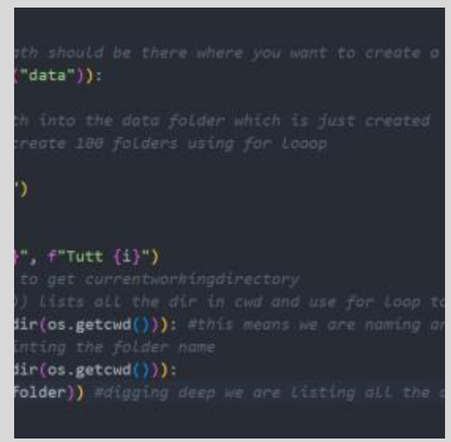
63. [9:33 PM]

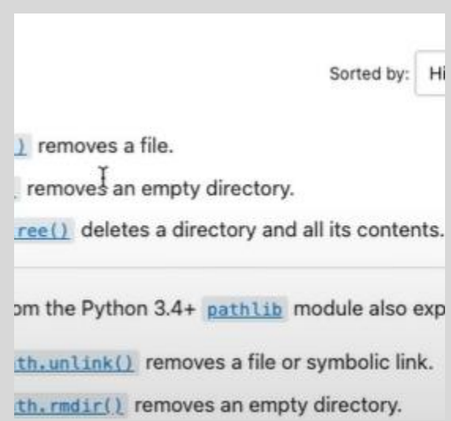


64.

UNSOLVED — 02/12/2023 10:24 PM

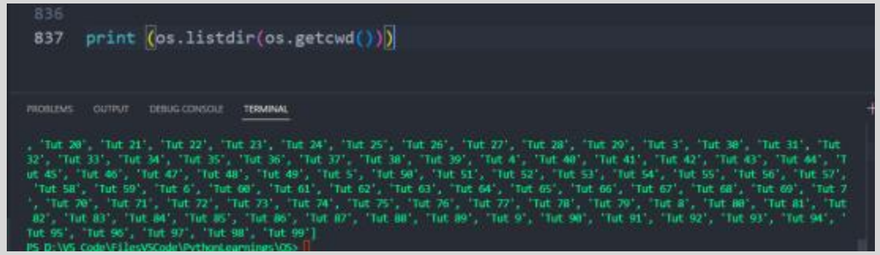
comment out the lines as necessary after creating the files comment it and run folders after seeing them if you don't want to see them again comment again and also check the remaining functions remove and all





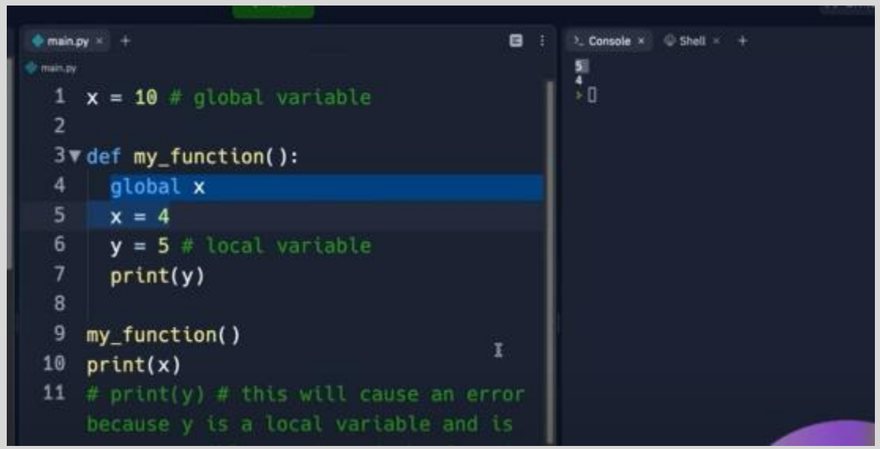
65. [10:28 PM]

if you want all at once.



66.

UNSOLVED — 02/12/2023 10:54 PM



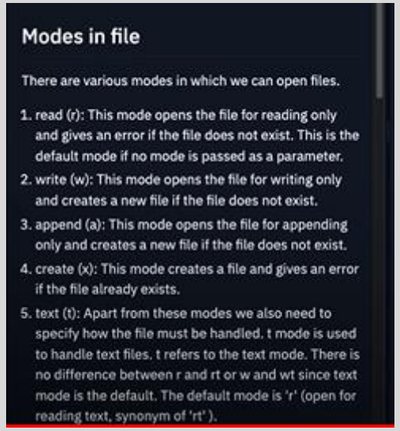
67. [10:56 PM]

I will write two lines here
1) global x
2) x = 4
what is the meaning of each line in python?

1) The first line declares a global variable called "x" which can be used in any part of the program.
2) The second line assigns the value 4 to the global variable "x".

68.

UNSOLVED — 02/12/2023 11:21 PM



Modes in file

There are various modes in which we can open files.

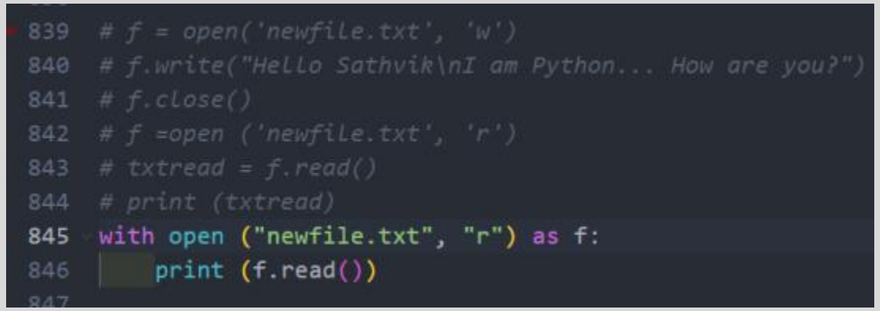
1. read (r): This mode opens the file for reading only and gives an error if the file does not exist. This is the default mode if no mode is passed as a parameter.
2. write (w): This mode opens the file for writing only and creates a new file if the file does not exist.
3. append (a): This mode opens the file for appending only and creates a new file if the file does not exist.
4. create (x): This mode creates a file and gives an error if the file already exists.
5. text (t): Apart from these modes we also need to specify how the file must be handled. t mode is used to handle text files. t refers to the text mode. There is no difference between r and rt or w and wt since text mode is the default. The default mode is 'r' (open for reading text, synonym of 'rt').

69. [11:23 PM]

70.

UNSOLVED — 02/12/2023 11:29 PM

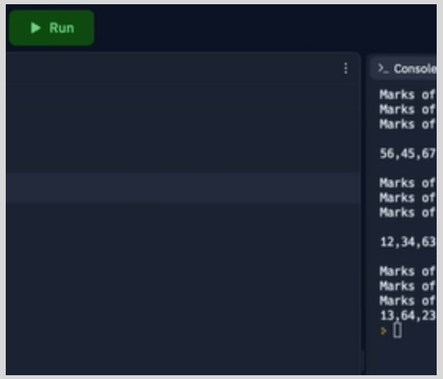
or else we can do this this with statement automatically closes it after the work



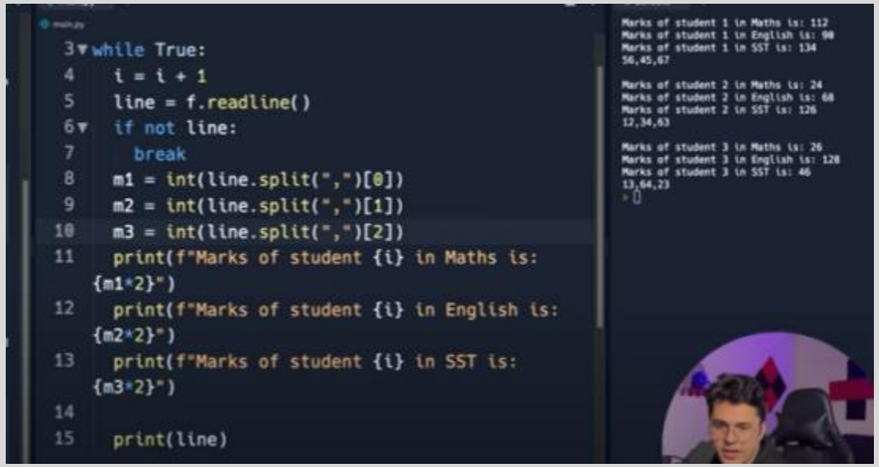
February 13, 2023

71.

UNSOLVED — 02/13/2023 1:28 PM

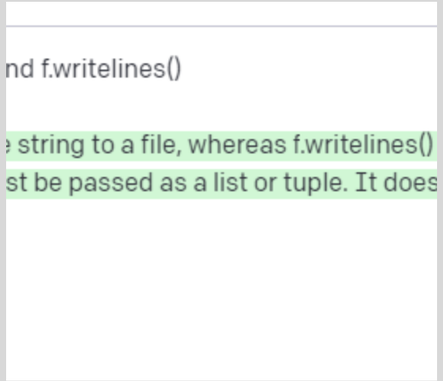


72. [1:29 PM]



73.

