Python and AI Power-Up Program Online Classroom-20250813_110127-Meeting Recording

August 13, 2025, 5:31AM

1h 34m 28s

Ajay Patel started transcription

Tarun Jain 0:10

OK.

OK.

This.

Yes.

OK.

Ajay Patel 2:48

Yeah, Tarun, we should start the session. Hardeep will join in 5 minutes, so no worries.

Tarun Jain 2:56

Yeah, sure. Just one second.

Yeah, I'll show my screen.

Tarun Jain 3:23

Sorry by mistake I clicked on leave instead of share.

Ajay Patel 3:25

Yeah, yeah, that's OK.

Tarun Jain 3:33

Uh, is my screen visible?

Ajay Patel 3:36

Yeah, your screen is visible.

RamKrishna Bhatt 3:37

Yes.

TJ **Tarun Jain** 3:37

Yes.

TJ **Tarun Jain** 3:38

OK, So what we'll do is we'll start off with the recap of what we did yesterday and then probably what we will do today is we'll try to cover and complete sets function and what is the scope under function and then the exception block. So this is pretty straightforward the exception.

Ajay Patel 3:42

Mm-hmm.

TJ **Tarun Jain** 3:57

Accept functions is something that we will explore more and since we already know most of tuple and as well as dictionary commands, it will be easy for us to get started with set as well. To begin with, yesterday we covered the list, so I'll just open the comment that we had on list.

Is it? Yeah. So basically in list what we did yesterday was we started off with indexing. So indexing is basically you have the list of components and if you want to extract any value from given element you just have to give the integer.

Right. And now second thing was we looked into the slicing part. One core concept of slicing is you need to have start and then the end. If you don't have end you can just leave it empty and if you don't have start you can give some value and leave it empty.

And the second thing was if you use slicing for string, the output will also be string. If you use slicing for list, the output will also be list. And if you use slicing for tuple, the output will also be tuple. So that is few things we need to remember.

And then we explored some of the functions that we had. So we looked into append. So append basically will add a new element at the last index which is -1. And another thing is this will return none right? You can't assign it to any variable.

Pop basically will remove. If you keep it empty then it will remove it from the minus

month index. If you want to remove specific index, you can also mention that particular index value, right? So once you mention pop of index, it will remove the value from that particular index.
So when I say value, it's nothing but the element.
Right. And then basically append will just add new one element. But if you want to add multiple elements in the given list, what you can do is you can use extend. So this is basically you merge to list right? And then we have copy. So copy whenever you are working with text preprocessing.

Ajay Patel 5:54
OK.

TJ **Tarun Jain** 6:05
Where you will manipulate your original list during that time. It's always a good practice to create the copy of it so that you can use it for evals or even compare it later. And then we have sort again sort. It's very simple understanding, but one thing what we have to notice is it will return none.
Same goes with reverse. This will also return none and then we have remove. So when we use remove we have to use exact value or exact element that is present in our list. So if you don't mention the exact value you will get the type error.
And then we have delete. So delete basically will remove that particular variable existence. So it will also delete from the memory component and as an alternative we also have clear. So clear it basically it will just remove all the elements, but the memory usage of that variable will be there. So does anyone have any questions? This.
Oh, methods as anyone explored.

Mitesh Rathod 7:08
Uh, not from IC.

TJ **Tarun Jain** 7:10
OK, so then we also explored one of the alternative on how you can use list comprehension. So probably I'll just show that example. So there are two syntax that we need to keep in mind whenever we are starting with list comprehension. The first thing is.

We have to start with the brackets, then write your for logic and using this for logic you will have one variable that you need to manipulate that variable how you need to use that you have to write on the left hand side. So this is 1 and if in case you have conditional based list comprehension what you can do is.
You can write for logic then you can add if logic. Once you add the if logic then you can add that particular print statement on the left hand side. So whatever you add on the left hand side is your final element. For example if you see here we look through all the 10 values and after we look all the 10 values we are checking the square root of.
Right. So if you notice we go from zero to 9 and then we have this square value. So whatever you have on the left hand side is your final set or final element that you have in your list. So let's quickly look into some of the quiz questions that I have. So can anyone tell me the answer of this?

RamKrishna Bhatt 8:25
First, we'll return in.

Ajay Patel 8:29
Orange will be appended on the fruits.

TJ

Tarun Jain 8:31
OK, it will be added after Mango and 2nd will be Mango.

Ajay Patel 8:34
012 and Mango will be print.

Mitesh Rathod 8:35
Banana.
Mango.

Hardip Patel 8:38
Edney will be none.

TJ

Tarun Jain 8:38

So what will this be printed?
This will be this.

Hardip Patel 8:42
Uh, none. None.

Ajay Patel 8:44
None.

Mitesh Rathod 8:48
Mango.

Tarun Jain 8:48
What if I make this as?

Mitesh Rathod 8:51
I think, you know, it would realize it will be true.

Tarun Jain 8:54
Capitalize dot starts with.
And.

Ajay Patel 9:01
Mm.

Hardip Patel 9:03
True.

RamKrishna Bhatt 9:04
2.

Mitesh Rathod 9:04
I think I know the solution.

Tarun Jain 9:08

So this is mango. Then we have capitalize then starts with M which is capital M mango which is true and add new is empty right? Because you can't assign append to any variable. Now can anyone tell me the answer of this?

Ajay Patel 9:14
Oh.

Hardip Patel 9:16
OK.

Mitesh Rathod 9:17
Yep.
Yeah, it will remain same.

RamKrishna Bhatt 9:28
9521.

TJ

Tarun Jain 9:30
This will be.

Mitesh Rathod 9:33
Uh.

RamKrishna Bhatt 9:34
This will be 1259 and 95.

Mitesh Rathod 9:38
Alright.

TJ

Tarun Jain 9:40
Now this will be.

Margi Varmora 9:40
Yeah.

RamKrishna Bhatt 9:42

95.

Ayush Makwana 9:42

9521.

Tarun Jain 9:47

OK, now this one. OK, let me summarize what we did in dictionary. Same goes for dictionary as well. Once we completed list, list again is dynamic collection and then we have tuple which is a fixed collection. So tuple basically is immutable whereas list is mutable. Now where do we use?

Tuple actually in most of the existing frameworks that we have or existing library, you will have some shapes, right? Either it can be dimensions or it can be shape or it can be size. All these dimension, shape and size will be saved in a tuple so that you can decouple it later.

How do we decouple it? So there are two syntax on how we can decouple. The first approach is you can use index and then you can decouple it and the other option was you use everything in a single line. Do we have that example?

OK, I'll just write it below this. So usually we had width, then height, channel, then image shape. So image shape basically has 512 cross 512 cross three. Now this 512 will be your width, height will be 512 and channel will be 3. So if this is.

Hardip Patel 10:50

Mm.

Tarun Jain 11:03

Three, then you should only assign three. Let's suppose you don't know what is the final one. You have 512, you have 512, you have four. Now if it is video, you will have four. In that case you can't assign it as channel, it should be something else. It should be frame. So during that time, if you are confused, you can just leave it empty.

But The thing is, if you're if you're decoupling it, you have to have all the exact same number of variables. You can't have different. So if I leave this empty, it will also work. So if I run this.

So this empty whatever you see right under score that is not assigning any variable,

that is just a placeholder.
Uh, did you see this? Now you have with.

Hardip Patel 11:47
Yes.

Tarun Jain 11:49
You have height and last variable you are unknown so you are just keeping it empty. In most of the times what will happen is I was looking into the audio examples. So for audio examples you will have around 5:00 to 6:00 elements sometimes and not every time you need all the five variables.
Now let's suppose you're resizing the image right? When you resize the image, the channel will be same. Initially also you had 512 cross 512 cross three. Now when you resize, you're just making it 256 cross 256 cross three. So the three remains the same, so you don't have to assign this variable. So there might.
So there might be times when you have to use shape and if you want to assign it in a variable, you might not need some of the variables. So just to utilize less memory you can leave it empty, right? And then we looked into dictionary which is key value mapping. Basically you have key.
Then you have value which is separated by colon. This is similar to what we see in Jason or object and once we do that if we need to add new element we can either directly add by prod price, add that particular new key and value and some of the commands that we saw was.
We had clear. Clear again is similar to what we saw in list. If you define any key value pairs and if you want to empty your dictionary you can use clear. Then you have delete which will remove the entire variable and for the best practice it's always used to. It's better to use GC which is garbage collector.
And then we looked into keys, which is a list. Then we have values, then we have items. Item is also a list, but every single list will have your key value inside a tuple. This is mainly used for looping and then you have get get. It's the best way to extract.
The value for the given key. If the key is not present then you can have a alternative message and then we have update. So if you want to update the price or any value of the key then you can use update and then copies again similar to what we saw in the

list.

