Python and AI Power-Up Program Online Classroom-20250811_113446-Meeting Recording

August 11, 2025, 6:04AM

1h 46m 15s

Ajay Patel started transcription

Tarun Jain 0:04
To reopen my laptop, I mean the browser.

Ajay Patel 0:07
No, no worries.

0:08
Sure.

Tarun Jain 1:17
OK, I'll try to share my screen again. It was for the permission.

Ajay Patel 1:20
OK.

Tarun Jain 1:26
Oh, now it's visible, right?

Ajay Patel 1:28
Yeah, it is visible.

Tarun Jain 1:30
So yeah, hello everyone. Welcome to this sessions. So mainly what our end goal for this particular sessions will be that we'll get started with basics. We'll start off with Python programming and then the major thing is these three main topics. We'll touch base on some of the.
The most important concept in large language models, which is a fundamental building blocks for learning Grag agents and we'll also have fine tuning your own

large language model in the end. So by own large language I mean we'll use the base model, we'll feed our own data and then we'll fine tune it. So this is the future.

Ajay Patel 1:53
Mhm.

Tarun Jain 2:09
Purpose of what this particular workshop is about. So here is a quick intro about myself, right? So I work as a super agent. It's just a fancy keyword for data scientist. So I work at AI Planet. It's a Belgium based AI startup. I work remotely and apart from that I've also been part of Google Summer of Core for.

Ajay Patel 2:12
Hmm.

Tarun Jain 2:29
In 2024 I was part of Run and Lab and in 2023 I was part of CM Microscope and I also am part of Google Developer Experts program in AI and in my freedom I watch anime and I read manga, so I know most of you are also anime fans, so we'll have some discussion around that.
Maybe indirectly via examples of code that we write. All right, so before you can proceed right, I just have a few quick questions. I'll probably open both the screen at the same time.

Ajay Patel 2:54
Mhm.
Mhm.

Tarun Jain 3:05
So you're able to see my screen.

Hardip Patel 3:08
Yes.

Ajay Patel 3:08
Yes, yes.

Tarun Jain 3:09
OK, so I just have a few quick questions. So how many of you are already aware of Python programming? I mean you have written some code in Python. You can probably just raise your hands.
OK, two of them. So have you built any LLM based application or LLM based app which is related to RAG or agents?

Ayush Makwana 3:35
No, not yet.

Tarun Jain 3:35
All right. OK. OK. So this is just to keep the pace on what we need to usually do for this particular sessions. So we'll start off with the basic itself. I'll probably also share some bit about what our total agenda is and what we will be doing in this particular week.
So this way it will make more sense what kind of examples we need to cover on in what quantity we need to cover. That will be more clear in the upcoming weeks as well. OK.
So here are a few important points. Usually, let's keep it interactive. We'll have lots of examples. We'll also play around. I write some piece of code. You write some piece of code. So usually I conduct workshops in a roundtable conference. So I'll tell what this roundtable conference is mainly about. Let's suppose we are 10 people.
In this room, I write some piece of code. There might be times that code might have alternatives, right? Because obviously we are programmers. We know you can't approach one solution with just one approach, so probably when I share something.
In the core demonstration, I might have my approach. You might have your own approach, so feel free to share your approach. OK, this is more optimized. This is more alternative. This way, when it comes to workshop styling, people will get more used to it. OK, I know one approach, you know one approach, right? And not just with the approach.
Approaches or the concept. If anyone have any questions, I might have my answers.

You also have some points which you can add on, right? So if you have those points that you want to add on, you can also add those details so that way everyone are engaged, right? So let's have interactive sessions. That way no one will sleep, right? That's the.
Whole purpose of my 90 minutes. I don't want anyone to just be distracted. So in between I'll add some examples which will be very encouraging as well. We'll take small breaks as well. During those breaks we can have few questions, keep it interactive and all.

Ajay Patel 5:28
Mhm.
Mhm.

Tarun Jain 5:39
And one more thing about the workshops that I conduct, all the code that we write will be completely live in the sense you write code along with me. But there might also be chances where I'll give a template code. So template code is basically you don't have the entire code.
You just have some piece of code which is already written, right? So this is just to save our time and this template code, whatever I'm telling right, it is the most basic code that we will write, right? We will not have any major concept code which is already written so that way.
The major concept is written live that we will understand what is the intuition behind this and whichever is the lightweight code rate that will be predefined and this will only happen for a few of the sessions. And also when I say about the template, all the source code will also be available on GitHub after every single session.
Probably I'll maintain a group or I'll just send an e-mail of all the participants who join, so that way you have the source code available even after this session. And I'll also maintain a single slides. So basically for first week we don't need any slides because it's just Python programming.
But once we start with an LP rack, there are few architecture diagrams I have for which we need slides and I'll also share this at the end. All right, so mainly this is it. We'll be having some of the references code as well, so that this reference code is basically for you to have some assignment.
So that way we can have some engagement or you know what you're doing in the

session and you can also replicate that somewhere. So this is the main purpose of the entire session, whatever we are conducting. And yeah, enough with the talk. Probably let's start with the core demonstration.
And for the code, what we will do today is we'll start with Python recap. So if anyone is starting Python for the very first time, we have some of the source code available and I'll also relate the concept with JavaScript, right? So I hope everyone of you are JavaScript developers, right?

Ajay Patel 7:45
Uh, not everyone, I guess. Uh, Ishan, Runak, uh, then who else? I guess Margi and Mitesh. They have four of them are majorly work on to a PHP side.

Tarun Jain 7:58
Oh, OK, understood. So I'll try to relate the concept and but most of the concept whatever I cover in quick start in Python recap, I was aiming that for two days which is around 180 hours, 3 hours. And if in case we need more time for that, we can probably have more examples and we can.

Ajay Patel 8:01
Yeah, but.
Hmm.
Mhm.
OK.

Tarun Jain 8:18
Increase the pace for that. But the major focus is to understand the Python concept, mainly for contributing to open source, because most of the time nowadays we have Python frameworks, right? You have Langchain, you have Lama Index, you have Agno.
It's fine that we use that framework, we write the code, but we'll also have times when you need to build your own custom component, right? So if you want to build your own custom components, what is the logic required? How to build? What are the concept required? That is what we will cover in intermediate Python concept.

Ajay Patel 8:53
OK.

Tarun Jain 8:53
I will give you one example. Let's suppose I write a Python code. This Python code has an LLM call. I'll do LLM call to ChatGPT. Now what I need to do is I need to track the LLM calls with just one line of code, so that is where you have decorators.
Now this might go over your head because you might have not heard decorators for first time. So this is mainly used to trace whatever you're writing within our functions, right? Functions. Everyone remember you just write a logic block.

Ajay Patel 9:24
Yeah.

Tarun Jain 9:27
And then you just call it right. So if you want to track that particular function, you have different frameworks you can use. So this is where when we talk about intermediate Python concept, I'll only share details on what is relevant for you to get started with open source.
And to understand the open source repositories, right? So that is the main purpose. So even if you go build drag, you will understand, OK, I can go contribute to this concept. If anything doesn't make sense, I will have my own custom component.
So the second agenda is to make you perfect or at least get you started with how to write your own custom elements. And once that is done, we'll get started with NLP where you can learn about pandas. Then we have NLTK. NLTK is the most important framework that we will be using in the.
The NLP and data science and once we are aware of that, probably I'll also uh cover ugging face. So how many of you know ugging face here?
You can just raise your hands. OK, one.
Cool. So just to give some context of what tugging faces, everyone of you know that Ghibli Studio Art, everyone were uploading their images, convert that into Ghibli Art Studio, right? So that is a stable diffusion model.

RamKrishna Bhatt 10:45

Yes.

Tarun Jain 10:48

What stable diffusion does is you upload an image or you just give a text. Hey can you generate a horse at space right? And then it generates the image. So stable diffusion most of the applications that you see.
Are closed source. If you see open source right, those open source models are available on Hugging phase. So Hugging phase you can consider it as a hub of open source models, whether it is on text related, image related, video related, you'll find all this open source weights available on Hugging phase.

Ajay Patel 11:23

Sorry to interrupt you. So can we say that hugging face is something like a GitHub for models?

Tarun Jain 11:23

For example, you have.
Hi it's a GitHub for AI models.

Ajay Patel 11:33

OK, OK.

Tarun Jain 11:36

So usually you have baits, right? You build a model and if you want to reuse that particular model, you have baits associated with it. So this weights are publicly available on arging phase. It's like you have repositories available on GitHub, just like that you have baits.

Ajay Patel 11:37

Mm.
Mhm.

Tarun Jain 11:53

AI models available on Eigenphase. So that is what we will be covering here in the essential Python modules for NLP and Data science. And then we'll get into the core concept of this entire session, which is much no LLM concept focused on LLM development centric like text preprocessing, prompt engineering.
Plus context engineering and why are these two different? We'll cover that in the upcoming classes and then how to use closed source LLA models and open source LLA models. And this two topics are developer friendly, which is how to get started with vector databases.
And if you want to boost your performance of building RAG, you need to understand this two concept which is indexing and filtering. It's not covered in most of the tutorials. There is a very special book which covers some of the concept. I'll also share that PDF once we are at this topic that PDFI just got it.
From one of the pirated websites. And then we have RAG, which is very important concept that everyone must know. And whenever you build any LLM based application, it's good to have evaluation metrics, right? How do you evaluate LLM? How do you evaluate RAG? How do you evaluate agents?
And then we'll start with agents and then fine tune them. So this is the overview of the entire session that we will be doing today. We'll start with quick start. Tomorrow we'll have Python recap. So this is the QR code. It's a template code again.
So if you can scan this or is it fine if I share the URL? That way it is very.

Ajay Patel 13:30
Yeah, yeah, sharing a URL would be much nice because you know.

Tarun Jain 13:34
Yeah, yeah, I'm not in person, so I'll be coming in person next week. So the NLP concept, we can do it in person because that is very important.

Ajay Patel 13:39
Yeah.
Mhm.

Tarun Jain 13:53
Oh, I've shared the link in the chat.
So before you can open the Collab notebook, I also want to share some details why

we are using Collab. So let's suppose everyone of us knows what is extension, right? Let's suppose if you are using JavaScript, you have dot JS and then you write the piece of code in that.
And if you're using PHP, obviously you'll have dot PHP and then you write a code just like that. In Python you write all the instructions that you want inside a dot PY file, and once you write dot PY file, this is called a source file.
In this source file you can have all your instructions that you're writing and then what Python does is it will automatically convert that into a byte code. So byte code is basically a dot PYC file. Now to someone who has already used Python, you might have noticed.
As soon as you run up Python code, you see this file Pi cache.
Right, so obviously now you might not notice, but once you start writing code in VS code, right, what I want you to notice is write a Python file. Once you write a Python file, just run it and as soon as you run it, you will have.
By cache folder right? So what this by cache folder does is it will say dot PYC file. So this PYC is nothing but the low level bytecode which will convert our source code into runtime.
So the runtime engine that we use here is PVM. So here PVM stands for Python Virtual Machine.
So what this will do is whatever code you have, it will convert this code into instructions.
And then execute it.
At runtime, and This is why you usually prefer Python to be as a interpreter.
Now what is this interpreter? Probably if you have worked with JavaScript, you have something called as just in time, right? So.