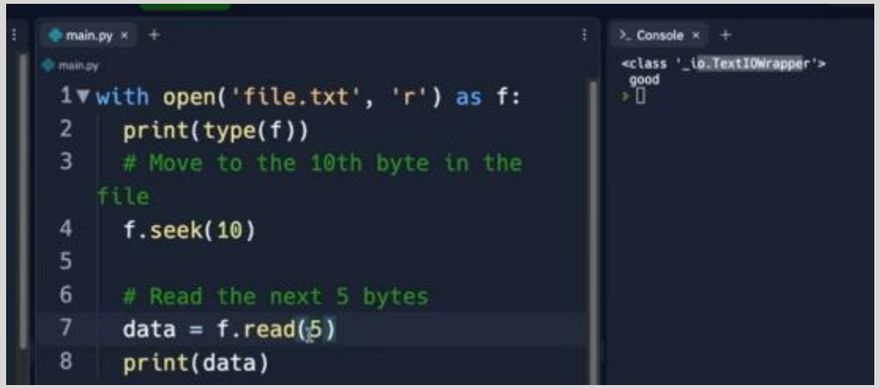
UNSOLVED — 02/13/2023 1:36 PM



74.

UNSOLVED — 02/13/2023 2:28 PM

seek moves the pointer to the first letter to that byte



75.

UNSOLVED — 02/13/2023 2:38 PM

```
my_empty_set = set()

# or

my_empty_set = {}
```

76. [2:43 PM]

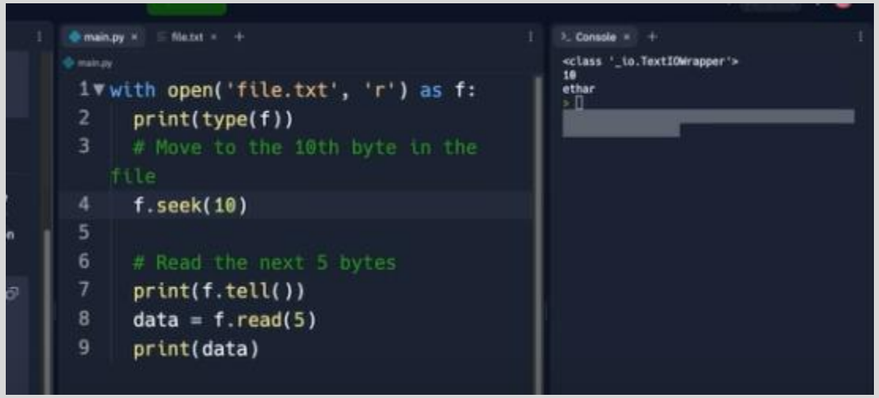
```
myset = set()
# print (type(myset))
for i in range(10):
    myset.add(i*i)
myset.update([1,2,3,10000])
print (myset)
```

77. [2:44 PM]

you can use update([1,2,3.10000]) or ({1,2,3,100000})

78.

UNSOLVED — 02/13/2023 2:55 PM



79. [3:01 PM]

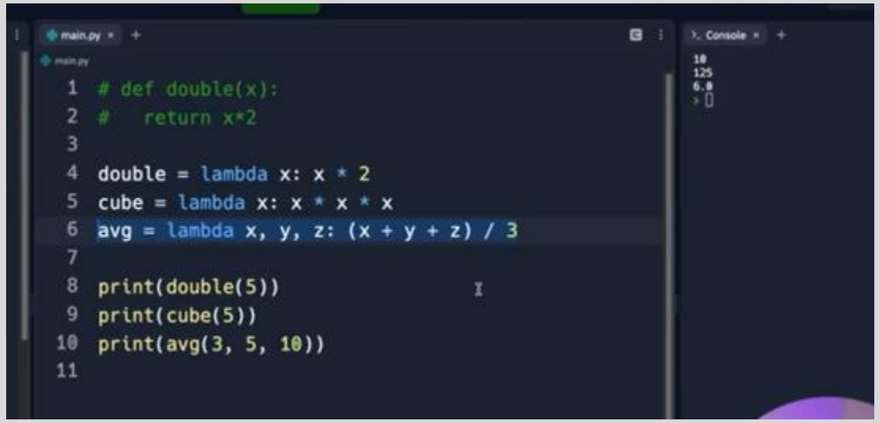
```
with open ("newlines.txt", "r") as f:
    f.seek(7)
    print (f.tell())
    print (f.read(40))
#'''seek points the index given here 7
# tell tell the index it is pointing now
# read reads number of characters from the index
# if read(4) it reads the current position and next three'''
```

80.

UNSOLVED — 02/13/2023 3:35 PM

if we add truncate it cuts the file to that bytes it keeps only those number of first characters.

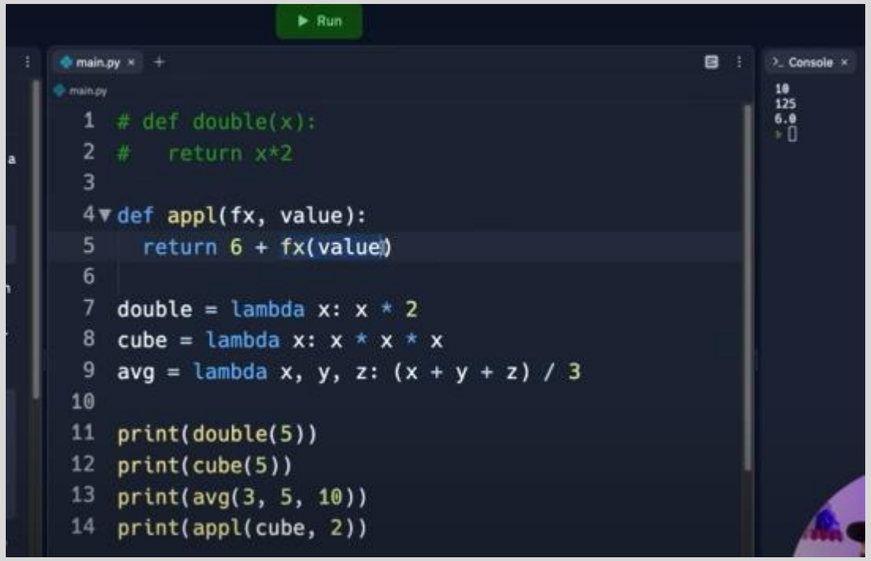
81. [3:40 PM]



82.

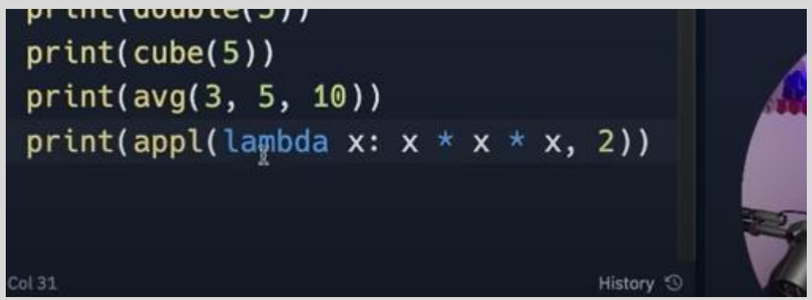
UNSOLVED — 02/13/2023 3:44 PM

we can pass function as a argument also



83. [3:46 PM]

or also do this

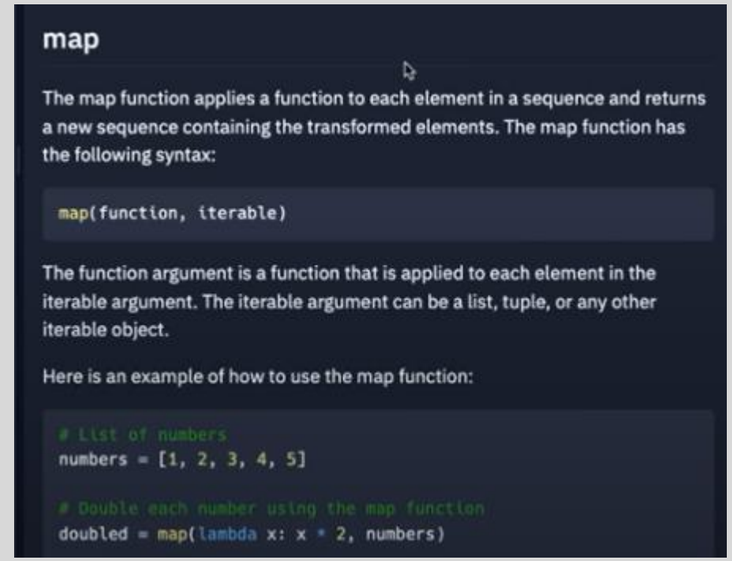


84. [3:50 PM]

how to use map function

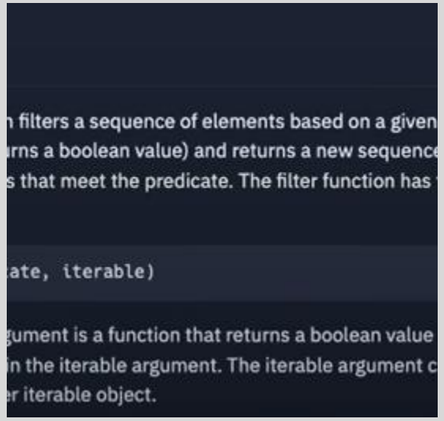
85.