OK, so can anyone tell me the answer of this note?

Mitesh Rathod 14:05

So first it will update to 75,000.

RamKrishna Bhatt 14:07

Did me the picture thing.

Tarun Jain 14:10

Yeah, this one.

Mitesh Rathod 14:13

OK, the keyboard are not available.

Ayush Makwana 14:15

Not available.

RamKrishna Bhatt 14:16

Not available.

Tarun Jain 14:21

What if I use this thing rod price keyboard?

Ajay Patel 14:29

None.

Tarun Jain 14:31

Will this be or will it be error?

Ajay Patel 14:34

Sorry, no.

Tarun Jain 14:36

Uh, does anyone remember the other name?

Mitesh Rathod 14:42
Oh, we have to do something.

Tarun Jain 14:46
Can anyone repeat again? It was not audible.

Mitesh Rathod 14:47
PC.
X not exists or something not it it is something regarding X what you're doing.

Tarun Jain 14:56
So what is this thing called? Let's suppose you have laptop mouse. What are these two things? What is it right? So you'll get the error.

Mitesh Rathod 15:03
Is that keys is key?

Margi Varmora 15:05
E-mail phone.

Tarun Jain 15:10
So now I'll just print this. You have 75,000. Now when you use get get it has to get the value of the given key. But I don't have keyboard as a key in my given product price so it will print as not available.
But if I use product price and then directly if I use keyboard then it will throw an key error. Why? Because that key doesn't exist.
OK, now let's proceed with the next question.
Oh, So what will be the output of this?

Ajay Patel 15:47
6 result.

Mitesh Rathod 15:48
It will be like a every power of.

Margi Varmora 15:50

Five in 30s.

Ajay Patel 15:52

Eight. No, no, no, 4/16/36.

TJ

Tarun Jain 15:58

Oh, can you repeat?

Ajay Patel 15:59

41636 of 4 + 1537 and 37 no no 517 and 37.

RamKrishna Bhatt 16:02

OK.

Hardip Patel 16:04

517 and 37.

TJ

Tarun Jain 16:09

OK, yeah, we had this plus one.

Ajay Patel 16:13

Yeah, that's good.

TJ

Tarun Jain 16:14

OK, so now this is a tuple and then what I'm trying to do is I'm making it mutable. So I guess we looked into this example yesterday.

Hardip Patel 16:29

AB and XYZ.

RamKrishna Bhatt 16:29

Yes.

Tarun Jain 16:32
AB and XYZ. So why? Why can we do this? Because even though this is a tuple, inside this tuple we have the element as a list. So list is a mutable right? So I'm just using data one up and Z.

Ajay Patel 16:33
XYZ.

Hardip Patel 16:33
XYZ.

Ajay Patel 16:39
Mm.

Tarun Jain 16:49
So this is the first index and when I do append Z will be added at the end. So this is the final example. So can anyone tell the output of this?

Hardip Patel 17:04
Uh, pen bag, pen bag.

Margi Varmora 17:04
I will.

RamKrishna Bhatt 17:05
A cartwheel printed.

Ajay Patel 17:07
Pen and beg.

Tarun Jain 17:10
So what will be this card card?

Hardip Patel 17:10
Both will be pan and back.

Ajay Patel 17:10
No.

RamKrishna Bhatt 17:12
But as it is.

Margi Varmora 17:15
Cartoon same copy card will remove it.

Ajay Patel 17:15
No.

Ishan Chavda 17:16
Then moved in.

RamKrishna Bhatt 17:20
Don't be scare me.

Ajay Patel 17:21
Back.

Hardip Patel 17:23
And yeah, that one as well.
I.

Tarun Jain 17:28
Cool. OK, so now what we'll do is we'll proceed with the set.

Ajay Patel 17:29
Mm.

Hardip Patel 17:31
Yeah.

Tarun Jain 17:37
So for set basically what we can do is it's similar thing. I'll just use one variable called programming language.

Hardip Patel 17:43
So the so for the for the last quiz the if we use copy in the second line it will be then.

Tarun Jain 17:55
It's the same thing.

Ajay Patel 17:55
Start dot copy then it will be then pen book.

Hardip Patel 17:57
Oh.

Tarun Jain 17:59
You can use copy as well. Yeah, in most of the case it will be same.

Hardip Patel 17:59
Would it be the same thing?
But wouldn't the card change it to pen book and bag?

Ajay Patel 18:09
Oh.

Tarun Jain 18:09
For which one?

Hardip Patel 18:11
The so the now the print card will be pen, book and bag, right?

RamKrishna Bhatt 18:12
Oh.

Tarun Jain 18:13
OK, OK, OK. So basically if you use copy probably here you will have pen, bag and book.

Hardip Patel 18:21
Yeah. OK. OK. Just for understanding. Right. OK. OK. OK. OK. Perfect. OK. OK.

Tarun Jain 18:24
So if I run this you will have pen, book and bag if you use copy.

RamKrishna Bhatt 18:25
Yes.
Thanks for the my question.

Ajay Patel 18:29
M.

Tarun Jain 18:33
If you don't use copy, it's like just repeat. Yeah, if I use the ID of this ID of cart.

Hardip Patel 18:36
The same reference, same ID, yeah.

Tarun Jain 18:45
And then if I use.
ID of copy part it's the same 77 but if I use copy, I use copy.

Hardip Patel 18:52
It would be different.

Ajay Patel 18:55

Same.

Hardip Patel 18:55

Oh.

Ajay Patel 18:56

Hmm.

Hardip Patel 18:57

So, OK, OK, so if we just copy, then it'll be different, OK.

Ajay Patel 19:03

If it is copy then and then it will assign a new memory.

TJ

Tarun Jain 19:04

It is 824 and this is 200.

Hardip Patel 19:07

Right, OK.

TJ

Tarun Jain 19:10

Cool.

Hardip Patel 19:10

So just one follow up question. So the copy will be deep clone. So let's say if instead of book there will be another another set or couple or whatever.

Ajay Patel 19:27

Complex data structure.

Hardip Patel 19:27

It will also be.

Tarun Jain 19:30
4.

Hardip Patel 19:33
So let's say like if a card has pan book and back currently, but if what if it has an array inside? I mean like it could be anything, would it be cloned as well?

Tarun Jain 19:37
OK.
Like this.

Hardip Patel 19:47
Yeah.
Would it be?

Tarun Jain 19:49
Yeah, this will close it.

Hardip Patel 19:51
OK, it will be deep clone, OK.

Tarun Jain 19:54
Oh, what do you mean by deep clone?

Hardip Patel 19:56
I mean like a would the idea of a array of book remain the same?

Tarun Jain 20:04
No. So this is just an element, right? The ID is mainly for variable, not for elements.

Hardip Patel 20:06
OK.
OK. I mean like for that element, it will be cloned as well. Is it something like that?

OK.
Mm.

Tarun Jain 20:19
I actually didn't get clone, actually clone in the sense.

Hardip Patel 20:22
OK, OK, OK, not clone, I mean like copy. So the values will be copied or just the reference to the book array will be copied. OK, OK, so it is a deep clone, a deep copy.

Tarun Jain 20:32
Values, values, not reference.

Hardip Patel 20:37
OK. OK. OK. Thank you.

Tarun Jain 20:38
Yeah.
OK, so as of now when we saw list we had square bracket.
And we had parenthesis for tuple and for list we had sorry for dictionary we had curly brackets but every single key had colon and then value. So for set also we will be using curly brackets itself. So here I.
I'll just define programming languages.
And what I'll do is I will define Python.
Then I'll define Java.
PHP go and then again I'll define Python.
And what I'll do is I'll copy the same thing and instead of programming language here I'll make it list.
And I'll just add square bracket. So how do I check the data type?

RamKrishna Bhatt 21:45
Alright.

Tarun Jain 21:46
Hello yeah just by adding type. So this is set and if I use list this is list. But now if you

notice one thing, whenever you're using set rate, no matter how many values you add, if there is any duplication it will only consider one of it. So now.

Ajay Patel 21:49
OK.