Ajay Patel 16:07
Hmm, even PHP also has the same traits.

TJ **Tarun Jain** 16:10
Right, so basically you don't run the entire code at once. Obviously you're compiling the code after you compile the PBM that we have the Python virtual machine. What this will do is it will run one instruction at one single time. So if you have a code of 10 lines.

Ajay Patel 16:12
Hmm.

Tarun Jain 16:30
And then if you get error 3 errors, it will only tell you one error at a time, right? So you're just executing one line at a time which is interpreter. So the URL that I've shared you, this is a URL of Collab. So what Collab will do is you just write one line at a time and then you just execute it.
You understood why we are using Colab. It's a easier way to write Python program because you can see what is happening at every single line, right? So if you have 10 lines of code, probably sometimes you might want to know OK, what is happening in this line, what is happening in this line. So we will start with Colab.

Ajay Patel 16:54
Hmm.

Tarun Jain 17:10
And once we start with Collab, then we will proceed with IDE, which is interactive or development environment, right? So for the second or third session we'll be using IDE, but for starting two days we'll just use Collab.
OK, so how many of you are using Pollab for push time?

Ajay Patel 17:29
I guess in one of your session we have used.

Tarun Jain 17:32
OK, I have, but I don't think everyone were there in that session.

Ajay Patel 17:35
Yeah.

Tarun Jain 17:37
OK, so I'll also share some of the tips and how to run, what are the different commands that you can use for Collab. So basically Collab, it's very simple. We just

have to write Python code and we have different features.
Now, one of the features that is very good in Colab is the GPU environment. Let's suppose we are building any or if you're inferencing any open source model, right? So you have something called as change runtime type. I'll repeat again. Are everyone are are on the same screen?

Ajay Patel 18:13
Yes.

Tarun Jain 18:13
OK, so can you see runtime?

Hardip Patel 18:14
Yeah.

Ronak Makwana 18:16
Yes.

Tarun Jain 18:19
Once you click on runtime, there is something called as change runtime type.

Hardip Patel 18:20
Yeah.

Ajay Patel 18:25
Hmm.

Tarun Jain 18:26
And you might notice you will have CPU and you will have T4GPU and then you will have GPUs.

Ajay Patel 18:31
Mhm.

Tarun Jain 18:33

A 104 won't be there or you can select on T4GPU. You can click OK and then you can click save. So now what has happened is you can inference any LLM model which requires GPU until 15 GB of VRAM, right? So you can click.

Ajay Patel 18:36
OK.
OK.

Tarun Jain 18:53
Click on this. Can you see this Ram and disk?
So this will tell you how much of the RAM you're utilizing within your code and what is the disk space you're utilizing within the code. So this resources is very important. Probably for first few session we might not notice anything, but as soon as we start with tugging phase.
You will get to know OK why this resource is important and how your RAM boost and how your disk space is utilized. You will easily be able to manage when you're using Collab. So you feature and there is other shortcut as well.
If you just click on any block and if you just click on B you have code shell. So this is what you call as interactive shell.
So interactive shell in the sense you can write any Python codes which needs to be interpreted right? And then you also have something called as text. So text basically if you want to I write a piece of code under what's updating right? So that time you can use text and it supports markdown language.
I hope everyone knows markdown.

Hardip Patel 20:06
Yeah.

Tarun Jain 20:06
Then it will preview on the right hand side. All right, so these are the two things for timing you need to know. And now what we can do is there are an instruction written at the start of the code. So what you need to do is.
Create the copy of this particular code and then you can write anything that you need. So why are we supposed to do this? Whatever I shared you just now, you have share access, right? I click on share.

Have given you view access. So let's suppose you write any piece of code here. You'll write word 2.
Equals to session and now once you close this particular tab and if you reopen this again, whatever code you write it will be gone right? So just to avoid this issue what you can do is you can click on file.
Then click on Save a copy in Drive. This will create a complete new notebook for you where you can edit, write, do anything that you want, right? So this is very important. Whenever you get any piece of code but which is running on Collab, it's always better to create the copy for it.
And once you create the copy, you can edit anything that you need.
So I'll repeat again. What we will do is we'll create a copy so that we can edit the code that we write, and in order to copy the notebook, I've already given the three lines of instructions. One is we click on File, then save a copy in Drive.

Ajay Patel 21:28
Mm.

Tarun Jain 21:42
Once we click save a copy in drive, it will ask you a window. Do you want to open this in a new tab? You can just click on S.
So let's just see how many of you have already done. So can you see this particular notebook which is copy. So you should have copy in beginning.
Is it done?
OK.

Ajay Patel 22:10
Yes.

Tarun Jain 22:11
All right, so now you might see a new extension here. Earlier I told you we write code in Python right dot PY. Here if you notice you have IPY and B file. So this is nothing but interactive Python notebook. So this is an extension mainly whenever you're using Collab or.
Jupiter notebook. So Jupiter notebook is basically you run. It's a IDE itself, but you run it within your local system, right? But instead of installing that locally, we are just

using it on collapse and the extension is IPYNP, which is interactive Python notebook.
OK, so now that you have created you can just rename. I'll just write it as Python recap.
We will start off with the basic data types and after we cover basic data types we have some of the important control statements. So control statements again all of us are already familiar with. We have if conditions, while loops, for loops and all and then we have something called as.
Collections. So if you notice data type collection, this is where we'll write our own code. Before that, whatever we have, since it is common in most of the programming languages, I've just kept that as a template code, right?
So as of now, are everyone on the same screen?

Ajay Patel 23:34
Yes.

RamKrishna Bhatt 23:36
Yes.

TJ **Tarun Jain** 23:37
OK, so let's start off with the basic string operations because again you have integer. Let's suppose I'll just tell A equals to 10. Then you can do type of A. So here what we are doing is you have something called a type. So type will basically tell me.
What is the data type of that particular variable?
For example, if you see a, it is 10 which is an integer, right? So you will have class int.
Yes.
So it's connecting once it is connected, right? If you notice here green tick mark on the top right, that means you're connected to your Collab instance. And once you connect to the Collab instance, what you need to notice is you need to notice this green tick marks to those who are working with Collab.
Now, why is this green tick mark very important? Let's suppose I want to reuse this particular variable again. I want to just use add 10.
Equals to a plus 10 and then I have add 10. So basically what is happening here is if you want this a to be active in your entire Python shell, this green tick mark needs to be there. So just ensure whenever you're using collab after you write any piece of code and if you think that.

Is final for that particular cell. You can just run this particular code and as soon as you run that particular code, the first thing what collab will do is it will connect to the runtime and that runtime is marked with green tick mark and once that is done you have your code executable right?
And now if you see you have integer, same goes for other data type as well. I'll just write 10.0 you have float and I'll write true and in most of the Python programming if you want to define boolean.
It should start with capital T In most of the programming languages you have it as true which is in small letters, but in Python program it should be capital T.

Ajay Patel 25:39
Hmm.

Tarun Jain 25:49
So now if you see you have bool, I'll create different code cell.
And then you have string. So for string we have word. I will just type word.
Is this clear? The basic data types that we have in Python, we have numbers, we have strings and these two things are very common and it is available in all the other programming languages. So our main focus for today will be covering list.

Ajay Patel 26:17
Yeah.

Hardip Patel 26:21
Yes.

Tarun Jain 26:32
Dictionary tuples and sets and how they vary in terms of memory usage, right? So we don't just want to write code, we'll also try to understand which particular data type takes less memory, which of them takes more memory, right? So that will be our input to understand.
So now that we have covered some of the data types that we know, we'll get into basic string operations and understand some fundamentals of what are the indexing, what is slicing, and how much memory consumption does a string does.
Can you change the values in the string right? So that is where you have a concept

of.
Immutable. Unmutable.
So mainly what we need to figure out is are you able to edit or delete the given word that you have? If you can do that, that is called mutable, right? If you can't edit, delete or do anything, that means it is immutable. Strings are immutable.
Now what? What do I mean by strings are immutable? Let's say I define name to be Tarun.
But by mistake I wrote Tarun. It was supposed to be Varun, right? So let's suppose you want to edit my name back from Tarun to Varun. You can't do name of 0.
Equals to model, sorry V.
So now what you are trying to do is you already have a name and you just made one particular small mistake which is you changed V with T and if you just want to change or edit the given string, you can't do that. This is an error.
This means you can't edit or you can't delete the strings which is immutable. So now what we will try to do is let's understand the indexing, how indexing works and what is slicing right?
I'm using text again. You have indexing.
And you have slicing. So indexing is basically let's suppose you have a word and you want to understand at what particular index you have specific word, right? So that is where you'll use index. For best example, name of zero. What is present in this particular location?
What is present in the given location? So what is this location? The 0th index, right? This is mainly indexing now when it comes to slicing.
You will give a range zero to three, right? The range based indexing is basically slicing.
So whatever logic that you write here, the reason why I'm doing it for string is because this same logic will apply for tuple, the same logic will apply for list, and the same logic will also apply for dictionary. But there is a bit different when you're using dictionaries.
So for time being, just think indexing and slicing is same for string list and tuple. The memory consumption will be different, which we will cover in a bit.
All right, so here I have a word called develop and you can check the type of the word and then you have word of 0. So this is basically indexing right? Then you have word zero to two which is DE.
This is what you call a slicing. And now whenever you give zero to two, what is the

index? Let's suppose I print this word.

So can anyone tell me what is the maximum index that we have here?

Hardip Patel 30:33

3.

Ajay Patel 30:38

7.

RamKrishna Bhatt 30:39

7.

Mitesh Rathod 30:39

Of a second.

Ajay Patel 30:41

7.

RamKrishna Bhatt 30:41

Uh, six. Sorry, six. It will start from soon.

Tarun Jain 30:43

It is 6. So why is it 6? It starts with 0. If you notice here word of 0 is B right? So if I give word of 6, what should it print?

Ajay Patel 30:48

OK.

Hmm.

Hardip Patel 30:59

P.

Ajay Patel 30:59

B.

TJ **Tarun Jain** 30:59

It should be P That means your final or maximum index is 6. Now if I do 7 it should tell index out of bound error.

Ajay Patel 31:01

Hmm.

Hmm.