UNSOLVED — 02/13/2023 4:01 PM



86. [4:02 PM]

how to use filter



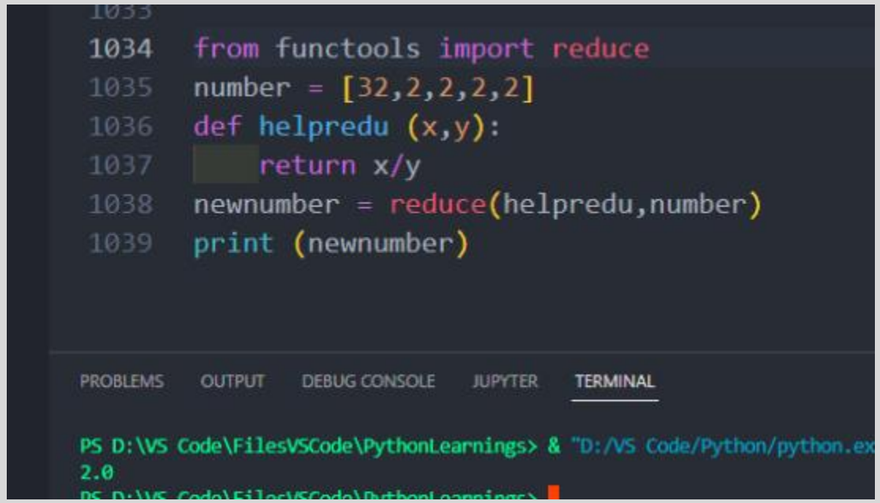
87.

UNSOLVED — 02/13/2023 4:20 PM

filter is used always with a function which returns a boolean value we have to turn map and filter to list

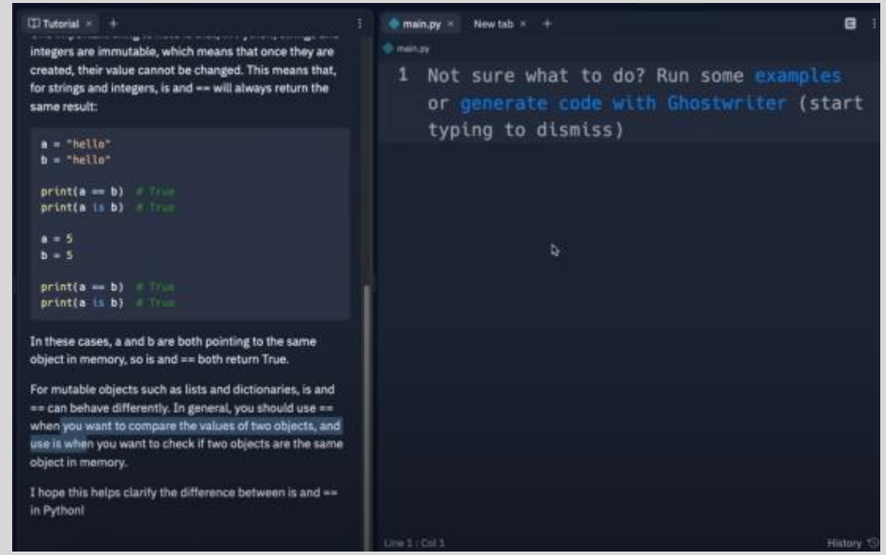
88.

UNSOLVED — 02/13/2023 7:40 PM

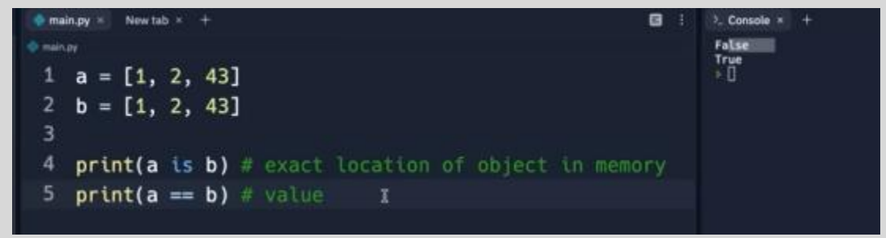


89.

UNSOLVED — 02/13/2023 9:06 PM

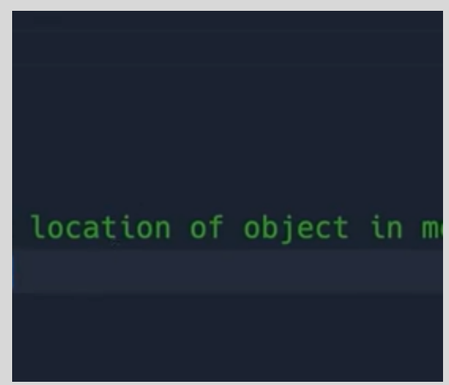


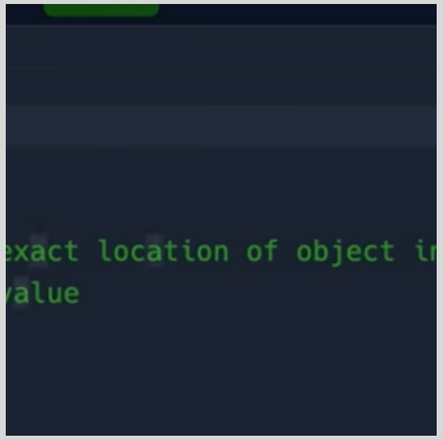
90. [9:08 PM]



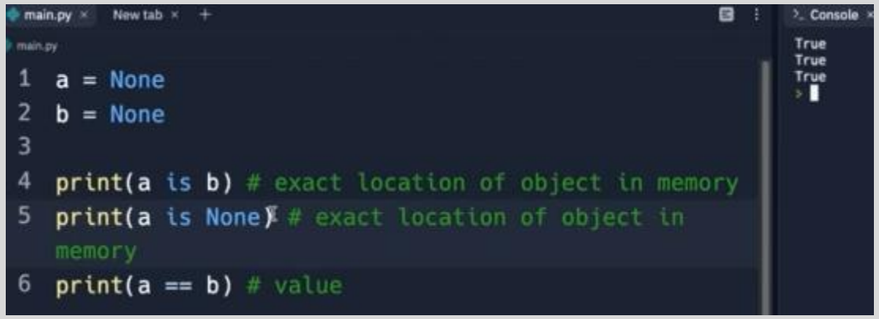
91. [9:10 PM]

here a and b are pointing at immutable things python don't allocate different memories for the same immutable things so a and b points at the same location





92. [9:10 PM]



February 14, 2023

93.

UNSOLVED — 02/14/2023 4:08 PM

Introduction to Object-oriented programming

Introduction to Object-Oriented Programming in Python: In programming languages, mainly there are two approaches that are used to write program or code.

- 1). Procedural Programming
- 2). Object-Oriented Programming

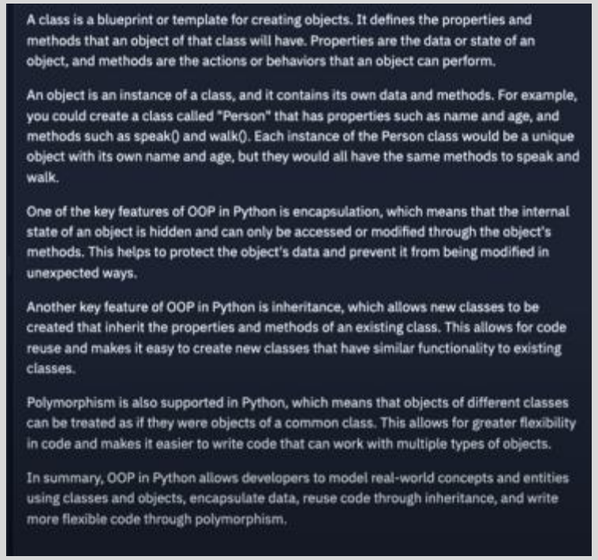
The procedure we are following till now is the "Procedural Programming" approach. So, in this session, we will learn about Object Oriented Programming (OOP). The basic idea of object-oriented programming (OOP) in Python is to use classes and objects to represent real-world concepts and entities.