Tarun Jain 22:05
So for example if I have Python, Java, PHP, Go, Python, if I do the length of programming language it will exclude one of the Python. So it will be Python, Java, PHP, Go which is 4. OK if I use list and then I if I check the length.
It is fine. Why? Because it will also consider Python.
So you understood the example. I'll just print both of them.

Hardip Patel 22:31
Yes.

Tarun Jain 22:34
So basically what will happen in set is in set, no matter how many values you had, it will only consider the unique values or unique elements. So in our case the unique elements is Python and now if you notice this is not in.
The order that we added. So I added as Python, Java, then PHP, then Go. But the output that you are getting is Go, then Java, then PHP, then Python. So The thing is set doesn't support indexing, so if you do programming languages of 0 you'll get an error.
Set doesn't.
Have induction.
So now what I'll do is if you want to add new elements, what I'll just use methods. Methods in set. So everyone remember some concept called union, right? Union intersection. So can you recall what is union? Let's suppose I have set A.
Set A and in set AI have 123. Then I have set B. In set B I have 3-4 and five. So what do you think is union union of?

Ajay Patel 23:56
Union will be 12345.

Hardip Patel 23:57
Um.
Yeah.

Tarun Jain 24:01
123445 correct. So if I use set TA dot.
Union ZB.
You will get 12345. What about intersection?

Ajay Patel 24:15
3.

RamKrishna Bhatt 24:16
Only three.

Tarun Jain 24:17
Just three. So these are the two comments. One is union.
All the unique.
Or all the elements.
From the to list.
And then we have intersection.
Only the unique elements.
That occurs.
In both the set.
And then we also have add comment. Let's suppose I want to add a new element in this set TA dot I have add, I'll add 5.
Now set A will become 12345. Again, this thing whatever you have here is none. You can't return. I mean you can't assign set A dot add five in any variable. So this is none and we also have.
Remove command. I'll just write set T then we have pop. So which element do you think will be removed?

Hardip Patel 25:34
5.

RamKrishna Bhatt 25:34

5.

Ajay Patel 25:35

Last.

Tarun Jain 25:38

Now if I just print get value one, can anyone tell me why it is 1?

Ajay Patel 25:46

Q. Does it follow a principle of Q?

Tarun Jain 25:50

So basically when it comes to this thing right set, whichever is added first will be removed first.

Ajay Patel 25:58

Hmm, first in, first out.

Tarun Jain 25:59

So is this a stack or is it a queue? It's a stack, right?

Ajay Patel 26:01

QQ. It's a queue. It's a queue. It's a queue. First in, first out and leave for here.

Mitesh Rathod 26:02

Is this?

Hardip Patel 26:02

It is a cue. It is a cue. Stack removes from the last.

Tarun Jain 26:04

OK, so it was.

Now if I run this again, it will remote too.

Ajay Patel 26:11

Hmm.

Tarun Jain 26:12

So we have had.
Which will add the new value, so there is no indexing.
It's like first go. You can just tell queue. Same goes for pop as well.
Whichever is added first.
Please remove first.
What else do we have? I'll just what is the command to check all the methods that we have? Let's suppose I have this variable which is set TA and I want whatever functions it supports.

Hardip Patel 26:54

DIR.

Mitesh Rathod 26:57

OK.

Tarun Jain 26:58

We have BAIER and you can just add the variable.
So add we already saw clear and copy is similar. Then intersection is there which is important update. It's also straightforward. I'll probably use discard.
Set TA dot discard.
And here what you have to do is you have to give element. This is similar to remove.

Ajay Patel 27:28

Mm.

Tarun Jain 27:30

Similar to remove where the parameter needs to be exact value given.
In the set. So here I'll just add what values we have. We just have three and five if I'm not wrong. If I print set here, we just have three and five. Why? Because we already removed one and two. Here. Now what I'll do is I'll just add 3.

And now if I print set here, this is 1. So based on all these examples, is set mutable or immutable?

Mitesh Rathod 28:11

We are tables.

We need to work.

Hardip Patel 28:21

Mutable.

Tarun Jain 28:22

So set basically is mutable.

So mostly when it comes to set you will only use these two commands. Either it will be intersection or union and apart from that all other commands is the right. Since it is similar to what you see in list, you can use list as an alternative, but whatever logic is the right in intersection and union that logic can be also.

To be replicated by list. So the only thing is you'll have to add your own custom code or your own custom logic where either you can use some kind of for loop and check whether this element is present or not and then you can just execute it. But in most of the cases you will never encounter list in many of the examples, right? Either it will. Will be list or it will be dictionary and if you are using any frameworks then it is tough. But it is always good to have some understanding of what are the basic data type. Just to summarize, since we completed the data types, all the data types that you've seen earlier, the common data types are.

We have numbers, we have list, sorry, we have numbers, then we have float, then we have string and then we have boolean and then we also have something called as collection. So collection data type is where you have list.

Mitesh Rathod 29:52

List.

Tarun Jain 29:52

Which is dynamic.

Mitesh Rathod 29:54

OK.
Yes.

Tarun Jain 29:56
And then you have couple.
Which is fixed.

Mitesh Rathod 30:02
Fixed.

Tarun Jain 30:02
And then you have dictionary which is mainly key value pair again which is dynamic.

Mitesh Rathod 30:05
M.

Tarun Jain 30:10
And then you have set which is dynamic but unique.
I hope this is clear, right? Most of the commands, whatever you saw it pop and this thing, it's similar to what you saw in list.

Hardip Patel 30:28
Yes.

Tarun Jain 30:31
Cool. So we'll get started with functions. Does anyone recall the keyword that we use for function? Probably this example before.

Mitesh Rathod 30:41
Next.

Tarun Jain 30:46
DF correct. So so far what commands are we used before is conditional statements. Let's suppose we have if condition, then we have elif, then we have else, then we had while loop and then we had for loop. So far if you notice the syntax of all.

These things you start with for then you add I in range and then you give colon. Whenever you encounter colon keyword at the end of any conditional statement and if you hit enter you might have seen two spaces right? So this is indentation.
For same whenever you are working with functions you have to define a keyword called def and after def you need to give a function name. In our case I'll just add greet for timing and now what I'll do is def greet you can add parenthesis.
And once you have parenthesis, just hit colon and enter. It will be an indentation. So this is a simple.
Uh, I'll just add print.
Welcome. So this is how you define a function. You have def, then you have the function name, then you have parenthesis. Once you have parenthesis you need to end with colon and once you end with colon you have to define your conditional statements or any logic within that particular function. But The thing is.
If you're using Collab you will just have two spaces, but if you're using IDE it is poor. For ID it is 4 spaces and this is ideal even in Colab. I'll always recommend once you enter it just add two more spaces but you won't get any error if you print this and now if you want to call you can just add greet.
Now I'll show one more example. This is print statement. What if I return something? I'll just return our name.
Let's suppose I add a name. Now this is nothing but a argument, or you can also call it as parameter. Now I'll just run this now greet. I'll just tell Tarun. Do you think this return will print anything if I just run this particular statement?

Ishan Chavda 33:02
No, it is.

Tarun Jain 33:02
In PHP also we have return rate or is it different?

Hardip Patel 33:06
Yeah, there is.

Mitesh Rathod 33:06
Yes, it is the same.

Tarun Jain 33:09
Oh, what?


Mitesh Rathod 33:10
It is the same. It will be written. It's written.

Tarun Jain 33:12
OK, so basically when you return any value it is supposed to be called inside a variable. Now if I just print greet Tarun it will print OK this collab thing is worst, but The thing is whatever this value is the right name.


Hardip Patel 33:23
Yeah.

Tarun Jain 33:28
Tarun will not be printed if in case you print this in VS code. Let me just add VS code. M.
OK, so now if I print this, I'll just print Python.
So if you see it only shows welcome, it's not printing the particular name. If in case you want to print this name, what you can do is you can just define a variable which is name equals to greet Tarun. Why? Because it's returning name and then what you can do is you can just print name.
So now anyone can tell me what will be the output of this?


Mitesh Rathod 34:36
Welcome to.

Tarun Jain 34:38
It will be welcome and Tarun in the next line, but if I do return here.
You can only define return once. Whatever is there after return, it's excluded. Now if I run this, it will just write welcome.
So any statements after return will be excluded. Does it make sense?
So I'll just revise the syntax for function is very straightforward. We start with the def, then add the function name. If we have arguments, we have to give arguments and

once we give arguments, the print statement will be executed every time you call the function. But if you want to return any value which has to be reused.