TJ **Tarun Jain** 31:10

You have index error. That means hey you don't even have seven. That means the maximum index is 6. Now the other way to understand this is you can just write length of word -1 is your final index. Now length of word is 7.

Why? Because if you want to calculate length, you will start from 1234 till seven, right? So these are the two things you need to understand. One is length keyword will print how much characters we have. Indexing is.

Ajay Patel 31:28

Mhm.

Hmm.

TJ **Tarun Jain** 31:42

Total characters -1 because it starts from zero.

Alright, so we started with indexing, which is, uh, the positive index.

Positive index means it starts from zero. You're going left to right and then you have slicing which happens left to right. Then you also have negative indexing. So now what will happen is if I give word -1 it will look at.

Which is the final index. It will start in reverse, so this is.

Reverse indexing or you can also call negative indexing.

So now if I print the word of -1 it will print P right? If I give word -2 it will look at what was the last but second character which is O.

Is this clear? Positive indexing, slicing and the reverse indexing. Now one logic what we need to understand here is.

Ajay Patel 32:35
Hmm.

RamKrishna Bhatt 32:41
Yes.

Tarun Jain 32:46
Whenever you are using slicing.
We use start and then we stop. So when we use this top right, let's suppose you define 024, but it will only print starting 4 characters, right?
Which is develop. Now whichever is at the 4th index that will be gone. So how much index do I have here? What is the maximum index?

RamKrishna Bhatt 33:16
3.

Ajay Patel 33:16
Uh, see.

Tarun Jain 33:17
It is 3. Why is it 3? Because I'll start with 0. I'll go till 4. This top is exploded. So one we need to understand now where is this used, right? Let's understand where this will be used.

RamKrishna Bhatt 33:17
B.

Ajay Patel 33:20
Hmm.
It's 0.
OK.

Tarun Jain 33:34
When we start working with text pre-processing, let's suppose you're building

sentiment analysis right now. What is sentiment analysis? You have a product and for the product you have multiple reviews from the customers.
And let's take an example of.
Hello, am I audible?

Ajay Patel 33:57
Yeah, yeah, you are audible.

Hardip Patel 33:58
Yeah, yeah.

RamKrishna Bhatt 33:58
Yes, yes.

TJ **Tarun Jain** 33:58
OK, I guess the screen was stuck, so I thought the Internet went off. OK, so oh, where was I again? Hello.

Ajay Patel 34:08
Uh, sentiment analysis. You are going to show us some example.

TJ **Tarun Jain** 34:09
OK, right, right. So let's suppose you have Flipkart, right? So Flipkart, you have a product mobile. So for that mobile you have positive feedbacks, you have negative feedbacks. Now what you're supposed to do is you want to build a system, AI system.

Ajay Patel 34:14
Hmm.

TJ **Tarun Jain** 34:26
Which will predict whether the given review is positive or negative, and I'll just write one feedback. This product was useful. I was.
Able to use it every day, right? So this is a positive feedback. But when you're feeding this data to an agent, or if you're feeding this to an AI model, you have to remove

the stop words, right? So what are the stop words now? Stop words are nothing but the common words. This is a common word. Was is a common word.
Common word I is a common word. So do you seriously think I this keyword is important in terms of of predicting sentiment analysis?

Ajay Patel 35:02
Right.

Tarun Jain 35:13
No, right. So sometimes what happens is there are few words.

Ajay Patel 35:13
No.

Tarun Jain 35:21
Which needs to be eliminated, right? And there in most of the times, this top words is a list.
And bunch of string elements. So this is where when you are working with text pre-processing, understanding the importance of slicing is very important, right? So this is where what I'll do is I'll try to relate to whatever we are learning with actual examples and where you will actually use them, right? So sentiment analysis.
In the in the text pre processing this syntax will be useful.
All right, so you understood. Whenever you are using slicing, you have two particular variables. One is start and one more is stop, but stop will be excluded. That means I'll go zero to four, but it will only have index till 3.
Right. So if I print this, I have zero and one, which is only one index. Two is eliminated.
I hope indexing and slicing is clear. We will repeat this in tuples and list as well to understand the memory usage and immutable, vixes mutable to see the difference.
But for time being this is how indexing and slicing works.

Ajay Patel 36:22
Yes.

Tarun Jain 36:42

OK.
Now that I said you that OK, strings are immutable, but how do we modify or add any words, right? So let's suppose I had a word which is develop, right? And if I want to make this word into development, right? So that is where what I can do is I'll have new word.
To word placement. So concatenation again. This is again a simple logic which we have already covered in other programming. Now if I click on new word you have development. So in most of the times what we usually do is do you know streaming?

Hardip Patel 37:21
Yes.

Tarun Jain 37:22
So let me give you an example. In ChatGPT, if you ask any prompt, you have one word, one token at a time. Are you seeing the entire response at once or do you see one word at a time? You see one word at a time, right?

Ajay Patel 37:33
Mm.

Hardip Patel 37:37
One minute.

Ajay Patel 37:39
Yeah.

Tarun Jain 37:39
So when we start working with applications like Streamlit. So basically what Streamlit is Streamlit will help you build a web UI, a based application. You have a Python code and without even knowledge of HTML and CSS you can build front end application.

Ajay Patel 37:43
Mhm.

Tarun Jain 37:57

Right for Python program. So in stream with application, if you want to add streaming, that means I ask a prompt and based on this prompt I'm getting the response I want to stream. Either you can use their own custom function. If not, what you can do is.
You have the final response.
Has a empty string and then what you can do is you can add one word at a time in that particular final response. So I'll show one example. Let's suppose I have a sentence.
Or I'll do something different.
Everyone knows what is follow or we cover this example later.

Ajay Patel 38:39
Yeah.

Tarun Jain 38:42
OK, I'm hope everyone was followed. So here what I'll do is I just have bunch of keywords.
Final.
Forward.
Product.
Now what you can do is let's suppose you have final.
Response plus equals to whatever is in your eyes. So here if you notice this symbol I plus equals to that means it is used for concatenation. Now whenever you concatenate you need to understand one logic.
Your existing word.
And the new word.
Needs to be string.
Now what does this mean? Here if you notice the word develop was a string, the new word whatever you're adding MENT, it is also string. Then only you can concatenate.
For here what I'm trying to do is you have final response. So what you are supposed to do in this final response is you need to just combine all these three words that you have in your response.
Now what are the words I have? I have final, I have cover, I have product right? But I'm starting with an empty response. Now if I run this and now if I print final response, you have final cover product. You can also give space if you need. So if you

want to give space again you can do less.
Or you can use something called as F string.
This will add certain space.
OK, the only purpose of showing this example now is this operator which is plus equals to. So plus equals to is mainly used for concatenation only when you have both word as string, so.
So how is it possible here? If you notice final response, what is final response here?

Ajay Patel 40:47
It's a string.

Tarun Jain 40:47
It's an empty string, but it's string. Now what you're trying to do is you're looping over these three words, final, cover and product. So first what it will do is it will add final inside final response and then final is also a string.

Ajay Patel 40:49
Yeah, yeah.

Tarun Jain 41:03
So if you want to combine them, you can use plus equals to which is a shortcut command. But if you give a space here, let's suppose you give plus then space, it's an error.

Ajay Patel 41:15
Mm.

Tarun Jain 41:16
OK, so it should be plus equals to.
Everyone understood the syntax for attenuation. You need to have two strings and then you can use plus operator, right? So this is 1 approach, the alternative approaches.

Ajay Patel 41:27
Yeah.

Hardip Patel 41:29
Yes.

Tarun Jain 41:45
You just use plus equals to. There is no space in between.
So whenever you have new string right, you're creating new string for password creation or just like as I showed for streaming during that time plus equals to is a common operator that we will be using.
And then you have repetition. So what happens in repetition is you have a word and then if you just use.
Asterisk symbol and then any integer. It will just repeat that for multiple times. So this is again one of the operations that is covered in string. It's not much useful, it is not covered in any other use cases, but where is this used? Let's suppose you're logging.
Right, you're writing the code and you want to log all the output or inputs that you're using. During that time you might have seen equals to 20 times.

Ajay Patel 42:30
Hmm.

Tarun Jain 42:44
Right, so this is just shorter command for using this kind of things, mainly for logging. You have step one, step two, step three. So after step one you have this kind of flow. You don't have to do this.

Ajay Patel 42:53
Mm-hmm.

Tarun Jain 42:59
Right. Instead of doing that, just use one single keyword.
Mark it 30 times, then use concatenation again.
Or it's better to use F string?

Mitesh Rathod 43:12
That's too.

Tarun Jain 43:16
I'll tell you what upstream works, so this is just one example.
So here what I'll do is in 230.
And I'll use word.
So in F string, if in case you want to have any variables inside that right, you can just use word. So if you notice here this word is already defined and if the word is already defined, it has to be closed within a curly brackets. So now what we'll do is I just want to give.
This equals into 30.
Not sure if F string is useful here.

Hardip Patel 44:21
We can use contact net with this and again do the other 30 so.

Tarun Jain 44:22
OK.
Hi So what we can do is we can have message which is.
Document loading right? So this is the message that you want to log. We want to log this message. I'll have a print.
F of message or we can use concatenate as Ardip said.
You have equals, repeat for 30 times and then have message.
And then again you can have plus equals 30 times.

Ajay Patel 45:03
M.

Tarun Jain 45:09
So this is mainly used for logging right? But most of the cases you will not find repetition.

Ajay Patel 45:12
Understood.

Tarun Jain 45:17
So are you currently, I mean in your collar notebook, are you currently present at this particular cell?
OK, so we complete the concatenation. The logic for concatenation is use plus operator and you need to have both variables as string. So you can call this as literals as well, right? So this word is a literal. Ment is a literal.

Hardip Patel 45:29
Yes.

Mitesh Rathod 45:29
Yeah.

Tarun Jain 45:44
And then you have place operator in between. So this is concatenation and then we have repetition. So for repetition your first keyword needs to be a string, then you need to have asterisk and then you need to have integer. You understood the syntax. In concatenation we have string.

Ajay Patel 46:00
Yes.

Tarun Jain 46:02
But for repetition, one is string and one more is a integer. You can't give any string here.

Ajay Patel 46:07
Hmm.

Tarun Jain 46:10

So if I print this, it's an error. So whenever you're using string with asterisk, it should be an integer.

Ajay Patel 46:12
OK.

Tarun Jain 46:24
I'll just write the logic here.
String. If you are using plus, the next thing needs to be string.
If you are using string with asterisk, the next thing should be a number.
Is this clear?

Ajay Patel 46:43
Yes.

RamKrishna Bhatt 46:44
Yes.

Hardip Patel 46:45
Yeah.

