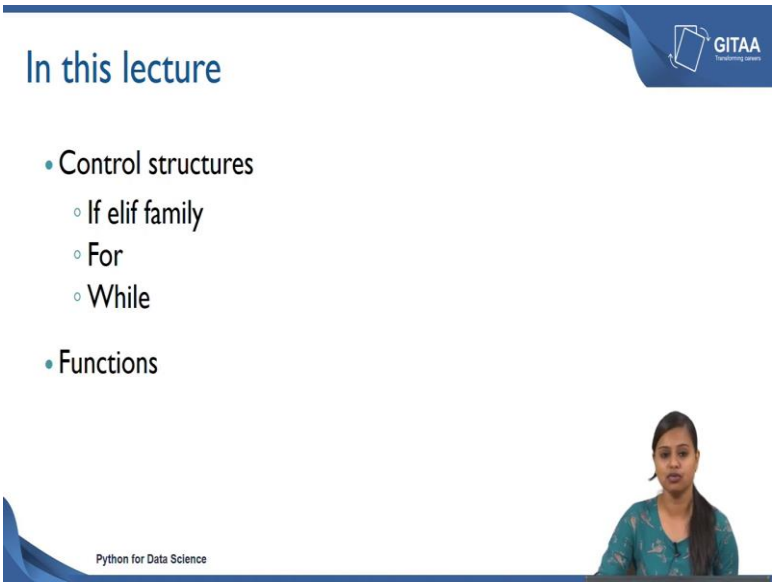
Python for Data Science
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 21
Control structures & Functions

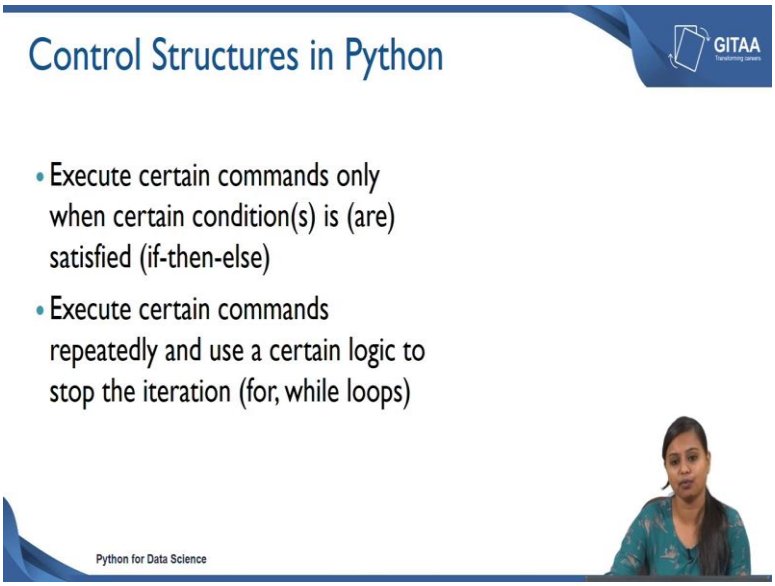
Hello all welcome to the lecture on Control structures and Functions.

(Refer Slide Time: 00:19)



So, in this lecture we are going to see about the control structures. And under the control structures I am going to cover if elif family, for loop and while loop. And, after that we are going to also look at the functions, what do we mean by function and how do we define a function using Python?

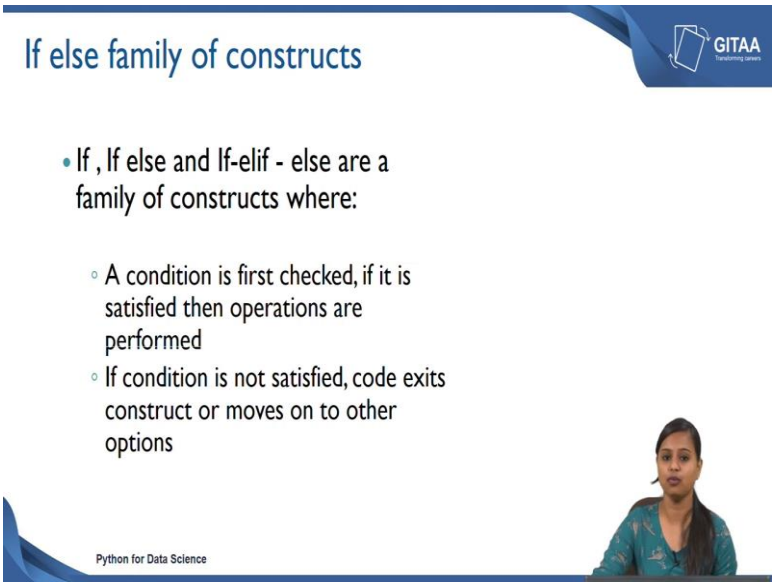
(Refer Slide Time: 00:35)



So, let us start with control structures in Python, whenever we mean control structures I am going to use if elif family for loops and while loops. Let us see where do we use if elif and for loops and while loops. So, whenever you want to execute certain commands only when the certain condition is satisfied.