A class is a blueprint or template for creating objects. It defines the properties and methods that an object of that class will have. Properties are the data or state of an object, and methods are the actions or behaviors that an object can perform.

An object is an instance of a class, and it contains its own data and methods. For example, you could create a class called "Person" that has properties such as name and age, and methods such as speak() and walk(). Each instance of the Person class would be a unique object with its own name and age, but they would all have the same methods to speak and walk.

One of the key features of OOP in Python is encapsulation, which means that the internal state of an object is hidden and can only be accessed or modified through the object's methods. This helps to protect the object's data and prevent it from being modified in unexpected ways.

94. [4:13 PM]



A class is a blueprint or template for creating objects. It defines the properties and methods that an object of that class will have. Properties are the data or state of an object, and methods are the actions or behaviors that an object can perform.

An object is an instance of a class, and it contains its own data and methods. For example, you could create a class called "Person" that has properties such as name and age, and methods such as speak() and walk(). Each instance of the Person class would be a unique object with its own name and age, but they would all have the same methods to speak and walk.

One of the key features of OOP in Python is encapsulation, which means that the internal state of an object is hidden and can only be accessed or modified through the object's methods. This helps to protect the object's data and prevent it from being modified in unexpected ways.

Another key feature of OOP in Python is inheritance, which allows new classes to be created that inherit the properties and methods of an existing class. This allows for code reuse and makes it easy to create new classes that have similar functionality to existing classes.

Polymorphism is also supported in Python, which means that objects of different classes can be treated as if they were objects of a common class. This allows for greater flexibility in code and makes it easier to write code that can work with multiple types of objects.

In summary, OOP in Python allows developers to model real-world concepts and entities using classes and objects, encapsulate data, reuse code through inheritance, and write more flexible code through polymorphism.

95.

UNSOLVED — 02/14/2023 4:37 PM

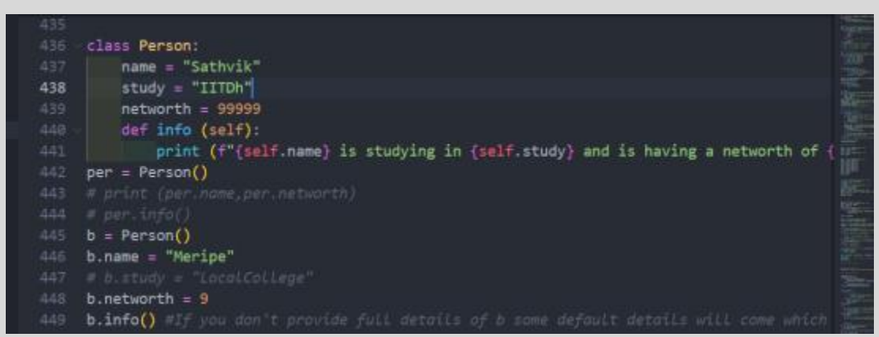
96.

UNSOLVED — 02/14/2023 6:52 PM

97. [6:52 PM]

or if you don't want this also put None for everything (edited)

98. [6:53 PM]



99.

UNSOLVED — 02/14/2023 10:28 PM

February 15, 2023

100.

UNSOLVED — 02/15/2023 10:11 AM

331

332

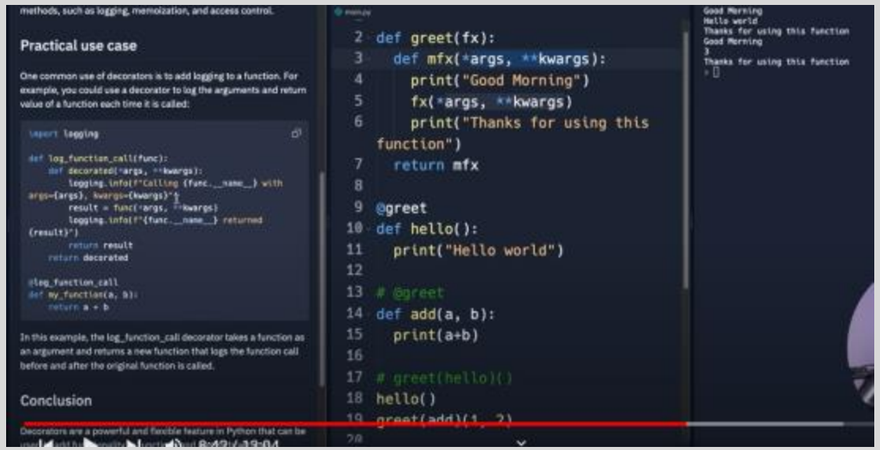
333 101. [10:12 AM]

334 or it also works fine

335

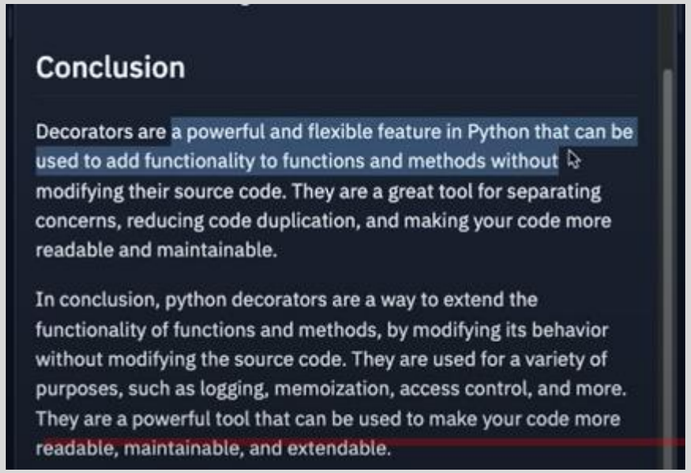
336 102. [10:14 AM]

337 NICE see this



338

339 103. [10:16 AM]



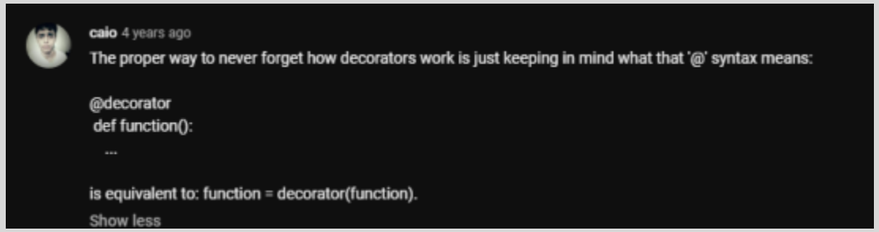
Conclusion

Decorators are a powerful and flexible feature in Python that can be used to add functionality to functions and methods without modifying their source code. They are a great tool for separating concerns, reducing code duplication, and making your code more readable and maintainable.

In conclusion, python decorators are a way to extend the functionality of functions and methods, by modifying its behavior without modifying the source code. They are used for a variety of purposes, such as logging, memoization, access control, and more. They are a powerful tool that can be used to make your code more readable, maintainable, and extendable.

104.

UNSOLVED — 02/15/2023 10:27 AM



caio 4 years ago

The proper way to never forget how decorators work is just keeping in mind what that '@' syntax means:

```
@decorator
 def function():
   ...
```

is equivalent to: function = decorator(function).

Show less

February 23, 2023

105.

UNSOLVED — 02/23/2023 1:59 AM

REGULAR METHODS AUTOMATICALLY TAKES OBJECT AS ARGUMENT (SELF) CLASS METHODS TAKE CLASS AS ARGUMENT STATIC METHODS DON'T PASS ANYTHING AS ARGUMENT (edited)

106.

UNSOLVED — 02/23/2023 2:14 PM

If our subclass contains just one / two more information than its parent class we can use super().init(arugementswhichshouldbecopiedfromparentclass) self.newargument or Classname.__init(self,arugementswhichshouldbecopiedfromparentclass) self.newargument

107.

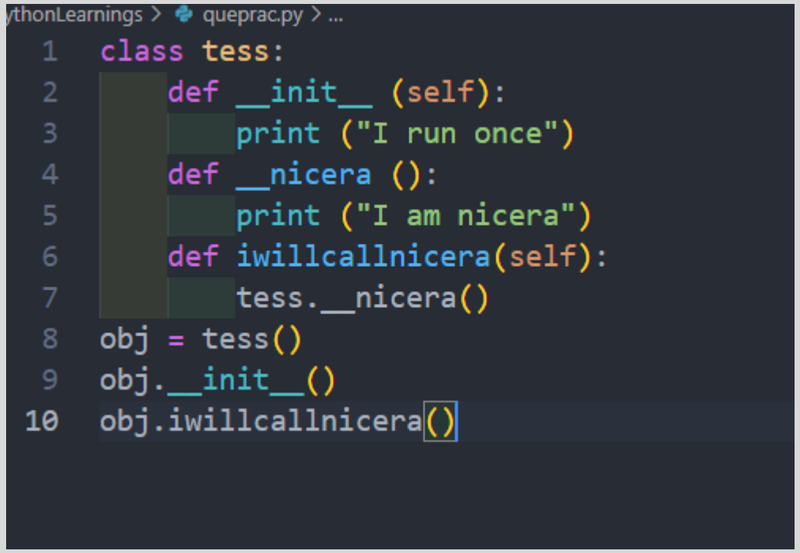
UNSOLVED — 02/23/2023 2:21 PM

We should not pass mutable datatypes as default argument

108.