Tarun Jain 46:46
OK, so as of now what we have done is we covered string. In string we looked at indexing, then we had slicing. After that we had concatenation and then we had repetition, right? It is simple variables, it's simple operators, right? So now what we will do is we'll come with functions.
So what word did we had? We had new word.
I'll zoom my screen a bit.
So this new word is nothing but development, right? So now if you want to do manipulation with this particular words, you have some of the functions that we have. So for example you have capitalize which will convert the entire word into capitalize. That means your starting word, whatever you have will be in a.
Upper case. So if I just print this now if you earlier if you saw you have development which is entirely a smaller case. Now once I use capitalize your first keyword will be D which is development right? Now if I use dot upper what it will do is it will add

everything into upper case.
And then you have lower as well new word dot lower everything is in a lower case.
Clear, everyone understood. You just have a word, then use dot and if in case you get confused right new word.

Ajay Patel 48:01
Yes.

Tarun Jain 48:10
dot you'll have the suggestions over here. Well sometimes if you make any spelling mistake probably we might have used different function. So just use new word dot and colab. What it will do is it will give you auto suggestion just like any other programming languages as well.
So here what I'm doing is I'm selecting capitalize.
Now where is this useful? Let's suppose you're building any applications like this where you need to ask user to either type S or no or if you want to check the password as well. So usually what you can do is in Linux if you notice it will tell you Y or N.
Either give us or no, and in most of the cases what we do is we'll give smaller case Y even though upper case Y is accepted. So what you can typically do is here you have new thing right here. How many of you know what is input?

Mitesh Rathod 49:04
We're taking input from the.

Hardip Patel 49:05
Cancel.

Ajay Patel 49:08
This is more majorly used in a console, I guess input.

Tarun Jain 49:09
So.
So input is basically you ask user to enter their own data, right? So as of now, how many functions have we covered? We covered type.

Ajay Patel 49:17
Mhm.
Hmm.

Tarun Jain 49:22
What is print? So print will just execute or display the result.

Ajay Patel 49:23
Hmm.

Tarun Jain 49:31
And what does type do?

Hardip Patel 49:34
Uh, type of variables.

Mitesh Rathod 49:35
Printer type of variable string, yeah.

Ishan Chavda 49:36
To the Tango Tour Vishal.

Ajay Patel 49:36
Type. Yeah. What are the data types of variables? Yeah.

Tarun Jain 49:38
Data type.
Right, so here if I give new word it should give me string and here if I define new word it should print development right? So as of now we only covered these two functions. Now you have something called as input.

Ajay Patel 49:44
Mhm.
It will output here, yeah.

Tarun Jain 49:55
So what input will do is it will wait for the user to enter their input. So it's mainly used for user input, right? So if I tell enter your name.

Ajay Patel 49:55
OK.

Tarun Jain 50:13
Colon and if I just run this so it's waiting until user will ask or enter its own user input.
Here I'll just tell Tarun.
And then enter. So now if I print name.
I have that right? So input is basically whatever if you want or if you're seeking any user input that that's when we'll be using input. So we have covered three functions. One is type, then we have print and then we have input right? And now what I'm trying to do here is I want to ask.
Ask user input to proceed or not. Either they have to give Y or they have to give no.
So once I take any input, I'll run this separately.
Check.
If I also print no matter what user will enter, I just have to convert that into lower case so that you can use that inside your conditional blocks. So now if I run this.
Here I'm telling S which is Y with capital. Now if I just click on enter.
And now if I print on check it is S with small case. OK, so this is mainly used for password. If in case you're adding any password this thing or if you want user confirmation and then you have conditional blocks, right? So whenever you have conditional blocks.

Ajay Patel 51:34
Mhm.

Tarun Jain 51:37
With user input, you can directly convert that into load.
And then you can add the condition. Now here you have new method which is dot starts with. So what we are trying to do is check dot upper.
Can anyone tell me what is the output of this?

Ajay Patel 51:57
Everything will be in uppercase YES.

Tarun Jain 52:00
Yeah.
Now if I give capitalized.

Ajay Patel 52:03
Then one will be kept in.

Mitesh Rathod 52:04
Why capital and Y?

RamKrishna Bhatt 52:05
Why only by?

Tarun Jain 52:09
Allure.

Mitesh Rathod 52:09
Everything is slow.

Ajay Patel 52:10
Everything will be over.

Tarun Jain 52:11
Right, so if you notice all these three functions that we used, everything was empty brackets. We didn't define anything inside. Now if I use starts with, if I run this, it's an error because starts with needs some kind of input that you are giving inside. Now what you can do is.

Ajay Patel 52:18
Mhm, mhm.

Tarun Jain 52:29
You have checked. You want to know whether check is starting with Y or not, so the return type will be boolean.

Ajay Patel 52:41
Understood.

Tarun Jain 52:42
So far whatever we did, we used string. We checked with a function. The output is a string. So if you notice here, if I use upper, the output is a string. If I use lower, the output is a string.
But when I'm using starts with the output is a boolean which is either true or it is a false. So if I print this, what should be the ideal result?

Mitesh Rathod 53:09
True.

Ishan Chavda 53:10
2.

Tarun Jain 53:11
It should be true now if I say capital Y.

Ajay Patel 53:11
Hmm.
Balls.

Ishan Chavda 53:16
Pause.

Tarun Jain 53:17
It is false because this is case case insensitive. Sorry, case sensitive.

Ajay Patel 53:21
Is sensitive.

Tarun Jain 53:26
Is this clear? You can also check with.
And I can add PDF false right? So mostly in some of the applications if you want to test whether whatever user is uploading, if that particular file is PDF or not, if it is PDF then only you can proceed with your actual application. If not you can tell hey whatever application you have.
It's not PDF, right? It's an invalid input. So understood the importance of ends with and starts with. This is mainly used whenever you have a string and following with the string you want to cover any conditional base statements.
Right, so let me add a note.
So you have string.
Followed with conditional statement.
By conditional statement I mean if or follows. So based on the condition only if you want to trigger some action during that time you use these two functions which is starts with and ends with and the data type which it will return is a boolean.
Is this clear? All right, so here what I'm trying to do is I'm asking a user input. If we say S which is Y, no matter what you're doing, whether it is capital Y or smaller Y, I will convert that into load. So this is the first string operation.
First string operation which is dot load and then what I'm doing is I'm using dot starts with which is Y only when it is Y I'll do welcome. If not I'll do invalid or try again. If I just run this I'll write.
Write my name.
In valid try again. Since there is not a loop, what I'll do is I'll run this again. I'll do one.
Is this clear? These two functions, one is starts with and one was load.

Mitesh Rathod 55:24
Yes.

Tarun Jain 55:27
All right, so you can experiment with this keywords. Only capitalize upper, lower starts with, ends with. Then probably we'll proceed with the other examples.

Until here everyone have you executed your code.
So you can also try with some of your own words so that way we know what we are currently ongoing. So till here we have completed the string operation functions.
After this probably we'll get into other topics so we can take two minutes break, try to understand whatever we have done so far.
If you want to execute any piece of variable, let's do that and let's take any questions if you have.
So everyone knows what this escape keyword is as well.

Hardip Patel 56:24
Yes.

Tarun Jain 56:24
I guess it's in other program.

Ajay Patel 56:26
Yeah, it is common in. Yeah, I guess it is common in every language slash.

Hardip Patel 56:29
So in that there is a little input. I just I search for this when I was I started working for Python. It's like both of the PHP, so most of them are PHP developers for them.

Margi Varmora 56:36
But.

Hardip Patel 56:47
It's both of them are REPL like environment, so like uh read eval, print and then loop. So all of the commands that that runs is pretty much similar. So like uh here we have length.
Alien we have similar in PHP called STR line, so everything is very much pretty much similar. So even it is I think even better than JS if you are transitioning for from PHP to Python.

Tarun Jain 57:03
Yeah.

Ajay Patel 57:06
Yes, yes, two of them.

Hardip Patel 57:18
Just a little.

Ajay Patel 57:18
Yeah, yeah.

Tarun Jain 57:19
I I actually was working with JavaScript last few days so that I can relate to the examples what you guys know, but I didn't know that people will be from PHP background. But yeah, PHP I'm not worked with what I look at the syntax just to give example.

Hardip Patel 57:24
Right.
So yeah like yeah so if anyone wants to just transition you just like check what is at REPL environment like redevelop print loop. So it's it's just something like this like if we come to this like we do the lower.

Ajay Patel 57:40
Yeah.

Hardip Patel 57:54
Then we get read, then we eval that variable, then we print and it will be printed to the next command and which is like starts with and we take starts with then we read, we eval it. We just take first character and then we print that first character it will. Go either to the output stream or to the another variable. Everything is similar for both of the environment, even the Python and the because we have this just in time compiler.

Tarun Jain 58:27
Yeah.

Hardip Patel 58:30
Uh.

Mitesh Rathod 58:30
OK.

Tarun Jain 58:31
Uh, does anyone have the?

Hardip Patel 58:32
Just a little bit or whatever I know.

Tarun Jain 58:35
Yeah, I again I said right, let's have it round table wise. Even if you have any input you want to share, that's more than welcome because I might have my approach, people might have their own approach, right. Programming has multiple approaches. So till here anyone has any doubt in whatever approach we saw, we had positive indexing.

Hardip Patel 58:46
Bye.

Ajay Patel 58:47
Yeah.
Mhm.

Tarun Jain 58:55
Slicing, then reverse indexing. So let's just have some of the rounds right here. I'll use a word.

Ajay Patel 58:57
Hmm.

Tarun Jain 59:05
Or I'll use a new variable itself. I'll just tell.

I'll write one piece.

So an e-mail of -3. So can anyone tell me the output of this?

Mitesh Rathod 59:23

E.

Ajay Patel 59:23

E.

RamKrishna Bhatt 59:26

E.

Mitesh Rathod 59:28

A for element, yeah.

Tarun Jain 59:32

02.

Sports.

Mitesh Rathod 59:37

1.

Ajay Patel 59:37

One with space.

RamKrishna Bhatt 59:37

11 need space.

Mitesh Rathod 59:39

Only one.

Hardip Patel 59:40

No one without space.

Tarun Jain 59:41
One space, one space P.

Mitesh Rathod 59:45
Spatial or not cut.

Hardip Patel 59:45
One without space.

Tarun Jain 59:46
OK.

Ajay Patel 59:46
Sorry, space will be. Yeah, yeah, space.

Tarun Jain 59:47
The space, right? Unman space.

RamKrishna Bhatt 59:51
We.

Tarun Jain 59:52
So now can you tell me -3 -, 1?

Ajay Patel 59:59
OK, that's tricky 123C.

RamKrishna Bhatt 1:00:02
Easy. See you. We see.