So, in that case you can go for if else statements, the condition can also be single or you can also give multiple condition, in that case you will have multiple else statements. The other one is when to use for and while loops wherever we want to execute certain commands repeatedly and use the certain logic to stop that iteration, in that case for and while loop will be helpful.

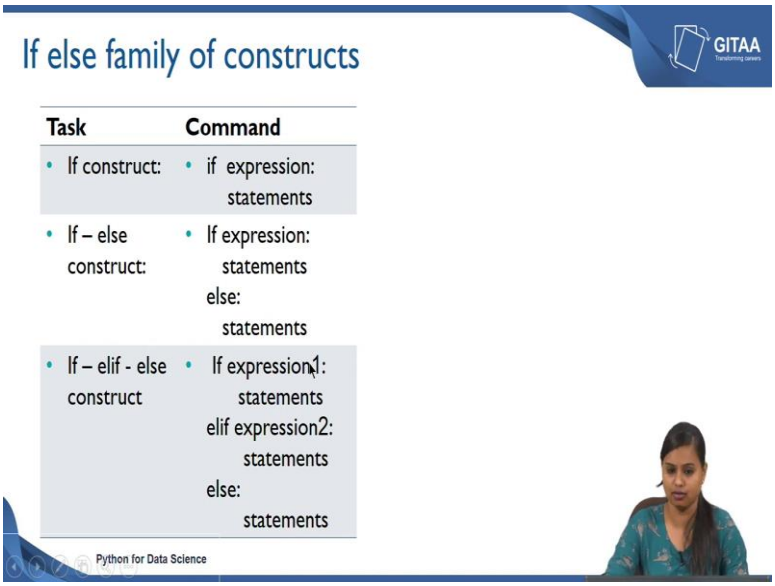
(Refer Slide Time: 01:15)



So, first we will look into the if else family of constructs, if else and If-elif-else are a family of constructs, where a condition is first checked, if it is satisfied only then the operations will be performed. If, the condition is not satisfied the code exits the construct or moves on to the other options. So, whenever we use just an if statement or with an else statement or with using multiple if's and multiple else clause.

The first check would be the condition, whenever the condition is satisfied only then the code will be executed or the statement will be executed, otherwise the code exits the construct itself and moves to the other options. So, that is how the if else family of the constructs works.

(Refer Slide Time: 02:01)

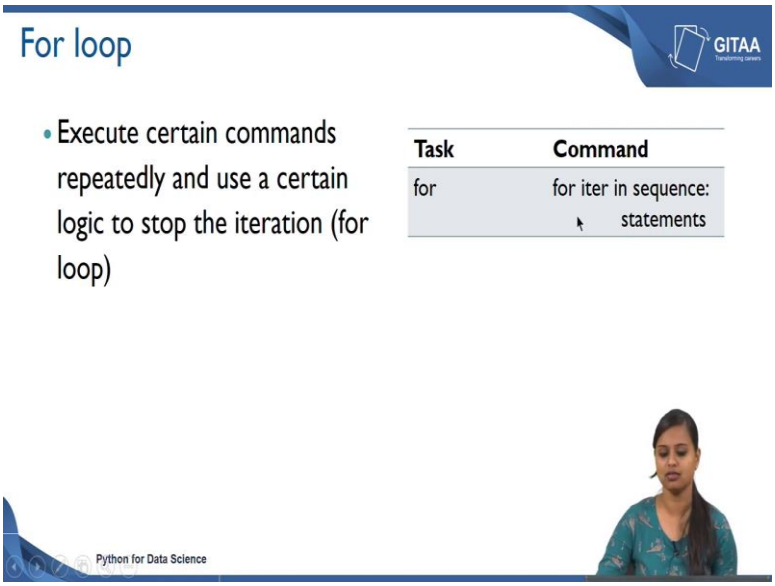


Let us see different task for each construct. So, first we will look into if construct, the command would be if expression colon and statements in the next line. If is a key word, if the condition is satisfied whatever condition you have given it under the expression, then the statements will get executed. Otherwise, the code exit the construct itself. Next, we will move ahead and see what is the syntax would be for If-else construct. It forms a basis from the if construct, wherever we have given the first statement, using the if keyword and followed by if keyword you have to give the expression to be checked, that is where the condition to be specified.

And, after that you give the statements that needs to be executed, if the condition is satisfied, if the condition is not satisfied give another statement under the else clause. So, using If-elif-else construct you can basically give multiple condition that needs to be checked in order to execute a statement or in order to execute any line of code. In that case the command would be like, if expression1 is being satisfied, then execute the statement, if it is not satisfied then execute the next condition.