UNSOLVED — 02/23/2023 9:51 PM

It will only work this way we can't a call a private function private method or function : leading it has two __ so we have to create a public method which call it in the class itself and private function can't be accessed from the outside



ythonLearnings > queprac.py > ...

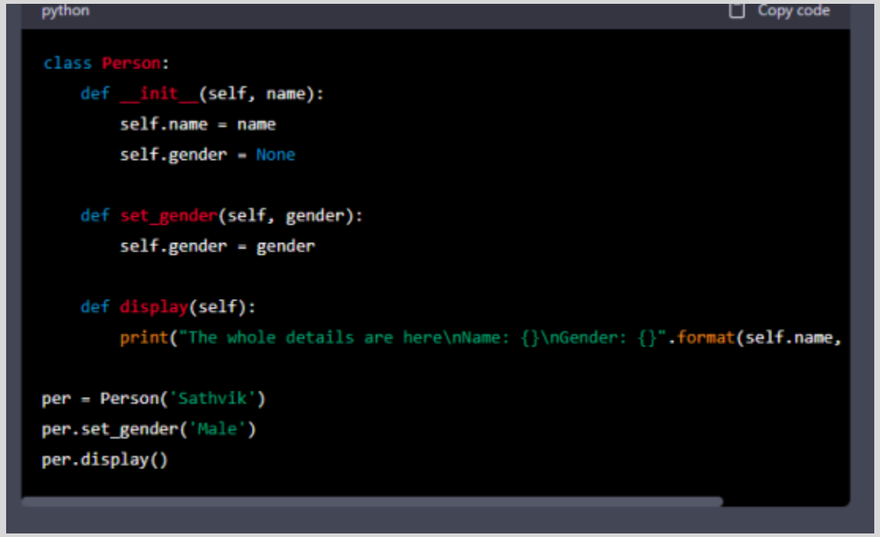
```
class tess:
    def __init__ (self):
        print ("I run once")
    def __nicera ():
        print ("I am nicera")
    def iwillcallnicera(self):
        tess.__nicera()
obj = tess()
obj.__init__()
obj.iwillcallnicera()
```

February 24, 2023

109.

UNSOLVED — 02/24/2023 11:55 AM

This is how we can give all the details necessary to init while giving what other methods wants sepcifically to them



python Copy code

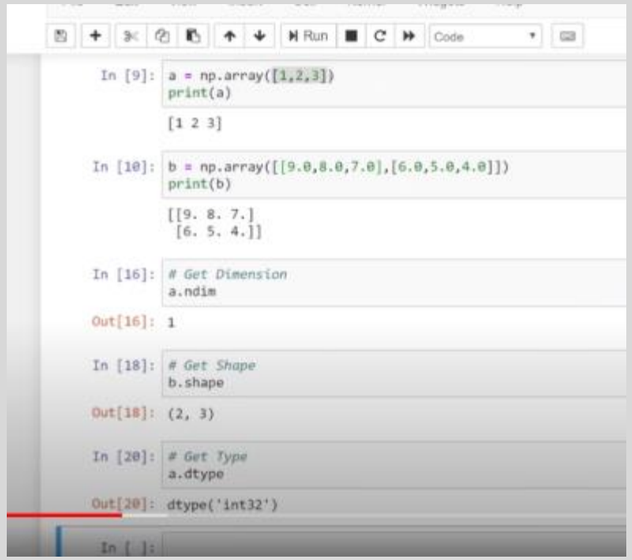
```
class Person:
    def __init__(self, name):
        self.name = name
        self.gender = None

    def set_gender(self, gender):
        self.gender = gender

    def display(self):
        print("The whole details are here\nName: {}\nGender: {}".format(self.name,

per = Person('Sathvik')
per.set_gender('Male')
per.display()
```

March 1, 2023

110.

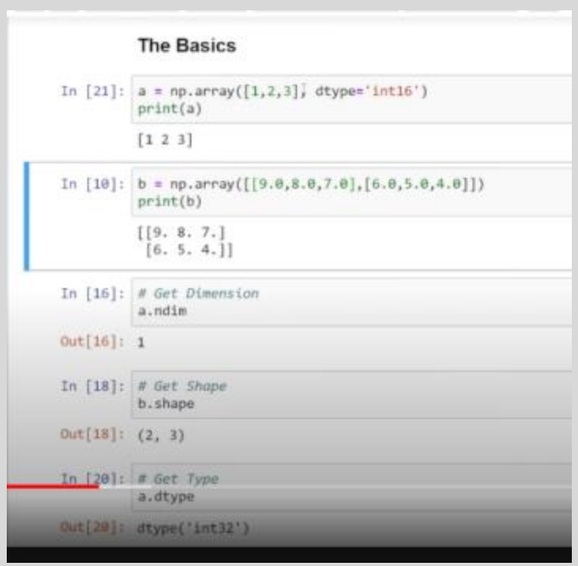
UNSOLVED — 03/01/2023 11:40 PM

numpy



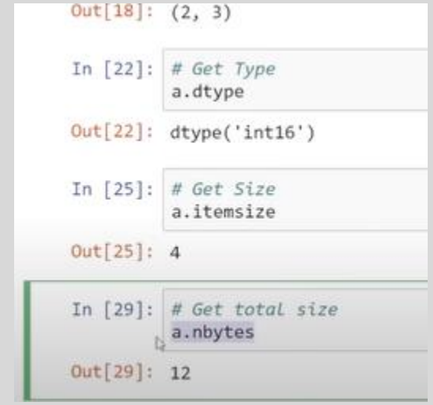
111. [11:41 PM]

we can also specify the datatype we want



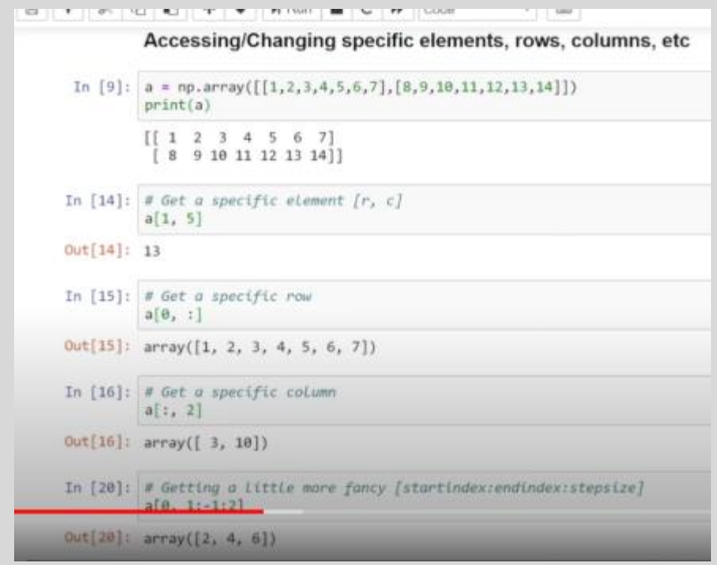
112.

UNSOLVED — 03/01/2023 11:48 PM



113.

UNSOLVED — 03/01/2023 11:58 PM



114. [11:59 PM]

```
In [15]: # Get a specific row
         a[0, :]
Out[15]: array([1, 2, 3, 4, 5, 6, 7])

In [16]: # Get a specific column
         a[:, 2]
Out[16]: array([ 3, 10])

In [20]: # Getting a little more fancy [startindex:en
         a[0, 1:-1:2]
Out[20]: array([2, 4, 6])

In [23]: a[1,5] = 20
         print(a)

         a[:,2] = 5
         print(a)

         [[ 1  2  3  4  5  6  7]
          [ 8  9 10 11 12 20 14]]
         [[ 1  2  5  4  5  6  7]
          [ 8  9  5 11 12 20 14]]

In [ ]:
```

March 2, 2023

115.