Mitesh Rathod 1:00:04
And.

Ajay Patel 1:00:05
See.

Tarun Jain 1:00:06
Easy. You understood the logic. What is happening? You're going minus in reverse direction. It is -1, -, 2, -, 3. And now what we're trying to do is in the slicing is going left to right.

Mitesh Rathod 1:00:07
You see?

Ajay Patel 1:00:13
Mm-hmm.

Mitesh Rathod 1:00:13
This direction.

Ajay Patel 1:00:17
OK.
Hmm.

Mitesh Rathod 1:00:21
But.

Tarun Jain 1:00:22
ECE Why is it -1? Because it is at stop. You understood. So total this will be of what you call -3 to one. It has three length but it will skip one which will have only two length.

Ajay Patel 1:00:23
Mm-hmm.
Mm-hmm.

Mitesh Rathod 1:00:29
Yeah.

Ajay Patel 1:00:30
OK.

Tarun Jain 1:00:39
Is it clear the logic?

Ajay Patel 1:00:39
Hmm.

Tarun Jain 1:00:43
OK, uh.

Hardip Patel 1:00:43
Yes.

Mitesh Rathod 1:00:49
Oh.

Ajay Patel 1:00:53
Mm.

Tarun Jain 1:00:54
I'll take some example as well.

RamKrishna Bhatt 1:00:54
Yes.

Mitesh Rathod 1:01:01
And select.
Fonts.

Ajay Patel 1:01:05
What?

Tarun Jain 1:01:07
So the data type whatever ends with will print is a boolean which is false.

Mitesh Rathod 1:01:23
Pause.

Ajay Patel 1:01:23
Oh.

Tarun Jain 1:01:27
What?

Ajay Patel 1:01:28
OK.

Mitesh Rathod 1:01:29
Pause.

Tarun Jain 1:01:29
This is error.

Ajay Patel 1:01:31
Oh yeah, just a second. An e-mail dot lower start. OK, yeah, it starts with. It requires it. It should be a number.

Tarun Jain 1:01:36
So here what I'm trying to do is.

Hardip Patel 1:01:39
This will be false, no?

Mitesh Rathod 1:01:42
Yeah, it should be false.

Tarun Jain 1:01:44
Yeah, this will be false because anime is one piece. Anime dot floor will be OS. I was actually trying to add.

Mitesh Rathod 1:01:51
In the lower case.

Ajay Patel 1:01:53
Hmm.

Margi Varmora 1:01:54
What is weird? Yes, that was.

Tarun Jain 1:01:57
Oh, what happened?

Hardip Patel 1:01:58
This again will be false.

Tarun Jain 1:02:00
Points. What is this?

Ajay Patel 1:02:01
Yeah, of course.
Yeah.

Tarun Jain 1:02:06
The.
What will this print?

Mitesh Rathod 1:02:10
O in lowercase.

Ajay Patel 1:02:10
Oh, in a small case.

RamKrishna Bhatt 1:02:11
In small or in the lowercase.

Tarun Jain 1:02:12
All our case. OK, so this was just recap. Now if I want to just print a message.
Using F format string that one piece is best. How will I do it using format string?

Ajay Patel 1:02:33
F OK, double quotes, then double quotes, then in a curly braces enemy.

Mitesh Rathod 1:02:34
Uh.

Tarun Jain 1:02:34
F double quotes.

Mitesh Rathod 1:02:38
Anyway.
Space is best.

Ajay Patel 1:02:42
Is best.

Tarun Jain 1:02:43
So whenever we are using F string.
As soon as you start with upstring, make sure you start with brackets and then what you can do is if you have any variables that you need to reuse, just add curly brackets right this whatever you enter inside curly brackets should already be defined.

Mitesh Rathod 1:02:57
We get this list, yeah.

Tarun Jain 1:03:03
If not, you'll get name error, right? OK.
And uh.

RamKrishna Bhatt 1:03:08
So can we can we can we say that it is a similar to a string literally in JavaScript so.

Ajay Patel 1:03:15
String literal or templating normally.

Mitesh Rathod 1:03:16
Yes, templating.

RamKrishna Bhatt 1:03:17
Yes, template templating.

Mitesh Rathod 1:03:20
With the backticks.

Ajay Patel 1:03:21
Like this.

Tarun Jain 1:03:24
OK so here again this is very simple T it's nothing but you give a tab which is space then back slash N is nothing but you start with a new line right? So if I print this after my name there is a space then this surname and then I have back slash N So this is just to show how escape sequence.

Mitesh Rathod 1:03:29
Yeah.

TJ **Tarun Jain** 1:03:44
Works, but this is common in all the programming language common across.
And this is very important when we will work with files because in files if you read any input right, let's suppose I have a file here TXT. Whenever you load that particular file you will only see back slash T or back slash N.
So usually for file you'll have multiple lines, right? After every line you might encounter back slash N and in some cases you'll also see back slash T So when you see those two keywords, you just have to ensure OK, there is a space and there is an extend right? So this will come very shortly when we do file manipulation. Till here I hope everyone.
Are done right? The basic functions and all. OK, so now I'll come with immutability. So as I said, Python itself has memory consumption and many people tell Python is slow. Here what we'll do is we'll look at new function now.

Ajay Patel 1:04:28
Yes.

TJ **Tarun Jain** 1:04:45
So we saw print.
We saw type, then we saw input. Now what we'll do is we'll work with ID. So what ID will do is it will print the memory address.

Ajay Patel 1:04:52
Mm.

TJ **Tarun Jain** 1:05:03
Of the given string or object, right? So here I have new word. So new word is nothing but development. Now if I print ID of new word, this is a memory address of that particular word.
What I want you guys to notice is just notice this last three words, right? So for new word it is 352. Now for word it is 136. It's a different word. Obviously it's a different memory address.

Ajay Patel 1:05:24

Mm.

Mm-hmm.

Tarun Jain 1:05:35

Now here the word is same.

So just notice what is word? Word is developed. OK, now what I'm trying to do is I'm using word dot lower. The word is same but the entire consumption, whatever you're doing, the manipulation will create a new object or new address.

So now you have 440. Now if I use upper you again have new address. So what I'm trying to relate this topic to is.

Ajay Patel 1:05:59

OK.

Tarun Jain 1:06:06

Let me create a new code.

So here in Word, what I want to do now is I don't want my word to start with D

Obviously you can use dot capitalize, but if anyone goes with this approach D every single word will create a new object itself in string. So whenever you execute this.

It will throw an error that shows that strings are immutable. Why? Because every single time you do any manipulation with string, it creates a new object. So that's the reason why it's targeted as immutable. Shortly when we check with list, right? I'll just show one example list example.

Everyone knows Hari.

Ajay Patel 1:06:51

Mm.

Hardip Patel 1:06:53

Yes.

RamKrishna Bhatt 1:06:54

Yes.

Tarun Jain 1:06:55

Right, so I'll just show one example here.
ID of.
List example.
It gave me some word right? You have 568. Now if I do list of example, I want to start my zeroth index with zero. OK.

Ajay Patel 1:07:10
Mm.

Tarun Jain 1:07:19
Now if I print same ID of list example, it's the same right? Whenever you're creating list, whatever object you're creating doesn't need any additional memory. So what you can do is you can use same.

Ajay Patel 1:07:25
Hmm.

Tarun Jain 1:07:35
What you call whatever elements you have, this is called elements. One is an element, 2 is an element, three is an element and I want to change one of the element. So what will I do? This is indexing and I will change one of the index to 0. When I check the memory consumption of that, it's the same. That means if the memory consumption of any variable is same, it is mutable.
That means you can modify.
Add or delete. Now how this function works? We'll look at the list examples, but for time being you understood it. List can be mutable. Strings are immutable. That means once your object is created, it has its own memory. If you want to manipulate it, it will create a different memory address.
So This is why strings are mutable and sorry, strings are immutable and lists are immutable.
This is very important keyword point. I usually don't write note, but uh, once you have any template you can just go through this. It's very important.

Mitesh Rathod 1:08:41
Uh, one question.

Tarun Jain 1:08:42
OK. Yeah.

Mitesh Rathod 1:08:44
Even though if we extend like even though we're adding a any new elements into the list. OK, adding a new items into the list, still it will be the same member address, right? It will. It will not change.

Tarun Jain 1:08:57
Enlist. Yes, enlisted will be same memory address.

Mitesh Rathod 1:09:01
OK.

Tarun Jain 1:09:03
The address will be same, but the number of operations that you do for addition and delegation, the usage will be different, but the address is same. So what we are trying to do here is printing the address.

Mitesh Rathod 1:09:10
Mhm.
OK.
Mm-hmm. Makes sense.

Tarun Jain 1:09:19
So where will this be used? So usually when we work with neural networks, right? In neural networks, obviously we have to track every single shapes that we have, right? So during that time, in most of the cases we use ID.
Just to keep track of our memory consumption, because neural Networks like RNNS, they take too much of memory usage. So during that time we usually keep track of the memory either for neural Networks or again for logging.
All right. OK, so till here, are we done? This is the final example of strings. OK, any questions in strings?

Ajay Patel 1:10:03
Nope.

Tarun Jain 1:10:04
OK, so this is again very user friendly topic whatever we are starting now which is conditional statements. So basically conditional statements you have if you have a new concept in.
Python, which is elif. You have it, then you have elif. Else is already there. Else is already there in most of the program. Do we have elif in PHP by any chance this keyword?

Hardip Patel 1:10:28
Yes, it's as if it is as if it is as if without space.

Ajay Patel 1:10:29
No.

Mitesh Rathod 1:10:30
That's it.

Ajay Patel 1:10:31
Tells it.

Tarun Jain 1:10:31
Oh yeah, yeah, OK, I'll see it's not there in Java and C++. OK, here we have Elif and again the other two are same. You have for loop and then you have by loop. What we will do is.

Hardip Patel 1:10:36
So it is a keyword but as if.

Ajay Patel 1:10:48
Nope.

Tarun Jain 1:10:49

We will look at how you can use this in one liners in Python. That means you can just write everything in one line and.

6.

We'll also look at the importance of indentation.

Indent.

So indentation is mainly the space like every single time whenever you are working with if condition or conditional statements or function as soon as you use colon once you enter in collab it is just two space.

In Colab it is 2 spaces.

But in IDE it is 4 space by default.

And the idea is 4 space.

All right, so instead of this thing, I'll use this.

So here I have age and I'll have income and I'm just checking whether it falls under minor or.

Particular age exits. So here if I use 65 first it will check if condition. If no it will go to elif. But here if I make this as 16 it will look at if condition and then it will directly exit. Why? Because here I have elif.

So let me just check this.

So do you think this checking will be printed or not? So what is the output of this?

Mitesh Rathod 1:12:25

Minus.