Hardip Patel 35:01
Yes.

Tarun Jain 35:21
Somewhere in your entire code. Then what you can do is you can save it or return in a return statement and whenever you use return statement you can't have any other logic after this. Let's suppose if you give A equals to 10 this A as a no value. So let's suppose I do return of A.
And if I do greet it will only print welcome Tarun. So basically since we are using collaborate, it's printing Tarun which should not print. So whatever you have here, yay equals to 10 and then return yay will be excluded. But if I remove this.

Hardip Patel 35:51
OK.

Tarun Jain 35:54
Now can you tell me what will be the output?

Hardip Patel 35:57
Mhm.
10.

Mitesh Rathod 36:00
Well cooked then.

Ishan Chavda 36:00
And.

Tarun Jain 36:01
It will be welcome and then it will be 10. But usually if you want to print this time, you need to assign it in a variable.
Does this make sense?

Hardip Patel 36:15
Yeah.

Tarun Jain 36:16
OK, let me quickly open some other examples.

RamKrishna Bhatt 36:16
Yes.

Tarun Jain 36:21
Just a second.
All right. Uh, sorry.
So now what we will do is we have to cover three major things in function.
One is the usage of parameter on how you differently you can use parameters. What are the different ways to define the parameters?
And second, it's the usage of ERGS and KWRGS. So in most of the Python frameworks what you will do is you will see this values. One is ERGS which is nothing but arguments.
And then you have KWRGS which is mainly called as keyword arguments.
So this is mainly used.
While.
Defining open source frameworks.
Not just open source frameworks, but any Python frameworks. Now let me give you one example. So as everyone of you use Open AI API at least once.
Open AI or either Gemini. So if you noticed, let's suppose I define import Google dot Gen. AI and then I'm doing model equals to.

Hardip Patel 38:34
Yes.

Tarun Jain 38:48
Google.
I'll just add it as it's a.
Here I'm defining model equals to Gemini.

So how many of you know there is a parameter called temperature?
Oh, hello.

Hardip Patel 39:10
Yes.

Tarun Jain 39:12
So do you know what is temperature?

Hardip Patel 39:15
Not exactly. It's just like liberty to give creative answer, yes.

Tarun Jain 39:17
I.
Correct. So if you notice Google AI Studio, not just Google AI Studio, you can pick any framework. If you only one if I use again face chat, I guess they're not currently existing. OK, so basically whenever you're using any models, right, every model has this parameter called temperature.
What it does is it has values from zero to one. If you give zero, that means your model will not be creative. It is not randomizing. Let's suppose you have your own context. You have your resume.
and you will attach your resume to
The AI model. Now if you want the AI model to only answer specific to your resume, during that time your temperature will be 0. That means you want your model just to answer from the given context and it should not be creative, right? So during that time if your value of temperature is close to 0.
It won't be creative, but if you give temperature to be 0.9 during this time, what will happen is your model will be more creative. So let me give you one case study. Let's suppose you are working in a financial. You have a customer who is working in finance domain.
In finance domain we have multiple policy documents, we have BRD documents and we have few judgments. So you will Add all these documents to AI model. Now this model needs to answer from the given document itself. Even if a single line is not executed properly, it might cause some issues, right?
So during that time, what you can do is your temperature will be 0.1 or it will be 0.

Now let's take second scenario of education. What you're supposed to do is you have chemistry textbook, you have physics textbook, you have mathematical textbook.
Now what you're supposed to do is you need to create a personalized learning. So when you create personalized learning for the students, you need physics to be very creative so that student can understand. And if you're using AI model, your temperature will be 0.7 or 0.8 or 0.9, which is very close to 1.
So if it is very close to one, that means your model will be creative. If it is close to 0, your model will stick to the context that you have given, which is very strict. So is this clear? Now the thing why I cover this now is.
Most of the models that you see will have only limited arguments that they will define whatever is additional rate that will be added as model KWRGS which is model keyword arguments. Inside this you will add all these things. One is temperature. Then you have top P.
So temperature will be 0, then top P will be 0.1. Then you have something called as top K So no need to worry about what top P and top K is. Now I'll cover this when we talk about open source LLMS. So this will be 50.
The major thing what you're supposed to notice now is genii is a function. Now whenever you call this function, you're giving a model. So this model is nothing but a argument or parameter, and then you're defining the value. If you're not sure of what are the other parameters that a function has during that time, you will define.
Find this keyword arguments. So this is very important which we will cover today.
One is arguments and one more is keyword arguments and the last part is scope.
This probably I think most of us already know what is local variable and what is global variables.
Do we have the same thing in PHP as well, the local variable and global variables?

Mitesh Rathod 43:18
Yes.

Ajay Patel 43:19
There is a concept, yeah. Global is concept over there in a PHP.

Tarun Jain 43:20
Yeah.

So basically this to whatever we have read, the usage of parameters and scope is similar to what other programming languages. The only different thing is this arguments and keyword arguments. So let's take one example.

Hardip Patel 43:34
We also, I guess we also have arguments and keyword arguments. Yeah, not like directly, but key based and yeah, we call it named arguments.

Tarun Jain 43:39
Uh, in uh PHP.

Mitesh Rathod 43:43
Except named argument.

Tarun Jain 43:49
OK, so probably while we cover these examples, we can relate to the topic, but this is very important. Whatever we cover in all these three points, right? The usage of parameters will be repeated in the same examples as well, but it's always better to have arguments and keyword arguments.

Ajay Patel 43:51
M.

Hardip Patel 43:51
Yes.

Tarun Jain 44:07
While we are not sure of what arguments that particular function has, so I'll cover that examples now. Our first goal is to check whether the given e-mail is valid or not. So I'll just add validate e-mail and.
You have e-mail as a parameter, so let's suppose I add any e-mail rate. I'll have let's of course tarun@eiplanet.com.
Do you remember yesterday we looked into split function? How did we split?

Ajay Patel 44:43
Yeah.

Hardip Patel 44:45
OK.

Tarun Jain 44:45
So we had different file path. If you remember we had data invoice dot PDF. Then what our goal was to split the particular file and get only the extension. So how do you split?

Ajay Patel 44:48
Hmm.
PDF.
Hmm.

Mitesh Rathod 45:02
Lights.

Tarun Jain 45:03
Yes.

Hardip Patel 45:03
And.

Tarun Jain 45:06
Oh, can you repeat?

Ajay Patel 45:07
No.

Mitesh Rathod 45:07
Uh, slice split. OK, not a slice. Yeah, OK.

Hardip Patel 45:08
dot split.

Ajay Patel 45:09
Darts late, not this.

Tarun Jain 45:10
OK, the auto complete code. So you do split and then you have.

Ajay Patel 45:17
Enter it.

Tarun Jain 45:17
At the rate and now what you can do is if you print check mail, how many elements will you have?

Ajay Patel 45:24
2.

Mitesh Rathod 45:25
2.

Tarun Jain 45:29
Two. Why? Because Tarun is 1 and erplanet.com is 1. Now how will you validate the e-mail? What is the logic that you will this auto? OK, not use this function I'll write here. What is the logic you will utilize?

Ajay Patel 45:33
M.

Mitesh Rathod 45:50
Uh.

Tarun Jain 45:50
Let's take some examples. This is 1.

Hardip Patel 45:51
So uh.

Tarun Jain 45:56
And I'll have a peak at.

Hardip Patel 45:58
It.

Tarun Jain 46:02
dot com.
Then I'll just tell name.com.
Add new.
dot com. So these three are obviously fake and then I'll have one more which is valid.
I'll just add Dhoni at.

Hardip Patel 46:20
Yes.

Tarun Jain 46:27
Wicket.com So how do we write this logic to check the validity?

Hardip Patel 46:32
Um.
Anyone wants to give it to go?
I guess OK, what I'll do is I will split by at the rate and if it is 2, I think like right now we are only using split, we are not using regex or anything like that. So that's why.

Tarun Jain 46:48
Split by at OK.
OK, I reject my friends.

Hardip Patel 46:57
We are using split yeah but by split if it it needs to be two and I will check the zeroth index for nonempty and the one I will take the the first index.

Tarun Jain 47:01
It.

Hardip Patel 47:16
And then I will split it again by dot and it should be minimum. The land should be minimum too at least.