If that is not being satisfied, then it also comes to the next else clause, where the that statement will get executed. So, the else clause statement will get executed only when the other 2 expressions are not satisfied. Basically, the other 2 conditions are not satisfied.

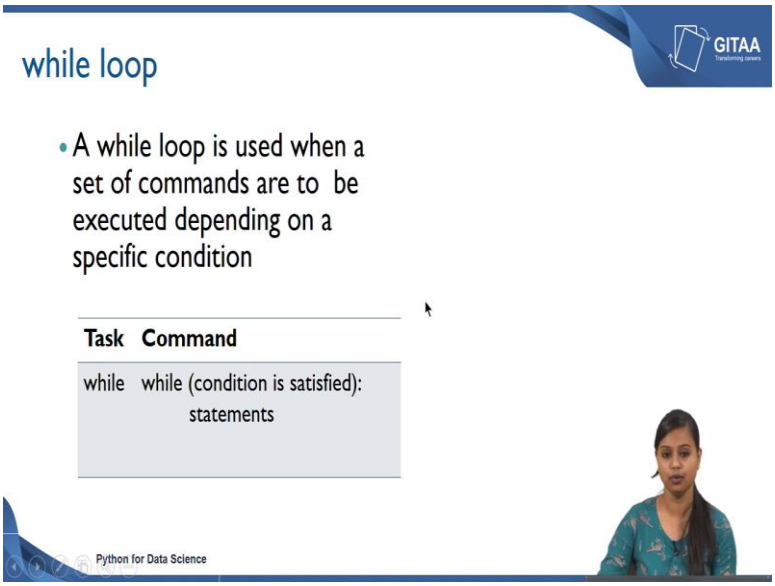
(Refer Slide Time: 03:37)



So, now we will look into for loop, now we are going to look into the syntax and the different loops here, in the upcoming minutes we will be looking at in terms of an example also.

So, first we will look at the syntax for for loop. So, whenever you want to execute certain commands repeatedly and use a certain logic to stop the iteration you can go for for loop. Let us look at the syntax on how to use the for loop, the command should be like for iter in sequence colon and followed by statements in the second line. So, this is the simple command that is used to construct a for loop.

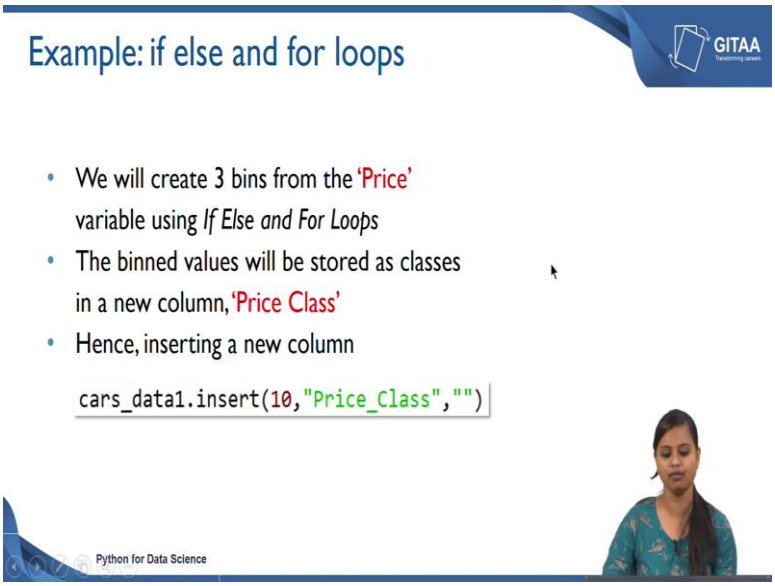
(Refer Slide Time: 04:13)



Next, we will see about the while loop and the syntax for while loop: a while loop is used, when a set of commands are to be executed depending on a specific condition. Basically, a while loop will be executed as long as a condition is true, whenever the condition becomes false, whenever the condition you have given becomes false, the while loop execution will stop.

So, the task being here is the while loop and the command to construct a while loop is while is the function inside the braces, you have to give the condition. And, as long as that condition is true the statements we have given in terms of statement will get executed, whenever that condition becomes false, that is when the while loop will get stop executing.