Ajay Patel 1:12:25

Minor.

Tarun Jain 1:12:28

Miner. So now if I make this as.

Now what is the output?

Mitesh Rathod 1:12:41

Mm.

Working adult.

Ajay Patel 1:12:44
Working adult with good income.

Mitesh Rathod 1:12:47
Yeah, good income.
First explanation.

Tarun Jain 1:12:52
You know why it's checking added? OK, I didn't run this wait working added with good income. All right, so we can also write if condition in one liner. So what we can do is you start with if then you give the condition.

Mitesh Rathod 1:12:55
Oh yeah.

Ajay Patel 1:12:58
Um.

Tarun Jain 1:13:08
If this is satisfied, this is like terminal or operator we have right if condition.

Mitesh Rathod 1:13:14
Got nothing.

Tarun Jain 1:13:16
Yeah sorry. So what we do is if condition. If this is true, then whatever is added here this is true.
Right. And then you have else.
Call back.
Understood this index you start with if condition and if this condition is correct it will get inside the left hand side variables. If no then you can have an else condition and then you can have a fallback strategy. For example if age is more than 18 I'll just tell eligible.
Eligible to vote?

Not eligible.

Can vote eligible to vote? So what was the age?

20 huh eligible to vote. You understood this index for one liners. Now where this is used again when we start with.

Ajay Patel 1:14:15

Hmm.

Tarun Jain 1:14:21

Text reprocessing in most of the cases will write if condition in one liners itself. But again this is not that much of cell because you can also write the logic in this approach as well, right? This is just to add lesser lines of code till here everyone are done. It's like I'm just screaming through.

Hardip Patel 1:14:35

Yes.

Mitesh Rathod 1:14:36

And one question.

Hardip Patel 1:14:37

Yes.

Tarun Jain 1:14:39

Now again you have multiple conditions. Now here when it comes to multiple conditions.

For logical oerators.

In Python you have and you have or and you have not. If you want to represent this in variables right or numerical operator, what you can do is you can use and just single and. So for PHP do we have double and or single and?

Ajay Patel 1:15:06

Double end.

Margi Varmora 1:15:07
Double it.

Tarun Jain 1:15:08
So here probably we just have to use single and.
But mostly if you use single Android, it is not considered as logical operator.
It is considered as bitwise.
You understood and is used that you can. Let's say you have a equals to 10 or I'll write 2V equals to false.
So what will be the output of this A&B?

Ajay Patel 1:15:44
Calls.

Mitesh Rathod 1:15:45
Box.

Tarun Jain 1:15:45
It will be false. Here if you see and you can call it as logical operator, but technically in Python programming this will be considered as bitwise, right? So if you want to use actual logical operators instead of all this.
Special characters you have to give and.
You can directly write and if I give or it will be true and for not it should start with just one variable. If it is true, it will convert into false, vice versa.

Hardip Patel 1:16:16
Yes.

Mitesh Rathod 1:16:18
OK.

Hardip Patel 1:16:19
Yes.

TJ **Tarun Jain** 1:16:22

OK, in Colab it will not work because in Colab if you do asterisk, right, what Colab will do is it will download any package. So in Colab this particular syntax is mainly used for giving any instruction. Let's suppose if I do pip install lansing.

Hardip Patel 1:16:29

OK.

TJ **Tarun Jain** 1:16:38

So it will install that particular package if I give any Linux based commands PWD. So can anyone tell what is PWD?

Mitesh Rathod 1:16:48

So good to see level.

Ajay Patel 1:16:49

Show the current part.

Mitesh Rathod 1:16:51

Oh, sorry.

TJ **Tarun Jain** 1:16:51

Probably you might have seen this CWD current working time right? There is CWD it will tell content. So basically in collab whenever you do this thing exclamatory mark that shows the command.

Hardip Patel 1:16:53

Present work working directly.

Ajay Patel 1:16:56

Start working directly.

TJ **Tarun Jain** 1:17:09

You understood. So here we can just write not of K which is false. There is a difference.

Mitesh Rathod 1:17:11
Mm.

Hardip Patel 1:17:11
Mm.

TJ **Tarun Jain** 1:17:15
So obviously there is importance of accelerometer mark when it comes to VS code. It will be different meaning, but if you're starting accelerometer mark in colab it will consider it as command. So this command it can be if you're installing any packages.

Mitesh Rathod 1:17:15
OK.

TJ **Tarun Jain** 1:17:31
Or if you are giving any CD operation, CD in the sense.
CD then you can give some example folder. So what you have to do is inside content. If there is any example folder it will change the directory to example.

Hardip Patel 1:17:37
OK.

TJ **Tarun Jain** 1:17:47
So I hope everyone knows what is CD, right? Change directory, MKDIR. If you want to give those kind of commands in Colab, you just have to start with Excel metric.

Hardip Patel 1:17:51
Yeah.

Ajay Patel 1:17:52
Mm.

Tarun Jain 1:18:00
OK.
So here what I'm trying to do is I'm checking the username if it is Tarun and if the password matches and if user is active. That means if my account is currently active then only you're printing the logic successful. Here if you notice you have and you have and and then again you have and you have not and then you have or.
Right, so these are mainly the logical operators. The only message that I wanted to convey here is don't use bitwise operators because in most of the code bases people do use and or not. It serves the purpose, but it's not proper logical operators, right? You can't.
Properly manage this. So is this clear example?

Ajay Patel 1:18:40
Mm-hmm.

Hardip Patel 1:18:44
Yes.

Mitesh Rathod 1:18:45
So what is the recommendable?

Tarun Jain 1:18:45
OK.
Oh, which one?

Mitesh Rathod 1:18:48
OK, yes. What is the recommendable like using a like and or yeah, OK.

Tarun Jain 1:18:51
Yeah, this one and or not just the normal.

Ajay Patel 1:18:54
Having a keyword would be nice. Having a keyword would be good of.

Mitesh Rathod 1:18:56

Give it would be nice. OK, makes sense.

Tarun Jain 1:18:56

I keep on.

We got based instead of operators because there are two different meanings, right?

If you use operators, it will consider as bitwise.

Ajay Patel 1:19:00

Because if.

Mhm.

Mhm.

Hardip Patel 1:19:07

So like bitwise operators for PHP is similar to similar in Python, only the double and that we use double ampersand that we use in PHP here is replaced by A&.

Tarun Jain 1:19:22

Right, so this is a common bitwise operator. Nonetheless, what programming is C++, JavaScript, Java, everything. If it is single, it's bitwise. But since Python has both the support, it sometimes is misconfused as logical operator, which is not.

Ajay Patel 1:19:22

OK.

Hmm.

Tarun Jain 1:19:39

So for logic operator, as you said, double and is just replaced with a keyword. That's it.

Ajay Patel 1:19:39

Mhm.

Tarun Jain 1:19:48

All right, so this example was not specifically for multiple if condition. It was mainly focused on logical operators, right? So now this is for if elif and else you don't have switch. What do you call? We have switch right based on the user condition you switch to the particular.

Ajay Patel 1:19:57
Mhm.

Tarun Jain 1:20:08
Which is not available in Python. You have to use if and multiple and if statements.
Now we'll get inside loops.

RamKrishna Bhatt 1:20:15
Sorry, sorry to interrupt. Just one question. Do we have ternary operators like JavaScript? We can.

Tarun Jain 1:20:22
No, we don't have in Python.

RamKrishna Bhatt 1:20:24
OK.

Tarun Jain 1:20:25
OK, so just a recap, we had print.
Then we had type then input ID. So can we can anyone tell why ID was used?

Mitesh Rathod 1:20:41
To check the memory, yeah.

Hardip Patel 1:20:42
To get the memory address.

Tarun Jain 1:20:43
From my address.

Ishan Chavda 1:20:44
No.

Hardip Patel 1:20:45
Thank you.

Tarun Jain 1:20:46
This we are simple. Now we will look at range.
So range basically what it will do is you start with range keyword then you define 10.
So what this will do is it will go 0 till nine. OK if you define range.
Of 0 till five it will go zero to four. So typically by default if you are giving only end index, start index is 0.
Start index by default.
Is 0.
So you have single index, you have two index which is nothing but start and stop and then you also have range.
Two till 11 then two. So this is start and and step. So what this will do is it will print two, it will print 468 and 10. Why will it print 10?

Ajay Patel 1:21:56
Mm.

Tarun Jain 1:21:57
Because the end index is 11. If I give this as 10, it will only print 8.

Ajay Patel 1:21:58
This.

Tarun Jain 1:22:04
OK, the final index whatever you have will be ignored, which is excluded. So these are the three syntax you can note down. Range one is just single index. If you give single index by default your start index is 0.
If you give both start index and end index, it will start with that particular value and then with whatever index you give -1 that is 1 comma five. It will go one to four. Five

is excluded. Same goes for range. We start with two, then it goes to 10 and then you have step value.
Step value is nothing but how much step you need to give. So here you have 2-3 is one step, four is 2 step. Is it clear? This is start, then end start, then start, end step.

Ajay Patel 1:22:51
Yes.

Tarun Jain 1:22:56
And this is mainly used if you are using for loops. He could directly print range of 10. You will not see the values right either you have to run a loop. If not you have to use list.

Ajay Patel 1:23:07
OK.

Tarun Jain 1:23:14
So now you can see zero to 10 or you can use tuple. Now what these are values we'll cover shortly, but here it just list. Now if you want to create what is inside that we can also run a for loop like for run in range 7.

Ajay Patel 1:23:24
OK.

Tarun Jain 1:23:30
So till where will it print? It will start with zero. It will go to six.
Is this clear? I've also written the range range decoding number is nothing but end of index. If you are giving 2 numbers, it is start and end. If you are giving 3 index, it is start and end step.

Ajay Patel 1:23:38
Yes.

Tarun Jain 1:23:50
Just like the team number, if you see here I started with two, I'm going till 10 but 10

is excluded. Why? Because after eight it is looking for 9:00 and then 10, but 10 is at excluding value.

And you can also go backwards. If you want to go backwards, what you have to do is your start index should be high, your end index should be lesser value and your step should be negative. I'll repeat if you want to do reverse.

Looping your step needs to be negative.

And start index should be greater than.

End index. Now what is the start index here? It is 10. And what is the end index? It is 0. So let me give you one more example. If I give 4 here and if I give -3, OK, I'll give -1. Can anyone tell me the output?

Hardip Patel 1:24:56

4.

Ishan Chavda 1:24:57

Minus.

Tarun Jain 1:24:58

-4 then.

RamKrishna Bhatt 1:25:00

-3 then minus.

Hardip Patel 1:25:01

I I guess that's it.

Ajay Patel 1:25:03

Nothing. Where?

Tarun Jain 1:25:06

Can anyone tell me why it is nothing?