Tarun Jain 47:24
Once again, what was the second thing? You split by at, then check the zeroth index. If it is not empty then.

Mitesh Rathod 47:31
Thanks.

Hardip Patel 47:31
Then I will take. I will split the first index.
By dot.

Tarun Jain 47:40
OK, split the first index by dot again. OK.

Hardip Patel 47:45
I'll check for the length, which should be minimum 2.

Tarun Jain 47:51
Take for the length of OK.
Length of the -1 index, right?

Hardip Patel 48:00
Length of the no, no, no, the result of it.

Mitesh Rathod 48:01
No, it can be anything.

Tarun Jain 48:02
Check of the of -1 index.

Hardip Patel 48:10
Is it -1 index that that also? Yeah, we want to check, but if you want to support the subdomains, I was planning for that anyways, yeah.

Tarun Jain 48:12
-1 index in the sense.

Mitesh Rathod 48:14
It's here last.

Tarun Jain 48:23
OK, check for the length if it is 2, right? Greater than OK.

Hardip Patel 48:27
Yeah, minimum two. Yeah, greater than equals to two.

Mitesh Rathod 48:27
Minimum 2, Minimum 2.

Hardip Patel 48:34
Yeah, then what you said like I will.
Check for the.
No.
Yeah, the last, the last index should not be empty. I guess that's it.
Oh, I mean like both of the OR any of the strings should not be empty. Yeah, yeah, yeah, yeah, yeah.

Tarun Jain 49:01

We should be fine, right? Like if in case this will come before you check the condition. If it is not empty, then you check whether it is more than two or not. Yeah, this also works, at least for this use case.

Hardip Patel 49:12
Exactly, yeah.

Tarun Jain 49:16
So you have validate e-mail. What we can do is you would check the e-mail first.

Hardip Patel 49:17
It.

Tarun Jain 49:25
You will add e-mail dot.
Split.
We can do one more thing. We can check if.
Art is there or not?

Hardip Patel 49:42
Yeah.

Tarun Jain 49:42
At I don't think we covered in statement separately, right? We used in when we saw a for loops, but we didn't check in separately. So if I do add in and if I just add e-mail, let's suppose my mail.

Hardip Patel 49:47
No.

Tarun Jain 50:00
Is tarun@trplanner.com.
And if I do add in mail.
True. So this can be my first condition if at.
Not in mail I can return false. I don't even have to check any other condition, right?

Split only I don't have to do so this will be the first condition.
You understood it what I'm trying to do. So first we'll just check if at is there or not. If at is not there, that means it's not a valid e-mail. I can return false. I don't have to check, split, add a new what you call condition and all now if this is not satisfied.

Hardip Patel 50:30
Yeah.

Tarun Jain 50:45
Here if you see this will return a false right? Once it return false, whatever you write down, it's fine. Now what will be your else?
Or probably I can add it as elif.
This makes sense. Mail dot split at zeroth index. OK, this is what you are trying to do. If the zeroth index is empty then it is returning false. Then again you are trying to check at splitting on the second index using dot. So whatever you wrote in the comment rate, who said that logic?
Hello. Yeah so this logic is being implemented by autocomplete so I'll just use the autocomplete. OK now it won't come. OK, I'll just do validate e-mail.

Mitesh Rathod 51:24
Hardip.

Hardip Patel 51:24
Sorry.

Mitesh Rathod 51:32
Yeah.

RamKrishna Bhatt 51:34
We can also add one more check.

Hardip Patel 51:37
M.
So it it on it only took the negative like false part.

Tarun Jain 51:45
Else returns fault.

Hardip Patel 51:50
Mm.

RamKrishna Bhatt 51:51
We can also check. Uh, we can also add the dot in first. We check whether this dot is there or not.

Tarun Jain 51:52
OK, this won't be right.
Oh, for which one?

RamKrishna Bhatt 52:02
Along with the at the rate, we can also take the code is there or not.

Hardip Patel 52:09
Oh, yeah.

Tarun Jain 52:11
You mean you want to give any condition here or?

Hardip Patel 52:13
No, I'm so he's saying that if we can also check for the dot the period so that yeah, if it is in there then there is no domain.

RamKrishna Bhatt 52:13
Yes, yes.

Tarun Jain 52:22
Yeah.
E-mail dot split.

Hardip Patel 52:29
No, no there there is no need to split no like if it is not there in mail.

Tarun Jain 52:29
So why is?

Mitesh Rathod 52:34
Not in.

Tarun Jain 52:37
No, So what what I'm trying to do is you split by add and then what you're trying to do is whatever you have in your first index. If dot is there, I'll just return true.

Hardip Patel 52:37
Then uh, yeah.
No, yeah, yeah.
Oh, yes, yes, yes.

Tarun Jain 52:49
So you understood this logic if at in e-mail that means it is true. Then I'm also checking in the second statement in the second index. If I have dot that means I'll return true and dot in e-mail dot split.

Hardip Patel 52:53
M.
Yes.
Mm.
Mm.
But then we don't know if uh the first like AI planet could be emptythen.com.

Ajay Patel 53:20
Yes, at the rate I.com will be, you know, invalid e-mail.

Tarun Jain 53:26

One second.

OK.

Oh, can you repeat now?

Ajay Patel 53:33

Actually, let's say user has input tarun@.com, so it is not a valid e-mail.

Hardip Patel 53:34

Open.

Tarun Jain 53:44

Oh.

Ajay Patel 53:44

But according to our yeah, according to our syntax, it it will consider it as a true.

Tarun Jain 53:49

And the 0th index should not be empty.

Hardip Patel 53:52

Yeah.

Ajay Patel 53:54

No.

Tarun Jain 53:57

But that will just complicate what do you do?

Ajay Patel 54:00

Yeah.

Hardip Patel 54:00

Yeah, to go.

Ajay Patel 54:01
Reg X is much better than I would say.

Tarun Jain 54:02
dot -1 and length.

Hardip Patel 54:05
Yeah, this is this is good.

Ajay Patel 54:08
Yeah, auto solution is always good.

Tarun Jain 54:09
Well, this is much better. Yeah, so this is true. If not, we don't even have to check all this condition, we'll just return false.

Hardip Patel 54:16
Yeah, I don't need it.

Tarun Jain 54:20
Cool. This makes sense. So I understood it. We are checking if this e-mail as at. If yes, then I'm checking the second condition where it is checking for the dot in the last index, right? What is there in the last index?

Hardip Patel 54:22
Mm.

Ajay Patel 54:23
Hmm.

Hardip Patel 54:24
Yeah.
Mhm.

Ajay Patel 54:34
M.

Tarun Jain 54:35
And then it's checking for the length if it is more than one. This is for the empty logic, so I'll just run this.
Tarunyaplanet.com it is true.
So regex is also there when we do the modules part, but this is just for parameter logic. I'll just do Tarun dot.

Hardip Patel 54:54
OK.

Ajay Patel 54:57
Yep.

Hardip Patel 54:57
OK.

Ajay Patel 54:59
Yeah, mm-hmm.

Hardip Patel 55:00
Mhm.

Tarun Jain 55:04
This logic.

Ajay Patel 55:04
It is still.

Tarun Jain 55:07
But yeah, here I'll use different again. I'll just use Tarun at AI Planet.
False. But this is only for logic. We don't have to look for accuracy, but you

understood right how usually we have function. So The thing is whenever you return true, that means it's stopping the particular entire flow. So once you see return, that's the end of the function. Now there is also one more thing.

Hardip Patel 55:16
Mm.

Ajay Patel 55:21
Yeah, yeah, yeah. Yes, yes, yes.

Hardip Patel 55:24
Yes.

Tarun Jain 55:35
That we can do instead of returning true or false, we can just return this.
Right. We don't even have to check for if condition. If this is true, it is true. If it is false, it is false.
You got it. So whenever you have any function which is returning true or false.
You don't have to use.
If condition.
If condition.
You can directly use return and the boolean. So this boolean is nothing but but your entire logic.
So does it make sense? Because in most of the time whenever we define function in text preprocessing, right? Considering text preprocessing, we will check if there is any additional functions. Let's take the same example of sentiment analysis.
Right in sentiment analysis.
You have Flipkart and you have a review. So in this review you have hashtag.
You have URLs, you have HTTP URL. So do you think you need URL for predicting sentiment analysis? Let's suppose I have wikipedia.com ragging.
OK, so this particular review is on dragging and you have hashtag which is relevant to rag and then you have something related to a URL. So whenever you are trying to predict whether the given review is positive or negative, do you think URLs will make any sense?