(Refer Slide Time: 04:57)

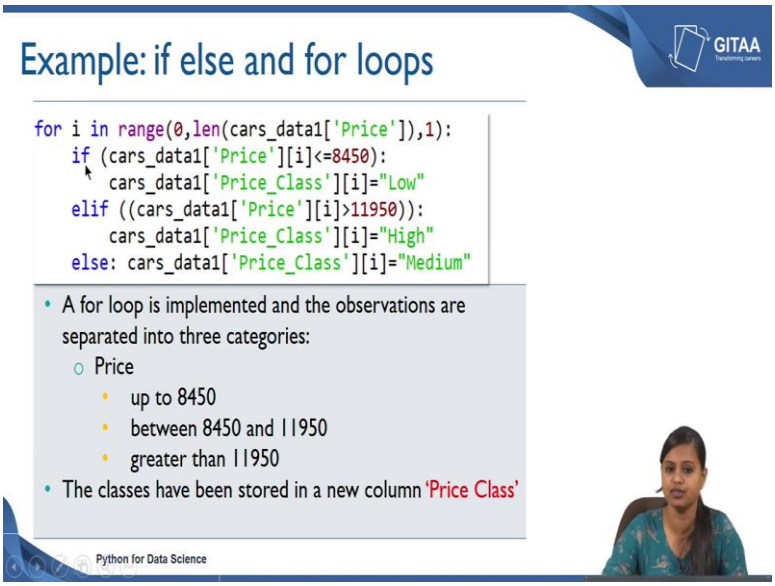


So, now we are going to see an example to know how we can use if else and for loops. So, here we have been working with the data set called Toyota, where we are seeing how to read them and how to do basic pandas data frame operations? From there I have just used a single variable called price, which represents the price of the cars, price of the pre-owned cars. So, I am using the price variable from the Toyota data, where I am going to create 3 bins from the 'Price' variable using if else and for loops.

Because, the price variable is a continuous variable where it just has the values for the price of the pre-owned cars, if I want to segregate those prices into 3 buckets, then I can use if else and for loops, let us see how to we do that. The binned values as I mentioned I am going to bin the values. So, those binned values will be stored as classes in a new column as price class. So, in that case I should be creating a new column to the existing DataFrame.

So, now, I am going to create a new column to the existing Toyota data, where I have read and kept it as cars_data one. So, using the .insert function, I can give the position to which the column should be added and the column name and I have given blank. So, that all the it will create a column with the blank values. So, now a new column has been created as price class so, that we can store all the bin values as classes.

(Refer Slide Time: 06:25)



So, using if else and for loops, I am going to convert all the values into 3 categories, where the categories represent the range of the price; one is being “Low” and other one is being “High” and other one is being “Medium”. So, if I want to segregate those values into 3 categories I want to basically give some condition in which it can happen. So, I am going to use that using if statement.

So, what I am starting here is I have used the for loop I have initiated the for loop here, where I have given the iter in sequence, where I is the indexing variable in the sequence. I have given the sequence using the range function, where I have given the starting value as 0 and it should have the value till the length of price. So, the value will be 1436 and I have given comma 1, then it will the iteration will happen in the steps of 1.

So, now I have given from which the iteration should happen. Now, I have given the sequence in which the iteration should happen. Now, this is time to give the condition on price. The first condition being I want to make I want to make some records, I want to bucket some records as loop by giving the condition, whenever the price is less than or equal to 8450, then keep them as in the range “Low” right.

In that case I can just give a condition, saying access the price variable from the cars_data 1 dataframe and give a condition here, whenever it is less than or equal to 8450, then execute this statement that is cars_data price class becomes “Low”.

Basically, equate a value “Low” to the new variable price class. So, this will happen for each and every row and it will check whether the price of the car is less than or equal to 8450 or not. If it is then it basically equate them equate the basically then it will name the row as “Low”. So, there is another condition using elif statement, because I need to give 2 conditions here; one is on “Low” and one is on “High”. And, whatever is not being satisfied with “Low” and “High”, I keep them as “Medium”.

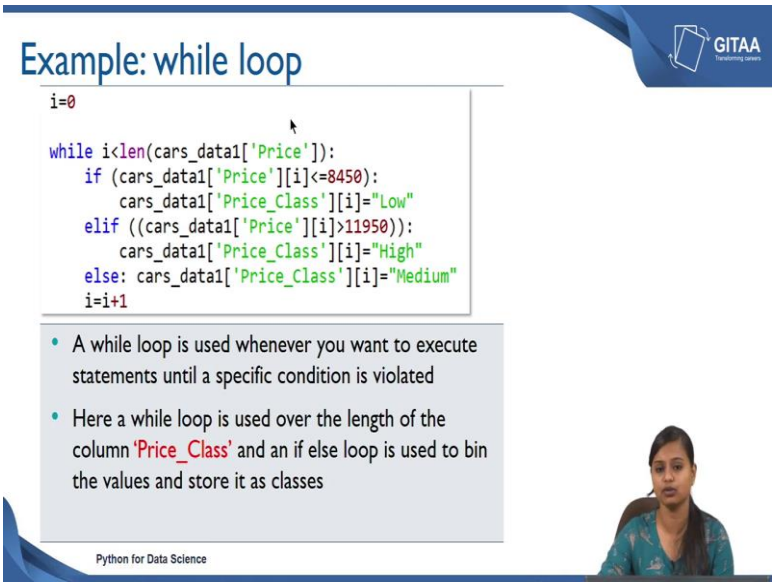
And all those rows will be kept it as “Medium”. So, in this case I have 2 conditions here; one is price less than or equal to 8450. And, the other one is being whenever the price is greater than 11950, then in that case the price will be, in that case those rows will be named as “High” and that will be stored in the column price_class. So, whenever these two conditions are not satisfied, whenever any row is greater than 8450 and less than 11950, then all those rows will be named as “Medium”. So, there is a bound here.