UNSOLVED — 03/02/2023 12:16 AM

```
[[ 1  2  5  4  5  6  7]
 [ 8  9  5 11 12 20 14]]
[[ 1  2  1  4  5  6  7]
 [ 8  9  2 11 12 20 14]]
```

*3-d example

```
In [25]: b = np.array([[[1,2],[3,4]],[[5,6],[7,8]]])
         print(b)

         [[[1 2]
           [3 4]]

          [[5 6]
           [7 8]]]

In [28]: # Get specific element (work outside in)
         b[:,0,:]
Out[28]: array([[1, 2],
                [5, 6]])
```

116. [12:17 AM]

```
In [40]: # All 0s matrix
         np.zeros((2,3))
Out[40]: array([[0., 0., 0.],
                [0., 0., 0.]])

In [42]: # All 1s matrix
         np.ones((4,2,2), dtype='int32')
Out[42]: array([[[1, 1],
                 [1, 1]],

                [[1, 1],
                 [1, 1]],

                [[1, 1],
                 [1, 1]],

                [[1, 1],
                 [1, 1]]])

In [43]: # Any other number
         np.full((2,2), 99)
Out[43]: array([[99, 99],
                [99, 99]])
```

117. [12:20 AM]

```
In [49]: # Any other number (full_like)
         np.full_like(a, 4)
Out[49]: array([[4, 4, 4, 4, 4, 4, 4],
                [4, 4, 4, 4, 4, 4, 4]])

In [52]: # Random decimal numbers
         np.random.rand(4,2)
Out[52]: array([[0.83329042, 0.4378868 ],
                [0.75339986, 0.12535178],
                [0.4241573 , 0.67521787],
                [0.44924988, 0.97534681]])
```

118.

UNSOLVED — 03/02/2023 12:34 AM

```
In [49]: # Any other number (full_like)
         np.full_like(a, 4)

Out[49]: array([[4, 4, 4, 4, 4, 4, 4],
                [4, 4, 4, 4, 4, 4, 4]])

In [56]: # Random decimal numbers
         np.random.rand(4,2)

Out[56]: array([[0.07805642, 0.53385716],
                [0.02494273, 0.99955252],
                [0.48588042, 0.91247437],
                [0.27779213, 0.16597751]])

In [59]: # Random Integer values
         np.random.randint(7, size=(3,3))

Out[59]: array([[5, 3, 3],
                [4, 4, 5],
                [0, 4, 0]])

In [ ]:
```

119. [12:35 AM]

```
In [73]: # Random Integer values
         np.random.randint(-4,8, size=(3,3))

Out[73]: array([[-2, -4, -4],
                [ 6,  6,  3],
                [ 3,  2,  2]])

In [76]: # The identity matrix
         np.identity(5)

Out[76]: array([[1., 0., 0., 0., 0.],
                [0., 1., 0., 0., 0.],
                [0., 0., 1., 0., 0.],
                [0., 0., 0., 1., 0.],
                [0., 0., 0., 0., 1.]])

In [ ]:
```

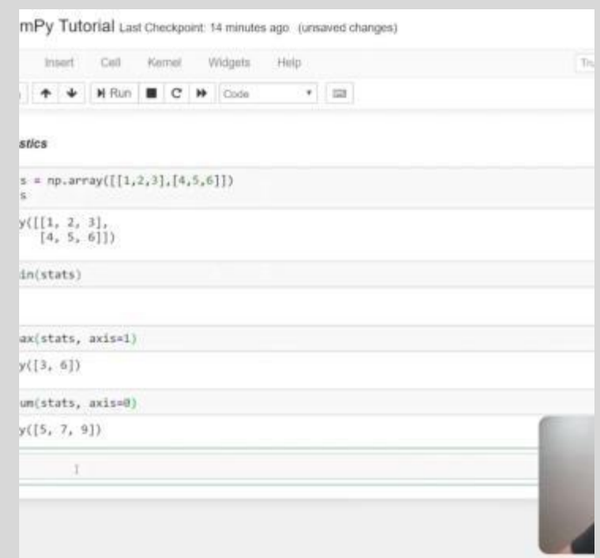
120.

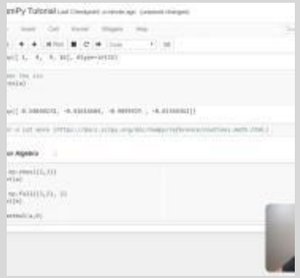
UNSOLVED — 03/02/2023 1:30 AM

```
In [88]: output = np.ones((5,5))
         print(output)

         z = np.zeros((3,3))
         z[1,1] = 9
         print(z)

         output[1:4,1:4] = z
         print(output)

         [[1. 1. 1. 1. 1.]
          [1. 1. 1. 1. 1.]
          [1. 1. 1. 1. 1.]
          [1. 1. 1. 1. 1.]
          [1. 1. 1. 1. 1.]]
         [[0. 0. 0.]
          [0. 9. 0.]
          [0. 0. 0.]]
         [[1. 1. 1. 1. 1.]
          [1. 0. 0. 0. 1.]
          [1. 0. 9. 0. 1.]
          [1. 0. 0. 0. 1.]
          [1. 1. 1. 1. 1.]]
```

121.

UNSOLVED — 03/02/2023 2:35 PM





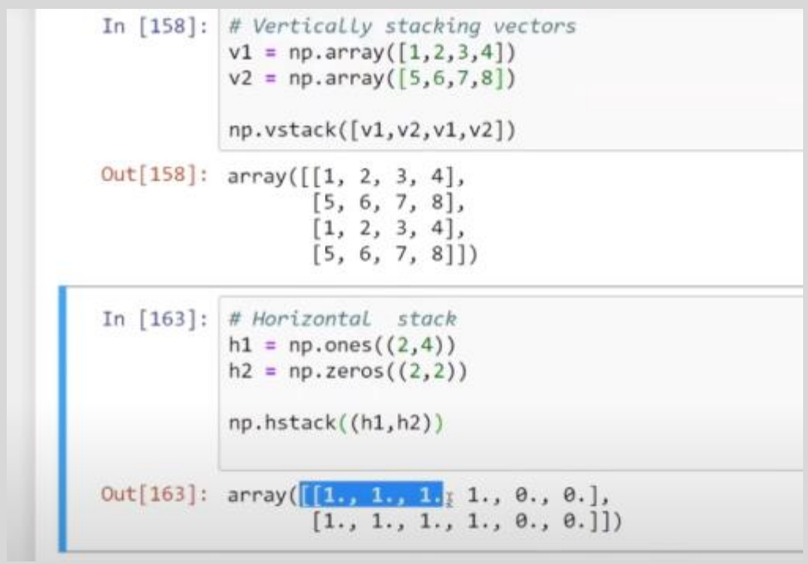
122.

UNSOLVED — 03/02/2023 3:27 PM

Reorganizing Arrays

```
In [151]: before = np.array([[1,2,3,4],[5,6,7,8]])
          print(before)

          after = before.reshape((2,3))
          print(after)

          [[1 2 3 4]
           [5 6 7 8]]
```

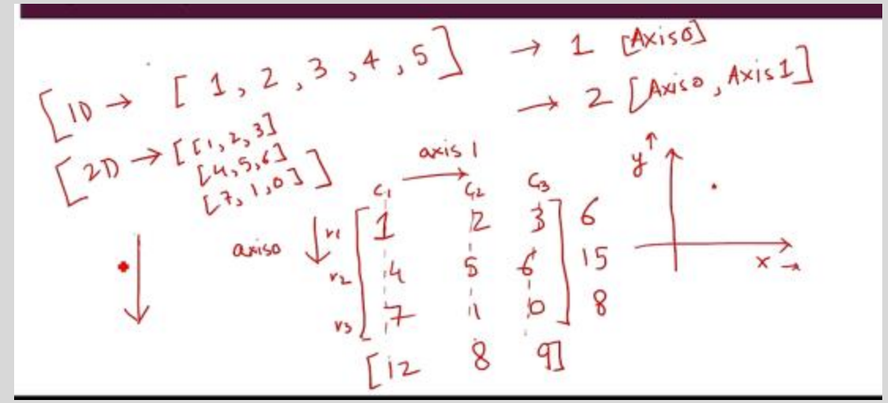
123. [3:28 PM]



```
In [158]: # Vertically stacking vectors
          v1 = np.array([1,2,3,4])
          v2 = np.array([5,6,7,8])

          np.vstack([v1,v2,v1,v2])

Out[158]: array([[1, 2, 3, 4],
                 [5, 6, 7, 8],
                 [1, 2, 3, 4],
                 [5, 6, 7, 8]])

In [163]: # Horizontal  stack
          h1 = np.ones((2,4))
          h2 = np.zeros((2,2))

          np.hstack((h1,h2))

Out[163]: array([[1., 1., 1., 1., 0., 0.],
                 [1., 1., 1., 1., 0., 0.]])
```

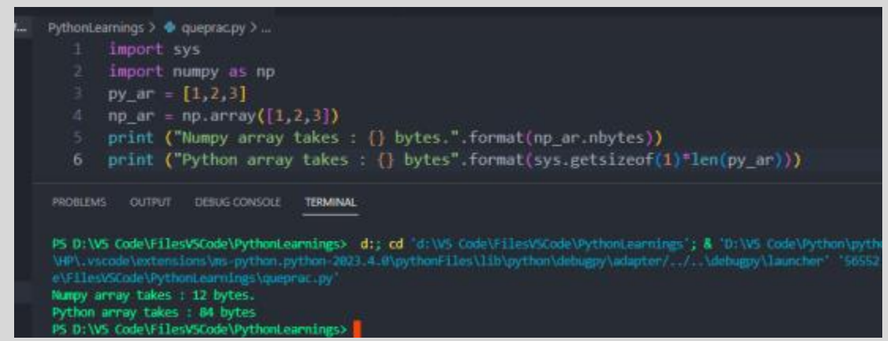
124.