Mitesh Rathod 57:22
Only the hashtag.

Tarun Jain 57:24
Huh. Hashtag will make sense because sometimes people even write text in hashtag, right? You start with this product. Let's suppose Lux was the worst product. Hashtag makes sense.

Mitesh Rathod 57:26
Yeah.

Ajay Patel 57:40
Hmm.

Tarun Jain 57:41
But does URL makes any sense when it comes to sentiment analysis?
No, not right. So we have something called as def remove URLs and then you will add the tweet and here what you can do is you can add return and some logic.

Ajay Patel 57:46
Normally no. Normally no.

RamKrishna Bhatt 57:46
Do not.

Hardip Patel 57:47
No, I just minimal.

Ajay Patel 57:52
Hmm.

Hardip Patel 57:53
M.
Is.

Tarun Jain 58:01

Right, the thing what you're trying to do here is you're returning true or false. So if it is false, you're just removing that particular URL. In most of the cases you will have your function to be true or false itself. So during that time you don't have to use any if condition. So This is why I picked this example.

So let's look at another example. I have def add.

AV.

I'll just return a + B. So when it comes to Python, whenever you're calling the function right, let's suppose I'll call the function. I can use B equals to 10.

And then I can give a equals to five. So this is what you call as positional.

Positional.

Argument or parameter?

Definition. So The thing is it doesn't matter what order you call in, but if you define proper variables you can just call that particular function. In our example if you see you have a, you have B. So here what you are trying to do is you are defining B equals to 10A equals to five. So the order doesn't matter. The only thing is you are defining that as a position.

Now if I print that it will give me 15. So let me take some actual example like def LLM response.

Ajay Patel 59:14

Hmm.

Tarun Jain 59:23

And I'm giving user query and I'm also defining what model I need right? So here if model equals to.

Jemmy 9th.

Let's suppose it will take the query. It took the user query. I'm adding it as a placeholder.

From.

Ready.

And then what we will do is we will just return.

Some response.

And Elif the model double equals to open AI.

Front equals to query.
Return some hallucinated response.
Elif model.
Claude.
Return some coding response. So here basically what we are trying to do is we have a function called LLM response and in this LLM response we support different models. So this model is nothing but if it is what you can do is you can define a prompt then you can have that particular.
Client where you're defining the Gemini model and once you have that client defined, you can return the response which the LLM returns. In my case, I'm just using a placeholder which is some response. Same goes for Open AI, same goes for Claude, right? If it is related to Claude, I'll give mostly the coding related response. And whenever it is related to Open AI, probably I'll give general knowledge based and whenever it is related to translation, probably I'll use M. Now if I use LLM response, it doesn't matter whether I have to define query first or model first. If you want to define model, you can define model first.
Which is Gemini and then query can be what is the capital of India. Anything works. So now you have some response which is inside Gemini. So this is positional parameters. You start with the exact parameter that you have defined in the function and then you give certain value and then same goes.
For the other parameter that you have and you define the value. So does this clear or what is the other type of using function which is positioned?

Hardip Patel 1:01:50
Yes.

Mitesh Rathod 1:01:51
Yeah.

Tarun Jain 1:01:51
Parameter. If you know the position right, you can directly define what is the capital of India and Gemini.
So this is the easiest method, but not many of the times we'll use just one or two arguments. We might have multiple arguments, so it's better to define based on the positional parameter, which is also very better readable code rather than just

defining.
The direct values. So now if I run this then I'm getting some response, but here I need to know the exact order of it.
Is this clear?
Hello. Yeah, just a second. I'm getting a call.

Hardip Patel 1:02:31
Yeah.

TJ **Tarun Jain** 1:02:36
We.
OK, sorry, they have shared the cab now itself. I told them to send it at 12:30. You just give me one moment. That's I'll come.
Uh, OK. Hello.

Hardip Patel 1:04:06
Hey.

TJ **Tarun Jain** 1:04:06
Yeah, so is this clear? One is directly we can define without any parameters. So most of the times you'll have function which will not have any parameter. Then you have return function and after return whatever you have will be ignored.
I'm just recapping what we did after return.
Nothing will be considered.
And then we had a function where we have an argument. So whenever we have arguments, there are two ways we can call this. The first way is by defining the order that we have. But if we are not sure of the order, during that time you can use something called this positional parameters.
Where you can define the exact same parameter that you have defined and then assign the value as as we have done here. So for example I have LLM response and I have two parameter here.
Sorry about the disturbance. Then we have LLM response. So basically in LLM response, if you are not sure of what are the parameters that you have and in what order it is during that time, what we can do is we can just define it as query equals to value and then model equals to that particular model name.

So this is the 2nd way to define the parameters. Now what we will do is we will look into arguments and keyword arguments.
So basically when it comes to arguments, right, I'll just add comment.
Let's suppose you have a list.
List of marks.
Or it can be list of product.
8.
If you want to pass any list component as a parameter to the function, during that time you will define it as arguments. And second thing is if you have dictionary for keyword arguments if you have any card.
Which is dictionary where you have key, you have value, then you have any product which is key value.
During that time you have to define it as a keyword arguments, so basically the syntax is.
You have def, then you have any function name.
And then you have single argument.
Single argument is basically if you are using list and the syntax for.
This is syntax for arguments.
Now this is the syntax for keyword arguments. You have def function name, then you have double asterisk, then you have keyword arguments.
This is mainly used.
When you have.
Dictionary.
This is mainly used when.
You have list.
I start pass which is a logic pass logic. I'll repeat again. So we have a function and if you are if you have the parameter which is list and if you don't know which particular value to provide then what you can do is you can pass the entire list.
As argument which is starts with what you call single hash trick. So this will consider.
This will consider all the elements in the.
List as the parameter.
And if you have dictionary as your input variable, what you can do is you can define as double asterisk. So this will only ensure that if you want to use any position and if you have it defined as a single asterisk, it is very simple to understand and same goes for dictionary as well.

So once you define function name and if you give double asterisk keyword arguments, it will consider the entire dictionary. Now where is this used? I already gave one example. Let's suppose you have any model client. For example, I have Open AI client.
In this open AI client we have model name, then we have API key.
And here we also have temperature. You have top P.
You have top key. What else do we have? We have model. Then you have API key, temperature, top P, top K Then there is also repetition penalty.
Then you have streaming.
So in most of the cases, if you notice these three parameters, temperature, top P, top K, if you don't know the order of it, what you can do is let's take this separately and you can define this as a keyword argument.
So you can remove this and then you can define it as double asterisk model keyword arguments. So whenever you encounter this model keyword arguments within your logic, right within your program logic you can define.
Temperature equals to model keyword arguments, which is temperature.
And then we have toppy toppy toppy.
And what I'll do is I'll also define model equals to model.
And now you can just print all the statements.
So are you guys understanding? What is currently happening is most of the times whenever you're defining any function, you might not be sure of the arguments list that you're providing. You can just add it in a model keyword arguments just to make sure that that particular variable logic needs to be used in the key one code or not.
So if it is a dictionary during that time we will use keyword argument which is double asterisk. If it is a list you will use as a single asterisk which is a argument. So in our case what we are trying to do is we want to define an open AI client which has the possibility of temperature top P and top K which a user will give.
So if user gives those argument, you only have to use those arguments and then you have model. So model again this is simple parameter. Whatever you see here right in model to streaming is a simple parameter, only model keyword arguments.
Whenever I call it should be a dictionary. So I'll just run this.
Now what I will do is I will define my client equals to open AI client and for model what I'll do is model equals to Gemini.
OK, it should be GPT and then API key.
It will be something starting with SK.

And then repetition penalty. Repetition penalty will cover once we come to the open source LLM which is 1.1.
Streaming is false. Now what I will do is I will define my model keyword arguments. Which is a dictionary.
Temperature to be 0.1.
Copy.
To be 0.7.
And top K to be 50.
Do you guys understood what we are trying to do?

Hardip Patel 1:12:41
Yes.

TJ **Tarun Jain** 1:12:43
So if anyone wants to know what streaming is streaming, basically if you need one word 1 token at a time. If you don't need one token at a time, you can keep it as false. So false in the sense it will wait till you have full stop at the end of the code, then only it will print the response.
And now I'll just print this.
Is it throwing another?
OK.