So, the whatever rows that are being named as “Medium” will have all the rows wherever the price is greater than 8450, and wherever the price is less than or equal to 11950. So, this is how I can give using for if elif and else. So, if you see that to just to summarize whatever we have done it in for loop and if else, a for loop is implemented and the observations are separated into 3 categories right now.

So, the price being upto 8450 and between 8450 and 11950 and greater than 11950, and we keep the price less than or equal to 8450 as “Low” and we keep between 8450 and 11950 as “Medium” and whenever the price exceeds 11950, then we keep them as “High”. And, we know that the classes have being stored in a new column called price class.

So, in each of the records of price class there will be a label called row “Low” “High” or “Medium” that is exactly based on the condition, which is being represented here. And, you have done that so, using for loop we have seen how to do the iteration and using the if and else clause we have seen how to give condition based on a variable?

(Refer Slide Time: 11:01)



So, next let us see an example for while loop. So, a while loop is used whenever you want to execute statements until a specific condition is violated. Here, I am going to use a while loop, over the length of the column price class, and an if else loop is used to bin the values and store it as classes. So, whatever we have done it in the previous slide I am going to repeat the same thing using while loop. So, you can do that using both for and while.

So, here I have initialized my indexing variable as 0, the difference between both for and while loop is in the previous slide, you would have given your iteration step in the sequence itself using your for, but in this while loop you are giving your you are just initializing your indexing variable as 0 and you will give the iteration step at the last only. Because, the first check of the while loop is the condition check. So, the while loop will be executed as long as i is less than length of cars of data that is 1436, the while loop will get stop executing whenever it exceeds that condition.

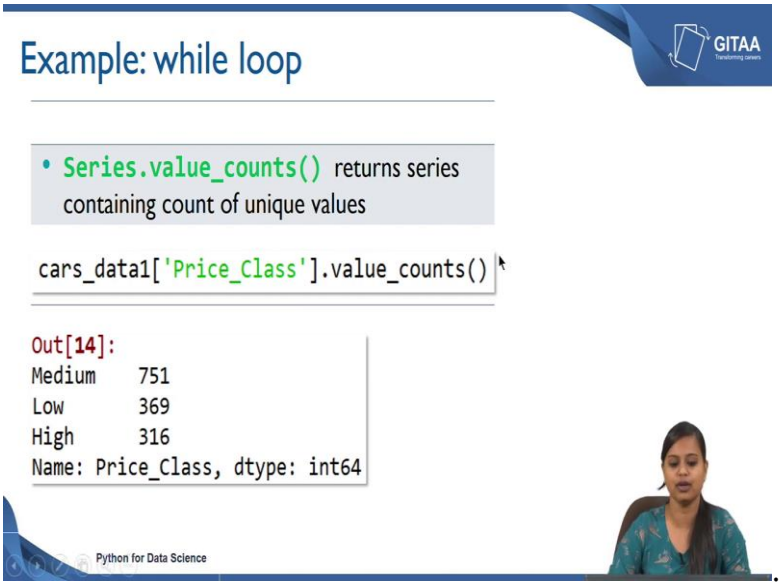
Whenever your i becomes 1437 your while loop will get stop executing. So, next I have a condition here, the same condition being represented here with the same if elif and else clause the difference being here is you give the iteration steps at the end of the loop, whenever you are using a while loop and you give the iteration steps at the beginning of the loop itself using a for loop. So, here if you recall whenever the price is less than or equal to 8450 then keep them as “Low” under the column price class.

And, whenever it exceeds 11950 then keep them as “High” under the variable price clause and whenever both the conditions are not satisfied, whenever the price is greater than 8450 and less than or equal to 11950, then in that case that observations will be named as “Medium”. So, now, we have seen the examples for both for loop and while loop to basically bucket all the price values into 3 categories as “Low” “High” and “Medium”.

There might be other functions which will do it thereby there are so, many inbuilt function that does this, but using a for loop when you have a control on whatever you are doing with the steps, then you can use a for loop and while loop. We have now 3 bins, now 3 categories now “Low” “High” and “Medium”. So, we do not know how many records fall into row, and how many records fall into “Low”, and how many records fall into “High”, and how many of them fall into “Medium”?