UNSOLVED — 03/02/2023 3:57 PM



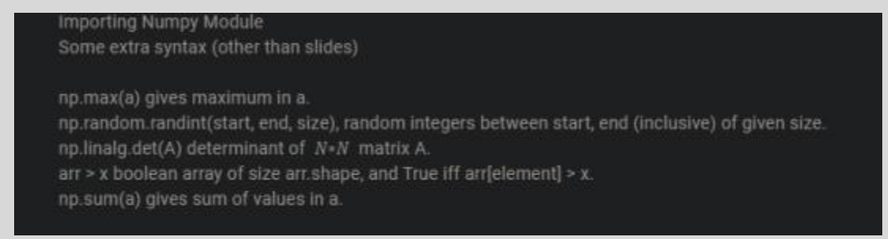
125.

UNSOLVED — 03/02/2023 4:25 PM



```
PythonLearnings > queprac.py > ...
import sys
import numpy as np
py_ar = [1,2,3]
np_ar = np.array([1,2,3])
print ("Numpy array takes : {} bytes.".format(np_ar.nbytes))
print ("Python array takes : {} bytes".format(sys.getsizeof(1)*len(py_ar)))

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

PS D:\VS Code\FilesVSCode\PythonLearnings>  d:; cd 'd:\VS Code\FilesVSCode\PythonLearnings'; & 'D:\VS Code\Python\pyth
\HP\.vscode\extensions\ms-python.python-2023.4.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '56552
e\FilesVSCode\PythonLearnings\queprac.py'
Numpy array takes : 12 bytes.
Python array takes : 84 bytes
PS D:\VS Code\FilesVSCode\PythonLearnings>
```

126. [4:26 PM]



```
Importing Numpy Module
Some extra syntax (other than slides)

np.max(a) gives maximum in a.
np.random.randint(start, end, size), random integers between start, end (inclusive) of given size.
np.linalg.det(A) determinant of N*N matrix A.
arr > x boolean array of size arr.shape, and True iff arr[element] > x.
np.sum(a) gives sum of values in a.
```

127. [4:26 PM]

Bharath B N 9:19 AM

Clarification regarding arr > x:

```
import numpy as np

arr = np.array([1,3,-2,4,5,3,0,7])

print(arr > 2)
```

output: [False True False True True True False True]

128.

UNSOLVED — 03/02/2023 11:00 PM

Some basic syntax (Apart from slides)

1. `arr.flatten()` converts **arr** into a 1D array with rows in order.
2. `arr > x` boolean array same shape, and **True** iff arr[element] > x.
3. `np.sum(a)` gives sum of values in a.
4. `np.random.randint(start, end, size)` random integers between start, end of given size.
5. `np.linalg.det(A)` determinant of $N * N$ matrix A.
6. `np.cross(x, y)` cross product between vector x and y.
7. `np.sin(x)` sin(x), `np.cos(x)` cos(x), `np.tan(x)` tan(x).
8. `np.std(x, axis=n)`, `np.median(x, axis=n)`, `np.mean(x, axis=n)`
9. `np.max(a)` gives maximum in a.
10. `np.random.randint(start, end, size)` random integers between start, end (inclusive) of given size.
11. `np.linalg.det(A)` determinant of $N \times N$ matrix A.
12. `arr > x` boolean array of size arr.shape, and **True** if `arr[element] > x`.
13. `np.sum(a)` gives sum of values in a.

March 9, 2023

129.

UNSOLVED — 03/09/2023 1:33 AM

Method Overloading

```
class Area:
        def find_area(self,a=None,b=None):
                if a!=None and b!=None:
                        print("Area of Rectangle:",(a*b))
                elif a!=None:
                        print("Area of square:",(a*a))
                else:
                        print("Nothing to find")
obj1=Area()
obj1.find_area()
obj1.find_area(10)
obj1.find_area(10,20)
```

130. [1:34 AM]

If we use same method in parent class and sub class also then if we point the object to the subclass the method of subclass will be executed this is called method overiding

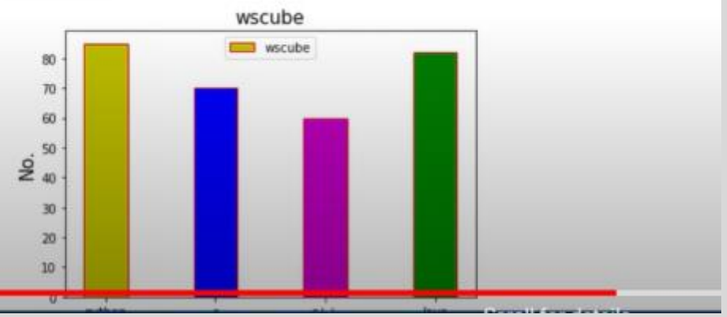
131.

UNSOLVED — 03/09/2023 3:13 PM

```
plt.xlabel("language",fontsize=15)
plt.ylabel("No.",fontsize=15)
plt.title("wscube",fontsize=15)
c=["y","b",'m',"g"]
plt.bar(x,y,width=0.4,color=c,edgecolor="r",linewidth=5,linestyle=":",alpha=0.5)
plt.show()
```

132. [3:14 PM]

```
In [24]: import matplotlib.pyplot as plt
         x=["python","c","c++","Java"]
         y=[85,70,60,82]
         plt.xlabel("language",fontsize=15)
         plt.ylabel("No.",fontsize=15)
         plt.title("wscube",fontsize=15)
         c=["y","b",'m',"g"]
         plt.bar(x,y,width=0.4,color=c,edgecolor="r",label="wscube")
         plt.legend()
         plt.show()
```



133.

UNSOLVED — 03/09/2023 3:20 PM

```
In [35]: import matplotlib.pyplot as plt
         import numpy as np

         x=["python","c","c++","Java"]
         y=[85,70,60,82]
         z=[20,30,40,50]

         width=0.2
         p=np.arange(len(x))
         p1=[j+width for j in p]

         plt.xlabel("language",fontsize=15)
         plt.ylabel("No.",fontsize=15)
         plt.title("wscube",fontsize=15)

         plt.bar(p,y,width,color="r",label="popularity")
         plt.bar(p1,z,width,color="y",label="popularity1")

         plt.xticks(p+width/2,x,rotation=20)
         plt.legend()
         plt.show()
```

134. [3:27 PM]

- Want a **change color**?
- Example plot:
  - Plots f(x) = x^2 for -5 <= x <= 5

```
x = np.linspace(-5,5, 100)
plt.plot(x, x**2,'g-')
plt.show()
```

135.

UNSOLVED — 03/09/2023 3:38 PM

```
In [15]: import matplotlib.pyplot as plt
         day=[1,2,3,4,5,6,7]
         no=[2,3,1,4,5,3,6]
         colors=["r","y","g","b","r",'g',"r"]
         sizes=[400,200,400,300,200,100,600]
         plt.scatter(day,no,c=colors,s=sizes)
         plt.title("Scatter Plot",fontsize=15)
         plt.xlabel("Day",fontsize=15)
         plt.ylabel("No.",fontsize=15)
         plt.show()
```

136. [3:40 PM]

```
In [24]: import matplotlib.pyplot as plt
         day=[1,2,3,4,5,6,7]
         no=[2,3,1,4,5,3,6]
         colors=["r","y","g","b","r",'g',"r"]
         sizes=[400,200,400,300,200,100,600]
         plt.scatter(day,no,c=colors,s=sizes,marker="*",edgecolor="g",linewidth=2)
         plt.title("Scatter Plot",fontsize=15)
         plt.xlabel("Day",fontsize=15)
         plt.ylabel("No.",fontsize=15)
         plt.show()
```

137.

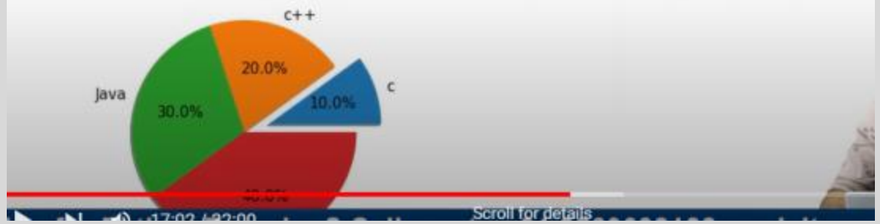
UNSOLVED — 03/09/2023 3:48 PM

```
In [36]: import matplotlib.pyplot as plt
         day=[1,2,3,4,5,6,7]
         no=[2,3,1,4,5,3,6]
         no2=[3,2,4,5,1,6,2]
         colors=[10,49,30,29,56,20,30]
         sizes=[400,200,400,300,200,100,600]
         plt.scatter(day,no,c=colors,s=sizes,cmap="viridis",alpha=0.5)
         plt.scatter(day,no2,color="r",s=sizes,alpha=0.5)
         t=plt.colorbar()
         t.set_label("ColorBar",fontsize=15)
         plt.title("Scatter Plot",fontsize=15)
         plt.xlabel("Day",fontsize=15)
         plt.ylabel("No.",fontsize=15)
         plt.show()
```

138.