Hardip Patel 1:13:18
Yeah.
Oh, I you think we don't even need to use this message, right?

TJ **Tarun Jain** 1:13:37
Huh. If you want to give temperature, let's suppose if I give temperature.
This is the easy way to do, but it's better to have dictionary.

Hardip Patel 1:13:53
OK.

TJ **Tarun Jain** 1:13:55
So let's suppose if you don't know what the keyword arguments is passed in the list.

What you can also do is you can just define temperature topi topi which is inside this particular value. So now if I run this.

Hardip Patel 1:14:09
No, that shows that there is an angle, OK.

Tarun Jain 1:14:12
No, you forgot comma.

Hardip Patel 1:14:14
Um.

Tarun Jain 1:14:18
So if you want to take one simple example, right, what we will do is let's take one simple example where you need to calculate the total marks and average of the student. OK, so now what I'll do is I will just use argument. OK, so if I use argument, how should I call this?
What will be the data type inside this?

Hardip Patel 1:14:42
Integer number of the I mean the list.

Tarun Jain 1:14:47
No. So again I'll do it. If it is argument, how should we call it?

Hardip Patel 1:14:54
It will be the least.

Tarun Jain 1:14:55
It should be a list. So here what I'll do is I'll add some marks. I'll add 80, I will add 90. I'll add 897645. OK, so now how do we calculate the total marks of this?

Hardip Patel 1:15:00
OK.

Tarun Jain 1:15:15

Total marks.

So what will be the logic?

You can just run a for loop and then you can add every single value. But we also have a shortcut command. So the shortcut command is you can just use sum.

And now if you print this.

You have the sum of it, so this is your total marks. So what I'll do is I'll just add sum and I will add arguments.

Now for the average, this is correct. Total marks by length of average. Is this clear?

The logic.

Hardip Patel 1:15:55

Yes.

RamKrishna Bhatt 1:15:56

This.

Tarun Jain 1:15:56

So now if I print this.

OK, why is it running?

OK.

Just one second, let me check the logic.

OK, this should be correct.

Open marks.

OK.

OK, one second if I remove this.

OK, now it works.

All right, so I'll repeat again what we are trying to do. So if you notice in the calculate total marks average we are defining an argument. So whenever we are taking this argument, I'm not sure what the user will call. So user will keep on adding multiple values that it has.

Ajay Patel 1:16:52

OK.
Hmm.

Tarun Jain 1:17:06
Let's suppose 768900. Now no matter how much values you pass, it will consider everything as a list. So now if I print the type of PRGS, let me just check if it is list or it is.
Tuple. OK, it is tuple.

Ajay Patel 1:17:25
Plus.

Tarun Jain 1:17:27
This is taking as tuple and once it is tuple what it is trying to do is it is trying to call the sum of it and then you're trying to calculate the average and once you have the average you're returning both the values but the only logic that you're supposed to. Look into this particular example is how you're calling it. So if you have multiple values if.
If you have multiple values to pack.
Inside the given function.
We are defining it as star when the input is list or tuple. So if you consider this particular example, it is a tuple.
And once you have the tuple, you are not sure how many values user is adding. So that's the reason why you are adding star, right? So what star will do is it will consider all the input values that you are passing. The same thing for keyword arguments. What we are trying to do is let me add text.
So for keyword arguments.
Instead of.
Defining the values directly. So what is happening in argument is you're just giving the values. You're not giving any keyword, whereas in keyword arguments you will define it as a variable. You will define temperature to be 0.1.
Then you will define uh.
Top K to be.
0.1 then top P again I'll define zero. So here if I check.
Deep type.

Of model keyword arguments. So what should this be?
What should be the class of this?

Mitesh Rathod 1:19:32
Amit.

Hardip Patel 1:19:33
I guess uh uh, dictionary dictary.

Tarun Jain 1:19:36
It will be dictionary.

Mitesh Rathod 1:19:36
OK.

Tarun Jain 1:19:38
So if I run this it will be dictionary.
So is this clear or do we have any doubts? We can take some more examples on keyword arguments and argument because this concept is very important.
Hello.

Hardip Patel 1:19:56
Yeah.

Tarun Jain 1:19:57
So I'll repeat again. The logic is define a function name and if you are defining single asterisk and then arguments, the user can call any number of values which is a list or it is a tuple.

Hardip Patel 1:19:58
I think I understood.
OK.

Tarun Jain 1:20:12
And once you define the tuple or list inside the functioning, you can utilize that

particular particular values as a list.
Inside.

Hardip Patel 1:20:23
Hello.

Tarun Jain 1:20:24
This function the args will act as tuple or list, but when it comes to keyword arguments.
Inside this function, the keyword arguments will act as a dictionary. So the only difference is in tuple you can just keep on adding values, but in keyword arguments you're defining the exact value which is temperature and then you're defining the value which is 0.

Hardip Patel 1:20:45
It.

Tarun Jain 1:20:56
So this entire thing will be added as.
Tim.
For HR of 0. So this is the reason why it is treated as dictionary inside keyword arguments.
Is this clear?

Hardip Patel 1:21:11
Yes.

Tarun Jain 1:21:13
So we'll take two more examples on this so that this concept will be clear. But The thing is, we'll have to stop here and then continue in the next class.

Hardip Patel 1:21:15
OK.

Tarun Jain 1:21:22
So that I have the cab waiting, that's why.

Hardip Patel 1:21:28
It.

Mitesh Rathod 1:21:28
Yeah, it's OK.

Tarun Jain 1:21:28
Hello.

Hardip Patel 1:21:29
Yeah, that's fine.

Tarun Jain 1:21:29
What I'll do is I'll just add a comment.
To pick up three to four more examples.
On keyword arguments and arguments. So just one take away what I'll do is.
Take away your assignment.
Look for at least.

Hardip Patel 1:21:55
OK.

Tarun Jain 1:21:56
10 parameters.
That Open AI has.
I'll tell you why open AI, because let's suppose how many of you know Brock.
Have you seen this before Grok?

Hardip Patel 1:22:16
Yeah, yes.

Tarun Jain 1:22:19
So Grok is a faster LLM inference, which if you give any open source model, right, it's generating response probably in 0.1 milliseconds, right? So Grok is fast and then you also have something called as Samba Nova. I'm not promoting the product, but what I'm trying to tell is.
Every single API calls, if you see right the API calls, is there docs?

Hardip Patel 1:22:44
M.

Tarun Jain 1:22:46
You can all quite done.

Hardip Patel 1:22:48
Uh, under developers I guess.

Tarun Jain 1:22:58
It.

Hardip Patel 1:23:03
Flower dogs.
Sunday the 5th, 5th.

Tarun Jain 1:23:10
It's not like.
I.
OK, so can you check this example? Now you have client dot chat dot completion dot create, then you're defining system prompt and then you're defining the model name, right? So model is what is this now compared to what we saw earlier in function?

Hardip Patel 1:23:20
Yeah.

Tarun Jain 1:23:33
What is this referred as the model?
No, no. So basically.

Mitesh Rathod 1:23:38
OK.

Hardip Patel 1:23:39
Oh uh uh, the open AI.

Tarun Jain 1:23:41
Here if you see this model API key penalties parameters, right? So message is a parameter, model is a parameter, right? So if you look at the syntax, yeah.

Margi Varmora 1:23:44
Argument.

Ajay Patel 1:23:44
Named arguments.

Hardip Patel 1:23:47
Yeah.

Ajay Patel 1:23:48
OK, parameters.

Hardip Patel 1:23:51
Yeah, yeah.

Ajay Patel 1:23:52
Named arguments so we can. So having a named arguments, we can interchange this place also that it doesn't.

Tarun Jain 1:24:01

Correct. So you can define model 1st and then you can define message. But the major thing what we have to notice is look at this syntax client at completion create right now if I come to Samba Nova.

Ajay Patel 1:24:05
Mhm.
Hmm.
Hmm.

Tarun Jain 1:24:23
Tutorials.
M.

Hardip Patel 1:24:32
Mhm.

Tarun Jain 1:24:32
OK, where did I have written the code?

Hardip Patel 1:24:36
This.

Tarun Jain 1:24:37
Wait, let me sign in.

Hardip Patel 1:24:37
It it might be under reference. Uh, you might have to check API reference.

Tarun Jain 1:24:45
So usually they when you create API kit shows the reference. So now if I click on Python if you see client dot chat dot completion dot create and then you have message you have model, how are they calling it? They're calling temperature separately, topic separately but when you if you.