So, let us just see how you are observations have been categorized? So, now, we have basically used a loop to combine all the price values into three categories; one as “Low” “Medium” and “High”. Now, I want to check how the categorization has happened.

(Refer Slide Time: 14:11)



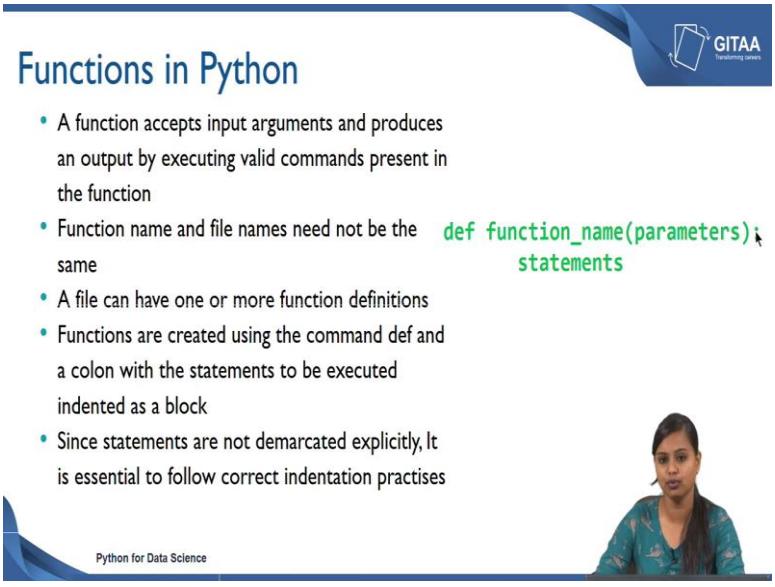
So, in that case I will be looking at frequencies of each categories, that is “Low”, “Medium”, and “High”, I can get the frequencies of each categories using value_counts function. So, whenever you have a series you can use value_counts that basically returns series containing the count of unique values.

I want to check the count of unique values under the price class, that is the variable name and cars_data one is the dataframe name, and.value_counts is the function, if you use that it basically gives an output which is shown here. So, you have 751 observation that falls under the “Medium” category that is basically those cars are in the range of “Medium”, those cars have the price in the range of “Medium”.

And, you have 369 observation with the “Low” range and you have only 316 observation with the “High” range. And, the name being price class and the data type of the output is being in 64. So, now, we will see how to basically convert your numerical values into a categorical variable right, because now we have converted the numerical variable price into the categories as “Low”, “Medium” and “High”, that becomes a categorical variable.

Now, that is why we have checked the unique count of each categories, see you can if you want to do that you can use either a for loop or a while loop. So, this is where all the while loop and for loop comes into play.

(Refer Slide Time: 15:43)



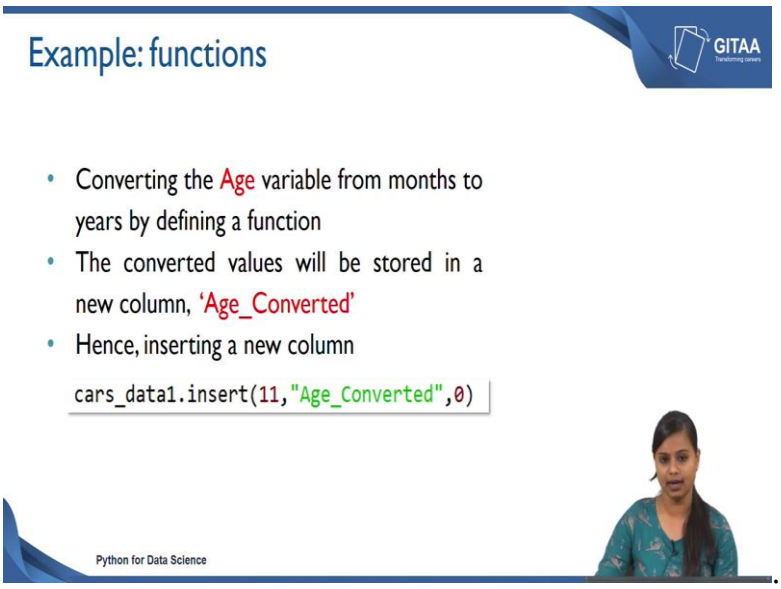
Let us see about the functions in Python. Basically a function accepts input arguments and produces an output by executing the valid commands present in that function. And, the function name and the file name need not be of the same, because you can have a different file name and a different function name, that holds good in Python.

And, a file name can have one or more function definition. Say, if you have a file where you have defined a function, that function file can have one or more function definition, there would not be any issues while you are calling a function in a different file. And, the functions are created using the command def and a colon with the statements to be executed indented as a block.