Mitesh Rathod 1:25:10

You should go for the first start with next to the greater there's nothing that's greater.

Tarun Jain 1:25:11
Read the sentence.

Ajay Patel 1:25:15
Shorter start that should be greater than index and index.

Tarun Jain 1:25:16
So let.
So which one is the greater index here?

Hardip Patel 1:25:24
-1.

Tarun Jain 1:25:24
Minus.

RamKrishna Bhatt 1:25:25
-1.

Tarun Jain 1:25:26
-1 is a greater index. If I give this as four, it will go 43210. It will not print -1.

Ajay Patel 1:25:28
Mm.

Hardip Patel 1:25:32
OK.
OK, so it the the step was amounted to 0, so that's why there was no.
Is that something?

Tarun Jain 1:25:48
Oh, can you repeat again?

Hardip Patel 1:25:51

OK. So I guess like the what do we say the end index was higher, so the minus amounted to 0 step in the third index of the range. So that's why there was no looping.

Tarun Jain 1:26:02
Right.
Yeah, if you notice here if I give -4, -4 is not greater than -1, so this is printed as none. So you also have variable called K equals to none.

Hardip Patel 1:26:06
I I don't know if I OK.

Mitesh Rathod 1:26:16
I see.

Tarun Jain 1:26:21
Right. So if I print K.
None. It's none. What if I just do K?
I can't see any output. Can you see that? No, because it's none. Probably in JavaScript you have none.

Hardip Patel 1:26:35
Yeah.

Ajay Patel 1:26:35
Hmm.
Well, even PHP, we do have null.

Tarun Jain 1:26:44
Yeah, so here it's just none. Whatever we did earlier, right? -4 is lesser than -1. So there was no loop. Everything is null. So for I will not even get executed. It's not even getting inside a loop.

Hardip Patel 1:26:50
Um.

Tarun Jain 1:26:59
So you understood this. Whenever you are using negative index, this point is very important. Start index should be greater than end index.

Hardip Patel 1:26:59
Yeah.

Tarun Jain 1:27:10
We'll take while loop as final example. After while loop is done, probably we'll start with data type collection tomorrow.
OK, I also wrote write your code here because today I just wanted to cover the recap. Tomorrow we can start with extra code. OK so while loop again we are starting with index value in while loop the syntax is while.

Ajay Patel 1:27:24
M.
OK.

Tarun Jain 1:27:37
Boolean right here, whatever Boole is there, you have to give a condition. So if that condition is true, then only whatever it is inside will be executed.
Executable.
Only when bool condition is true.
So now if I give bull.

Hardip Patel 1:28:15
Yes.

Tarun Jain 1:28:17
Condition invalid.
You got it. So whenever you're using while loop, just ensure whatever you're using your condition, it should be a boolean. Whereas whatever we used in follow, it was a range. You're defining the start range, you're defining the end range.

Hardip Patel 1:28:27
Yes.

Ajay Patel 1:28:28
Mm.

Tarun Jain 1:28:45
If the range makes sense, if it is true, then only you're able to print the results. But whereas in while loop whenever you're defining any condition, it should be a boolean itself. So this was just one syntax. Yeah, I had while then I I had a boolean condition. If that is true, this will be printed. Now can anyone tell me?
If I give this as true, what will happen?

Hardip Patel 1:29:06
The.

Ajay Patel 1:29:12
Executor only when boolean condition is true, but.

Tarun Jain 1:29:17
So this will be printed, but do you think it will stop?

Mitesh Rathod 1:29:22
No infinite that is.

Ishan Chavda 1:29:22
No, no, it's right. We're fine.

Ajay Patel 1:29:23
No, it it will print it again, right?

Hardip Patel 1:29:23
No.

Ishan Chavda 1:29:27
Infinite.

Tarun Jain 1:29:29
Why? Because there is no exit condition only. I told while loop through keep on going. There is no stop condition. So if you want to use stop condition, we have something called as break. Again, this is normal in other programming as well. You have break and you have.

Ajay Patel 1:29:37
Mhm.

Mitesh Rathod 1:29:41
Click.

Ajay Patel 1:29:46
Continue.

Hardip Patel 1:29:46
Continue.

Tarun Jain 1:29:46
Well, you have some continue as well, right? So here what I'll do is I'll just define break.

Mitesh Rathod 1:29:47
Continue.

Tarun Jain 1:29:55
Executable only when bool condition is true, right? So you had to have an example. I'm starting with employee count as zero, then I want to go till 10 and for the stop condition what I'm trying to do is I'm incrementing the value. Simple syntax.
So it will start with zero. It will go to 9. Why? Because I didn't give equals here.

Ajay Patel 1:30:17
M.

Tarun Jain 1:30:19
OK, now here what I'm trying to do is I'm trying to get inside a a what you call a.
This is mainly used for.
Oh.
Infinite loop.
But if the condition meets.

Mitesh Rathod 1:30:40
It look like.

Tarun Jain 1:30:42
You exit.
So what do I mean by this? So here what I'm trying to do is I'm asking username. If you notice here we are trying to use username and then you ask a input right? So this input is enter username. Now can anyone tell me what was the reason of defining this username first? Because you can directly use as well, right?

Mitesh Rathod 1:30:52
Empty.

Ajay Patel 1:31:03
Because.

Tarun Jain 1:31:04
You can also use this directly.
But why did I define it before while look? Can anyone tell me the reason?

Ishan Chavda 1:31:15
Oh, many dresses.

Hardip Patel 1:31:15
Otherwise condition would not be satisfied.

Mitesh Rathod 1:31:16
Think.

Tarun Jain 1:31:20
Oh, can you repeat?

Hardip Patel 1:31:20
In while otherwise the condition in while would not be satisfied, it will be username will not be assigned by.

Tarun Jain 1:31:31
Correct because here in while condition we need certain variable right? If I don't define this username is not defined, you'll get a name error. That means you have defined username. But here if you check your checking the condition.
Where is this username? So this is where what you're trying to do is you're defining an empty string. Then you're checking if it doesn't belong to Tarun. Then you are asking for the user input. Only when you have a proper input inside username you will end it.
So let's test this here. I'll give Deepak. OK, so now what is happening is I get inside this loop. I'm checking what is the input Deepak Deepak not equals to Tarun.
So Deepak.
Not equals to Tarun. So what is this?
What will this print?

Ajay Patel 1:32:33
Calls.

Ishan Chavda 1:32:33
Of course.

Hardip Patel 1:32:34
2.

Tarun Jain 1:32:35
It will be true because Deepak is not Tarun, right?

Ajay Patel 1:32:35
True.

Ishan Chavda 1:32:38
Excellent.

Hardip Patel 1:32:38
Yeah.

Tarun Jain 1:32:40
But if I give double equals.
Then it is false. But here what I'm trying to do is I'm giving Deepak not equals to Tarun. This is true. Now what is happening in the while loop? As long as no one enters my as long as no one enters my name, it will be.

Ajay Patel 1:32:44
OK. Yeah. Oh, sorry.

Tarun Jain 1:32:59
An infinite condition. So the better example what you can also do is if the username matches with Tarun then you can stop right? So if I run this again, OK, I already defined it. OK, can anyone tell me why it stopped?

Ajay Patel 1:33:19
Username equal to equal to blank.

Hardip Patel 1:33:22

So it is getting false and that's why there is no need to match condition. Oh, I mean like it is getting true.

Ajay Patel 1:33:25
And the.

Tarun Jain 1:33:27
Correct.

Ajay Patel 1:33:28
So initially it was, yeah.

Tarun Jain 1:33:30
See this is empty and then I'm giving double equals to Tarun. Now what is happening? It is false. It is not even getting inside the loop.

Ajay Patel 1:33:33
M.
Yeah.

Tarun Jain 1:33:39
We should do syntax. So here what I did was I I added not equals to so that there is an infinite loop.

Ajay Patel 1:33:47
Mhm.

Tarun Jain 1:33:48
Everyone understood what was happening. All right, now the better way. This is obviously not the best way to write the code. The best way to write code is you can use continue and break statements, right? So you start with while loop. This is the best way to start. You don't even have to worry about any operators, just define while.

Ajay Patel 1:33:51
Yeah.

Hardip Patel 1:33:52
Yes.

Tarun Jain 1:34:07
Then true. You don't have to give any condition, you just start with a simple while, give a boolean operator itself, then as the user input. Now if I give Tarun here.
Now Tarun not equals to Tarun. It won't even get inside this condition, so it will go to else and then it will print welcome. Once you find any user, if you have any function that you want to trigger, you can do that and then you can break if it is not Tarun.
Let's suppose if I give Deepak.
Then Deepak not equals to Tarun which is true. Then it will tell invalid try again. Is this clear what not equals to is doing? I'll just tell Deepak. Now what it will do is Deepak not equals to Tarun which is true.

RamKrishna Bhatt 1:34:49
Yes.

Tarun Jain 1:34:56
It's invalid. Once it is invalid I have continue. So what continue will do is it will go back to the loop which is our while loop again it will ask for enter username. Here I'll give Tarun.
So I give capital T will it accept or will it again tell invalid again?

Ishan Chavda 1:35:17
You have said.

Tarun Jain 1:35:18
It will accept, right?

Ajay Patel 1:35:21
Mm.

Hardip Patel 1:35:21

Try again.

Tarun Jain 1:35:24

I have lowered right? So if I give Tarun it will just tell welcome.

If I remove this word lower from here and now if I give Tarun it is invalid.

Ishan Chavda 1:35:36

OK.

Ajay Patel 1:35:37

Mm.

Tarun Jain 1:35:37

So understood lower also we used and we also used a continue and break.

OK.

And this is again one more example, but it is straight. You're starting with one of the value and then you're incrementing it. Once it reaches that particular level, you will stop. OK, you start with value, then condition, increment the value.

Once the condition meets, once the condition is false, you stop the while loop.

Till here everyone are clear. Anyone has any doubts?

So I'll just do a quick recap. We started with data types. We had numbers, we had strings, and once numbers and strings were done, we had different string operations.

So in string operations we also checked whether this thing is mutable or not.

And over the process we covered print, type, input, ID and range, right? So these are the few functions that we covered. Then we had indexing and slicing. One important thing about slicing is you have to have both start and ending index, right? But for indexing you just give one index.

Ishan Chavda 1:36:38

I do.