Hardip Patel 1:24:48

Mm.
Yeah.
No.

Tarun Jain 1:25:04
Look at the create function right? So create is a function inside that these two things will act as a dictionary which is keyword argument.

Hardip Patel 1:25:13
Right.

Tarun Jain 1:25:14
Right, So what I'm trying to convey here is look at the syntax, look at Grok. Then you can also check for a inference client.
Even these folks have the same logic. You have client chat completion and then create. So as a take home assignment what you can do is look at.
Where did that go?

Hardip Patel 1:25:52
Yep.

Tarun Jain 1:25:54
Take away assignment. Look at at least 10 parameters that Open AI supports.
The reason of the reason of checking open AI instead of Gemini or Claude is because.
Most of the.
Of the.
Other LLM.
Are Open AI compatible, right? For example, you have Glock, you have Sambanova, you have Together AI.

Hardip Patel 1:26:28
Thanks.
Hugging face.

Tarun Jain 1:26:36
You have a gin face.
And uh, there is also something called a fireworks dot TI.

Hardip Patel 1:26:41
I guess replicant.
But does the replicate also use the open? No.

Tarun Jain 1:26:46
Replicate is also open AI compatible. So these two are very awesome. Grok and Sambanova, it's super fast. Then there is also something called a Cerebrus. All these are open AI compatible. If you want to see where all this code is available then probably you can check Agno.

Hardip Patel 1:26:57
Mhm.

Tarun Jain 1:27:06
I'll send you the URLs so that you can at least make a list of 10 arguments. So we will use the same arguments and when I cover this examples right, I will use same example so that when we learn how to use open source LLMS, we might have already seen some of the variables that.

Hardip Patel 1:27:16
Yes.

Tarun Jain 1:27:24
This library supports.
So here we have models. So if you see you have Anthropic, you have AWS, you have Azure, you have Cerebrus, which I just said, you have Fireworks and you have Open AI, you have Samba Nova, you have Together. So now if I click on Samba Nova.

Ajay Patel 1:27:27
OK.

Hardip Patel 1:27:41
Mhm.

Tarun Jain 1:27:47
And if I click on Sambanova UI if you see a from agno dot models it's open AI like like you you're not even finding separate client but if I open Google.

Hardip Patel 1:27:50
Mm.

Ajay Patel 1:27:52
Mm-hmm.

Tarun Jain 1:27:59
Gemini, you don't see that line, so Google has its own logic, but now if you look at Glock and Sambanova.

Ajay Patel 1:28:05
Mhm.

Tarun Jain 1:28:09
I showed Sambanova. I'll show Brock.
OK, why did they remove? So Grok is also Open AI compatible as well. But what you can do is just look at this folder. This folder has minimum six to seven LLMs which are Open AI compatible. The only thing what you have to do is just note down the parameters. That's it.

Hardip Patel 1:28:20
OK.
Mm.

Tarun Jain 1:28:33
Now what are the parameters? Excluding what I mentioned, don't repeat this like model API, key repetition, penalty, streaming temperature. Apart from these things, if

you see any new variables, just note it down and add 1 statement what it is about. That's it.

Ajay Patel 1:28:33
Mhm.

Hardip Patel 1:28:52
OK.

Tarun Jain 1:28:52
Oh, is this fine?

Hardip Patel 1:28:54
Yeah.

Ajay Patel 1:28:54
Yeah.

Tarun Jain 1:28:56
So this is the URL.
This is very important. Like when we look into the open source LLMS, this will make more sense. Like what is repetition penalty? What is temperature? I today covered only temperature, but we also have top P We'll have top K Now there is something called as sampling in LLM.

Ajay Patel 1:29:10
OK.

Tarun Jain 1:29:19
To avoid hallucination. I'm not sure if I'll get that in the first URL, but.
Huh. So you will use this sampling algorithms, right? For sampling LLM is basically there are certain strategy, certain strategy in the sense what should be the temperature value, what should be the top K value and what should be the top P.

Hardip Patel 1:29:29
OK.

Ajay Patel 1:29:30
Reg.

Tarun Jain 1:29:45
So that LLM doesn't deviate from the data it is given. So when I said top P that means I'm giving certain finance data to my LLM model and it should only look into the data and answer which is RAG basically right? So all the parameters that I mentioned here.
Which is repetition, penalty, temperature, top P, top K. This is called as sampling parameters.

Hardip Patel 1:30:08
Yeah.

Tarun Jain 1:30:14
Which is very important when we cover RAG. So now only what we can do is let's list down. We don't even have to know what that is about. List down the parameters and one statement of what it is about parameters and one statement.
Why is it needed? And in tomorrow's examples I will take all the parameters that you have defined and then we will create the examples.
So is this clear? This URL is very important.

Hardip Patel 1:30:45
Yeah.

Mitesh Rathod 1:30:49
Yeah.

Tarun Jain 1:30:50
The diagonal one where you will see the Excel code and the parameters.

Ajay Patel 1:30:54

OK.

TJ **Tarun Jain** 1:30:57

Fireworks together.

I guess hugging phase I only added the code but remaining we can check it out.

Hardip Patel 1:31:13

Mm.

TJ **Tarun Jain** 1:31:14

Cool.

Yeah, anything else?

Hardip Patel 1:31:20

No, I just want to correct myself before I say that args and K keyword arguments are there in Pi PHP, but it isn't. There is a splat splat operator which does the same spread.

Mitesh Rathod 1:31:21

Yep.

OK.

Hardip Patel 1:31:36

So there is, but we don't have changes in Splat SPLT.

TJ **Tarun Jain** 1:31:37

Hello.

Can you spell that what operator?

SP.

Hardip Patel 1:31:47

SAT.

Tarun Jain 1:31:49
LAT OK, I'll look into it and I will.

Hardip Patel 1:31:50
Yeah, yeah, it's a pipe symbol. Sorry, it's a three dots split operator, but it is not similar to args and KW keyword arguments because it's a two different.
Type that one is the named and the first one is list which is not available in CSE. Just want to correct myself.

Tarun Jain 1:32:12
Got it. No, I'll check that code. So tomorrow probably we'll be completing Python basics and once we complete Python basics, we'll directly get into oops and whatever you need to understand open source LLM. So once you go through these files, right, Agno, you will come across data classes.
You will come across oops and you will come across meta classes. So 15th is what we will cover these three concepts. Oops data class. Data class is very important.
Whenever you are working with open source frameworks, data class is common.

Mitesh Rathod 1:32:35
OK.
M.
Mhm.

Tarun Jain 1:32:49
Data class and Pyrantic. Now why Pyrantic? You might have noticed some of the output format, right? Open AI has to generate only in Jason. It should only return in Markdown. How do you maintain the consistency? This is where Pyrantic comes in.

Mitesh Rathod 1:32:49
Really.
Yeah.
Peter.

Tarun Jain 1:33:05

So these are the four major things that we'll cover. Oops, data class, meta classes, meta classes. It's not important, but if you want to understand the code bases right to write your custom logic during that time, you can just see you will relate OK what this particular line is doing. So meta classes is just like.

Mitesh Rathod 1:33:10
OK.
Yes.
Yeah.

Tarun Jain 1:33:25
What you didn't list. So if you understand data classes, meta classes is just observation, it's not even writing code. And then pydantic. Pydantic, it's to maintain certain readability in the code, like how you should write code and how you should maintain the format. And this is mainly important for getting structure.

Mitesh Rathod 1:33:31
Yeah.

Tarun Jain 1:33:45
Responses like consistently Open AI has to be in Jason itself, right? For developers, if you're automating some things you need to have in Jason format. So Pyrantic is important. We will cover these four things in 15. So hopefully we should complete Python tomorrow.

Mitesh Rathod 1:33:47
OK.
OK.
OK.
Yeah.
OK, well, yeah, that's sweet.

Tarun Jain 1:34:04
Yeah, but just make sure you have at least one document ready tomorrow for the API keys, just parameters and one liners on it.

Hardip Patel 1:34:12
OK.
OK.

Tarun Jain 1:34:15
OK.

Mitesh Rathod 1:34:16
It's if you can get me. Thank you.

Tarun Jain 1:34:17
Yeah. Uh, thanks.

Hardip Patel 1:34:18
Thank you. See you tomorrow.

Ajay Patel 1:34:20
Thank you. Thank you, Tarun.
Yeah.

Ajay Patel stopped transcription