So, this is how you define a function you use a def function and the function name is followed by that, say inside the parenthesis you basically need to give the parameters based on which your calculations will be done. So, using the statements here you will give the basically an equation or an expression, that should be calculated that should be solved based on the parameters, that you have given inside the function name parameters.

And, the since the statements are not demarcated explicitly, it is essential to follow correct indentation practices. Because, other programming languages the Python does not support the curly braces, for any control structures or function rather it uses the indentation to explicitly show the demarcation. So, the indentation should be followed. So, your statement should be exactly slightly away from your first three letters of your function name, whenever you type a colon and give an enter, it will automatically comes with an indentation. So, it is suggested to not to change that indentation.

(Refer Slide Time: 17:33)

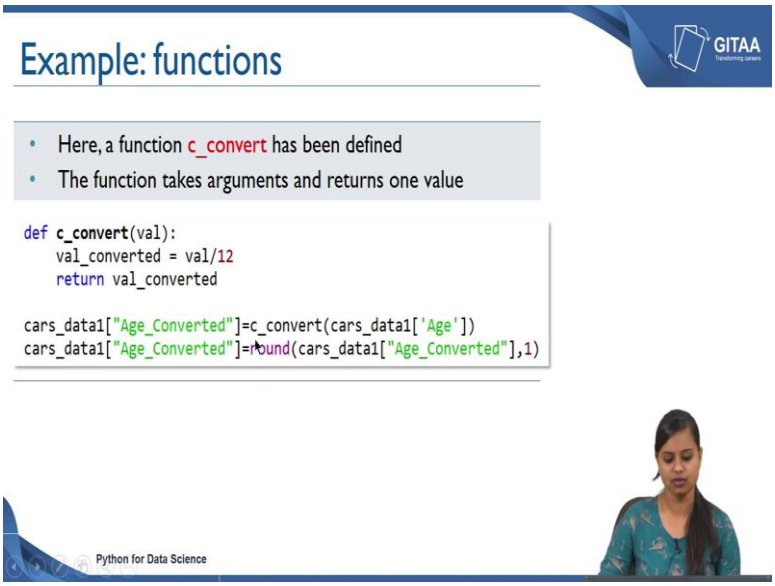


So, now let us see an example on how to define a function? So, we are going to define a function which will allow us to convert the age variable from months to years by defining a function. I am also using the age variable from the Toyota data, that we have that we have working with. So, in that case age is being represented in terms of months, but that does not convey me the exact information or the efficient information, but that is not the exact way where I can infer the age, in rather than keeping the age in terms of months I can also keep the age in years.

So, I want to convert the age variable from months to years by defining a function. So, the converted value should be stored in a new column. So, I do not want to touch the existing column rather I am going to store all the values in the new column called age_converted.

So, I should be creating a new column called age converted now. So, I have created age converted using the same.insert function, where I want to keep the age converted in the 11th position and initially I want to have all the values as 0, that is what I have given as 0 here. So, once you have executed this line a new column will be created as age converted.

(Refer Slide Time: 18:45)



So, now let us define a function to convert the age from months to years. So, here I am defining a function c_convert and the function takes arguments and returns only one value. So, def is the key word that is used to create a function definition or the command

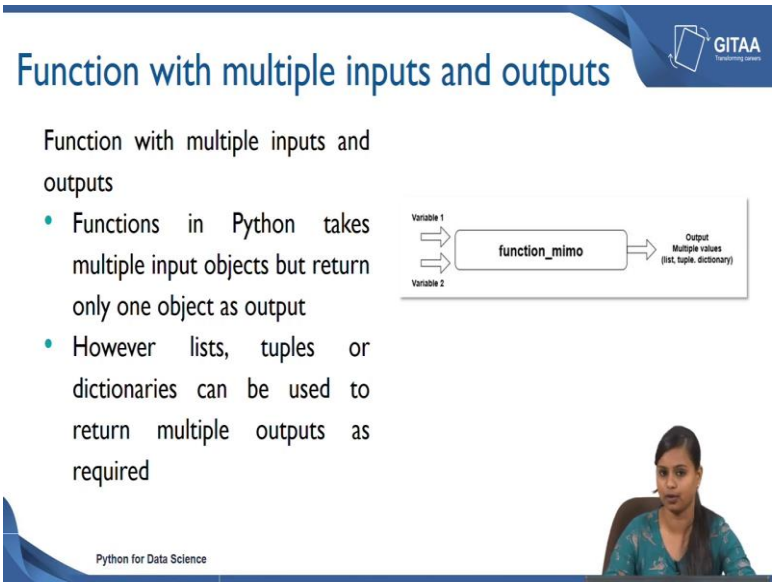
that is used to function, that is used to create a function definition. And, the function name is c_convert and the inside the function I have given the argument called val and followed by the semicolon in the next line I have a variable called val_converted that represents value converted.