UNSOLVED — 03/09/2023 4:01 PM

```
In [37]: import matplotlib.pyplot as plt

         x=[10,20,30,40]
         y=["c","c++","Java","python"]
         ex=[0.3,0.0,0.0,0.0]
         c=["r","b","g","y"]

         plt.pie(x,labels=y,explode=ex,autopct="%0.1f%%",shadow=True,radius=1.3,labeldistance=1.1,startangle=0
                 ,textprops={"fontsize":15})


         plt.show()
```



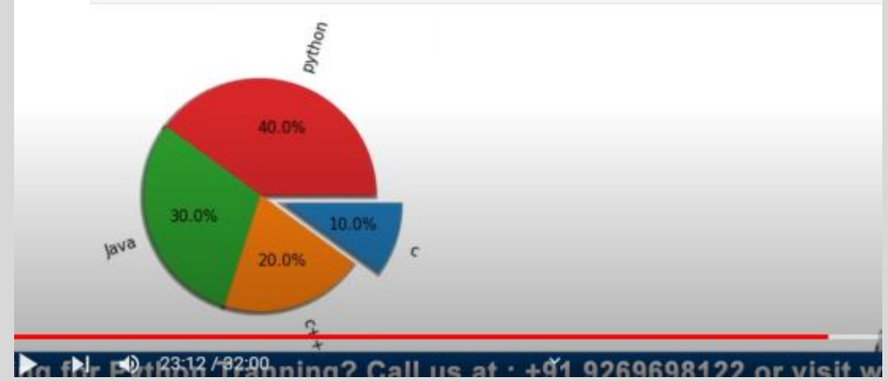
139.

UNSOLVED — 03/09/2023 10:15 PM

```
plt.pie(x,labels=y,explode=ex,autopct="%0.1f%%",shadow=True,radius=1.3,labeldistance=1.1,startangle=0
        ,textprops={"fontsize":15},counterclock=False,wedgeprops={"linewidth":5,"edgecolor":"m"},
        center=(3,6))


plt.show()
```

140. [10:15 PM]

```
plt.pie(x,labels=y,explode=ex,autopct="%0.1f%%",shadow=True,radius=1.3,labeldistance=1.1,startangle=
        ,textprops={"fontsize":15},counterclock=False,wedgeprops={"linewidth":5},rotatelabels=True)


plt.show()
```



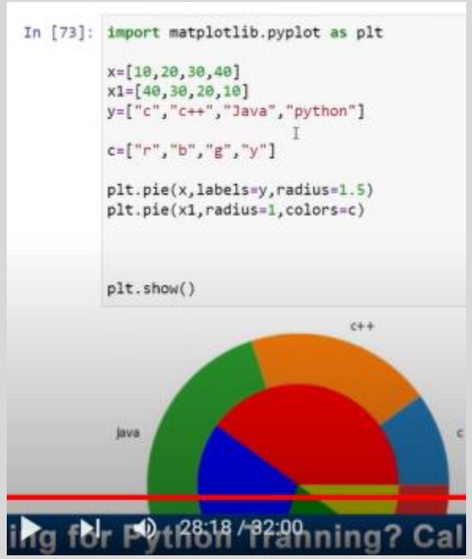
141. [10:18 PM]

```
import matplotlib.pyplot as plt

x=[10,20,30,40]
y=["c","c++","Java","python"]
ex=[0.3,0.0,0.0,0.0]
c=["r","b","g","y"]

plt.pie(x,labels=y,explode=ex,autopct="%0.1f%%",shadow=True,radius=1,labeldistance=1.1,startangle=0
        ,textprops={"fontsize":15},counterclock=False,wedgeprops={"linewidth":5},rotatelabels=False)


plt.title("Wscube Tech")
plt.legend(loc=2)
plt.show()
```

142. [10:18 PM]

dot pie chart

143. [10:19 PM]



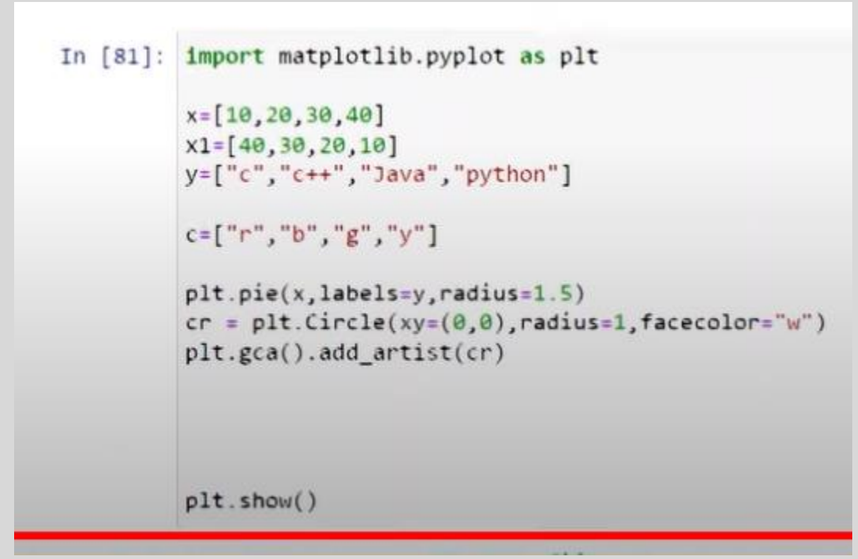
144. [10:20 PM]

To make a ring

145. [10:20 PM]

146. [10:20 PM]

or this



147.

UNSOLVED — 03/09/2023 11:51 PM

-`` : blue solid line
--`` : green dashed line
^`` : red triangles
D`` : cyan diamonds
>`` : magenta filled triangles pointing right
X`` : yellow X's

b	blue	T	T
g	green	s	square
r	red	d	diamond
c	cyan	v	triangle (down)
m	magenta	^	triangle (up)
y	yellow	<	triangle (left)

symbol	description	symbol	description
k	black	>	triangle (right)
w	white	p	pentagram
.	point	h	hexagram
o	circle	-	solid
x	x-mark	:	dotted
+	plus	-.	dashdot

March 14, 2023

148.

UNSOLVED — 03/14/2023 2:32 PM

that commented code is without using walrus operator and the uncommented code is by using the walrus operator

```python
# foods = list()
# while True:
#   food = input("What food do you like?: ")
#   if food == "quit":
#       break
#   foods.append(food)
foods = list()
while (food := input("What food do you like?: ")) != "quit":
    foods.append(food)
```

149.

UNSOLVED — 03/14/2023 3:50 PM

```python
import re
pattern = "was"
text = '''Intense Tropical Cyclone Dumazile was a strong
    tropical cyclone that brought flooding to Madagascar
    and Réunion in early March 2018. Dumazile originated
    from an area of low pressure that formed in the South-West
    Indian Ocean near Agaléga on 27 February. The system
    concentrated into a tropical disturbance on 2 March
    and was named the next day, as it intensified into a
    tropical storm. Amid conditions conducive for intensification,
    Dumazile strengthened over the next two days and reached
    peak intensity on 5 March as an intense tropical cyclone,
    with 10-minute sustained winds of 165 km/h (105 mph),
    1-minute sustained winds of 205 km/h (125 mph), and a
    central pressure of 945 hPa (27.91 inHg). The system
    weakened steadily over the next couple days because
    of increasing wind shear as it tracked to the southeast.
    Dumazile became post-tropical on 7 March'''

print (text.index(pattern))
# print (len('''Intense Tropical Cyclone Dumazile'''))
print (text.find(pattern))
print (re.search(pattern,text))
```

March 16, 2023

150.

UNSOLVED — 03/16/2023 5:20 PM

Sure, I will try to explain more clearly.

xlim is a function that controls the range of the x-axis on your plot. You can use it to zoom in or out on your data. For example, if you have data from 0 to 100 on the x-axis, but you only want to see the data from 20 to 80, you can use xlim(20, 80) to change the range of the x-axis.

xrange is a function that creates a sequence of numbers that you can use as data for plotting. For example, if you want to plot a line that goes from (0, 0) to (9, 9), you can use xrange(10) to create a sequence of numbers from 0 to 9 for both the x and y values.

Does this make sense?