Tarun Jain 1:36:57

And whenever we are using concatenation, we just have to ensure that string plus. If

you use plus the another value should be string. If you use string with asterisk, the other value should be an integer. So this is nothing but reputation.
And then we had capitalize upper, lower. So whenever you are using this kind of values, the output is a string. But whenever we are using starts with or very starts with, if you are using starts with and ends with whatever value you have as your output is a.
OK.
And this example we saw.
And except escape sequence. The common to escape sequence that you will see in Python will be back slash T which is for tab space and you have back selection which is for next line and format string. You know the syntax F then quotes.
If the variable is already defined, add it inside a curly brackets.
This was covered. I guess I don't have to recap of conditional statements.
And yeah, range. This is very important. If you're starting with end index only one value, your start index by default is 0. And if you're giving both start and end index, that means it will start with whatever you have given, but it will end with whatever value is there -1.

Hardip Patel 1:38:05
Mm.

Tarun Jain 1:38:21
And if you want to give step then it will be start end and the step value. But if you want to give negative step value, the major syntax what you have to keep in mind is start index should be greater than end index. This is a very important.
Our condition and once that is done, we looked into while loop, which is again is a straightforward.
Yeah, so this was mainly for the quick start. Once the quick start is done, from tomorrow we'll start with Excel Python where there is no code available and there is no template for this.
So this is very important, the modules and package library, because when we work with rag and agents during that time, this is going to be very useful.
Till here it's very straight. I mean whatever you see at a dictionary set, it's simple data types and once data type is done, we have function and in functions we'll try to

understand the scope. We have global and local scope which is again common in other programming language.

Ajay Patel 1:39:13
Hmm.

Hardip Patel 1:39:13
OK.

Tarun Jain 1:39:22
And then we have exception block. The only major thing in Python recap will be file manipulation and modules because this is what we will cover in the upcoming part as well, drag agents and fine tuning because that is the whole main purpose of the entire session.
I'll also come to Varodara. So during that time we'll cover the entire NLP session because NLP online is very difficult. We'll have slides, we'll have code, so slides online, you can't manage 2 screens. So yeah, anyone has any questions?

Ajay Patel 1:39:48
Mm-hmm.

Hardip Patel 1:39:49
Oh.

Tarun Jain 1:39:55
Before we wind up.

Ajay Patel 1:39:58
Oh, nothing, I guess. Nothing from my side. Anyone.

Mitesh Rathod 1:40:01
No.

Ajay Patel 1:40:04
OK.

Ronak Makwana 1:40:04
No.

Mitesh Rathod 1:40:04
Thank you.

RamKrishna Bhatt 1:40:04
No.

Tarun Jain 1:40:05
OK, to those who know Python, probably this might be boring, but just wait for one more session. We'll have action code.

Ajay Patel 1:40:12
Yeah, how deep?

Mitesh Rathod 1:40:12
It's actually kind of similar. It's not boring, it's not boring, but kind of similar syntax we found here like compared to JavaScript and PHP.

Ronak Makwana 1:40:12
Uh.

Tarun Jain 1:40:14
I'll still recommend. I'll still recommend. Hold on one second.

Hardip Patel 1:40:14
I mean, it's not boring.

Tarun Jain 1:40:26
All right, so I will still recommend what you have to do is just use the same collab notebook that you have, try with different values and if you have any doubts, let us know in tomorrow's session and once that is done, probably we'll wind up Python basics in more two days.

Ajay Patel 1:40:33
Mhm.

Tarun Jain 1:40:43
The advanced and whatever intermediate is there, right? What you need to know for open source, then we will focus much right during that time. If we get any error, we'll do live coding as well because I don't want to have pre-made code because pre-made code sometimes.

Ajay Patel 1:40:43
OK.
Mhm.

Ronak Makwana 1:40:49
OK.

Tarun Jain 1:40:58
Both of us will not write, so let's have live coding when it comes to intermediate concept, so that way if we get error, we'll know how to resolve.

Ajay Patel 1:41:00
Mhm.

Hardip Patel 1:41:02
Oh.

Ajay Patel 1:41:04
No.
Yeah, makes sense.

Hardip Patel 1:41:08
OK.

Mitesh Rathod 1:41:10
OK.

Ajay Patel 1:41:10
Yeah, yeah, yeah, actually.

Tarun Jain 1:41:14
No, I know teaching basic. Sometimes it becomes OK. I'm going back. I'm coming.

RamKrishna Bhatt 1:41:14
It was very interactive.

Ajay Patel 1:41:19
No, no, this is good. This methodology is good, like the explaining some concept. Then after some time we are recapping this concept again. So yeah, it is helpful.

Hardip Patel 1:41:21
That's fine.

Tarun Jain 1:41:32
We'll also try to relate where it is actually useful because some topic, some Python concept, you will never use it, right? For example ID. If you're not working with neural networks, you will never see ID function, but still memory usage. I wanted to show that for immutability and mutable.

Ajay Patel 1:41:36
Hmm.

Hardip Patel 1:41:46
Oh.

Ajay Patel 1:41:46
Mhm.

Tarun Jain 1:41:50
There will be a few functions that you see in the code which will never be used, right?
But it's good to know.

Ajay Patel 1:41:50
Mm-hmm.
Yeah, no, no.

Hardip Patel 1:41:58
Yes.

Ajay Patel 1:41:58
I think.

Tarun Jain 1:42:01
OK. Any other?

Hardip Patel 1:42:01
Thank you.

Ajay Patel 1:42:03
No, I guess that's it.

Tarun Jain 1:42:05
OK, or do let me know if you have any doubts. I'll be sharing the GitHub URL once starting today's are done.

Ajay Patel 1:42:12
Sure, sure. OK. And what ID we should? I mean, right now we will be using the Colab only, right?

Tarun Jain 1:42:14
Oh, thank you.
Uh, for starting two days, we'll use follow up. Just give me one second.

Ajay Patel 1:42:22
OK.

Mitesh Rathod 1:42:42
And.

Tarun Jain 1:42:48
OK, so we'll start with VS code. Once we start with the multithreading asynchronous programming, asynchronous is not possible in collab. I mean it's possible, but you won't even able to notice the difference, right? So during that time we'll be using VS code.

Ajay Patel 1:42:51
Good.
Mhm.
Mhm.
OK.
OK.

Tarun Jain 1:43:05
If anyone is using Cursor, we can use Cursor as well, but I'll not recommend that at least workshops.

Ajay Patel 1:43:10
Yeah, yeah, for workshop session, we nobody will use a cursor.

Tarun Jain 1:43:14
Yeah, we'll use VS code and we'll use collab. 70% will be collab, 60 to 70% for fast API. There is one session on fast API, so that will be on VS code.

Hardip Patel 1:43:16
Mm.

Ajay Patel 1:43:17

Mm-hmm.

Mhm.

OK, OK.

Hardip Patel 1:43:25
OK.

Tarun Jain 1:43:27
What are 30% PBS code?

Ajay Patel 1:43:31
OK, OK.

RamKrishna Bhatt 1:43:31
And is it recommended to install any specific version of Python?

Hardip Patel 1:43:32
So.

Tarun Jain 1:43:33
OK.

3.10, 3.10, 3.11 is fine.

Ajay Patel 1:43:36
That is the latest.

Mitesh Rathod 1:43:38
Yes.

RamKrishna Bhatt 1:43:40
OK.

Tarun Jain 1:43:41
So by default, if you use VS code, you'll get 3.10 itself.

Hardip Patel 1:43:46
Mm.

RamKrishna Bhatt 1:43:46
OK.

Tarun Jain 1:43:50
Any other questions?

Ajay Patel 1:43:52
No, nothing.

Tarun Jain 1:43:53
So if you're using VS code, there is just one extension called Python. That should be enough and once that is done, probably we have to install PIP. So these are the only two things. One is Python extension and then installation of PIP which is Python package.

Ajay Patel 1:43:58
Mhm.
Mhm.

Hardip Patel 1:44:02
Mm.

Ajay Patel 1:44:07
OK, OK, everyone will install it, so no worries.

Tarun Jain 1:44:12
Yes.

Ajay Patel 1:44:13
OK.
Thank you, Darren.

Tarun Jain 1:44:15
Oh, but there's a session where we will see how to use this copilot tools.

Ajay Patel 1:44:21
OK.

Tarun Jain 1:44:21
Right, so you have one session on mermaid dot live. So what will happen is you create flow charts in mermaid, you attach that in your code base and then you use copilot which can improve your coding performance. So there are one or two session which is only on prompt engineering and tool usage. There is no code.

Ajay Patel 1:44:25
Mhm.

Hardip Patel 1:44:26
Mhm.

Ajay Patel 1:44:28
Mhm.
Mhm.
OK.

Tarun Jain 1:44:39
So we do have that in agenda as well.

Ajay Patel 1:44:39
Good.
OK, great, great.

Tarun Jain 1:44:43
No.

Ajay Patel 1:44:44
OK.

Mitesh Rathod 1:44:44
OK.

Hardip Patel 1:44:46
Do we have something with this more about the course? Do we have something with the Python environments like Conda or V or something?

Ajay Patel 1:44:46
OK.

Tarun Jain 1:44:58
Oh, Conda is not included, but we can have that installation if anyone is facing issues in installation, so we can have Conda setup as well.
But as of now in agenda, it was not added.

Ajay Patel 1:45:10
OK.

Hardip Patel 1:45:10
OK, OK. OK. No, I'm just asking.

Tarun Jain 1:45:12
Now we can do that if anyone is facing issues.

Hardip Patel 1:45:16
I I I went through a lot of problems while setting up the Python environments, so that's why I'm asking. Otherwise, yeah, no problem.

Mitesh Rathod 1:45:20
Oh.

Tarun Jain 1:45:29
Cool. Yeah, contacts. Sometimes if you're using Windows, it's very tricky to install it, but for Mac it should be borrowed.

Ajay Patel 1:45:30
Nobody is thinking.

Hardip Patel 1:45:36
Yeah.

Tarun Jain 1:45:38
Mac and Linux is easy.

Hardip Patel 1:45:38
Yeah.

Ajay Patel 1:45:40
I guess everyone has a Mac and Linux, so that is not an issue.

Hardip Patel 1:45:41
Yes.

Tarun Jain 1:45:43
OK, we can try that. If anyone is facing issue, we can have 5 to 10 minutes at the end to resolve it.

Ajay Patel 1:45:45
Yeah.
OK. Thank you, Tarun. Thank you for your time and it was a wonderful session. Yeah.

Hardip Patel 1:45:51
OK.

Tarun Jain 1:45:54

Yeah, yeah.

Mitesh Rathod 1:45:54

OK.

Thanks, Tony. Thank you. Okay, bye bye. Bye.

Tarun Jain 1:45:58

Yeah. Uh, looking forward to, yeah.

Ronak Makwana 1:45:58

Excellent.

Hardip Patel 1:46:02

Bye. Thank you. Bye, bye.

Ishan Chavda 1:46:02

Yeah. Bye. Bye. Bye.

Ronak Makwana 1:46:02

Goodbye.

Ajay Patel stopped transcription