How am I going to convert I am going to convert that value by dividing, whatever value that is given here by 12. So, this is the default argument. So, whenever you called the any function you can use the c convert and give a variable there or give a value there, that will be divided by 12 and it will return the value converted val. And, it will return the value or that is stored in the val_converted. So, now, using this function definition, I can basically convert the age variable into months, I can basically convert the age variable from months to years.

How, I can basically use the same function here that is c_convert and inside the function I just basically need to pass in the arguments for val to divide any number by 12. So, I want to divide all the numbers from age by 12. So, I have given cars_data1[‘Age’]. So, all the observations under the age column will be divided by 12. So, there I get the age converted to years. So, if you see I am storing that into the new column called age_converted from the data frame cars_data.

So, this will this new variable will have all the age that has been converted into years. And, whenever you are doing any numerical operation Python always comes with 5 to 6 decimal points, I do not want to have a variable which has so, many decimal that also does not convey the exact information of year. So, I want to round it off to only 1 decimal. So, that it does not have 5 to 6 decimal point it will just have a digit after a decimal point.

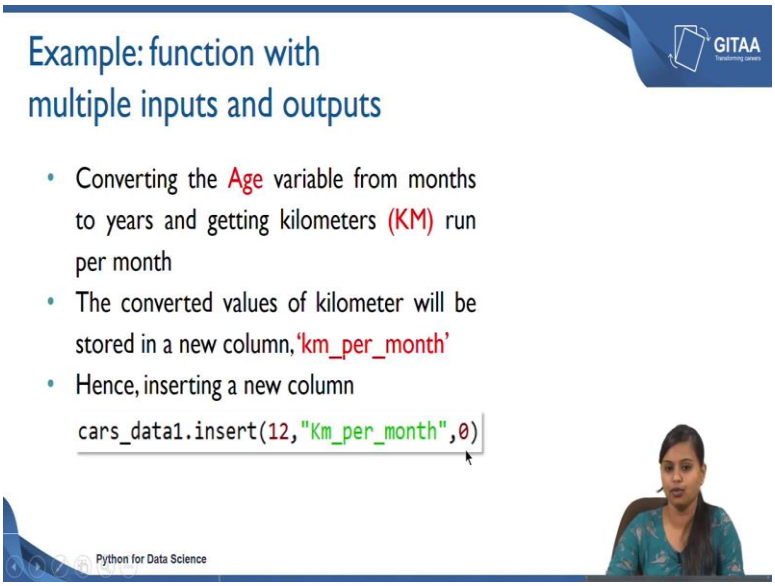
(Refer Slide Time: 21:11)



So, till now we have been seeing about a function which accept a single argument or multiple arguments and which will arrive at a output value single output value. Now, we are going to move ahead and see how to define a function with multiple inputs and arrive at a multiple outputs. So, function in Python takes multiple input objects, but return only one object as output. So, you can have variable one and variable 2 as inputs to your function.

But, your output will be in a form of only single object, but that object can contain multiple values like a lists can contain multiple elements and a tuple can contain multiple elements and a dictionary can have multiple keys in values. So, in that case you will have multiple results in form of a single object. So, like I said list tuples or dictionaries can be used to return multiple outputs as required.

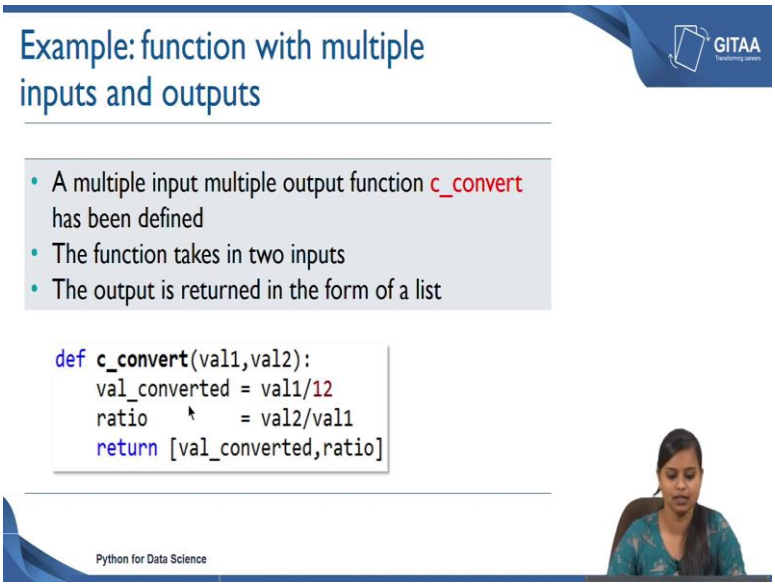
(Refer Slide Time: 22:11)



Let us see an example to see how function with multiple inputs and output works. So, here by defining function with multiple inputs and outputs I am going to do two things; one is converting age variable from months to years and another one is getting kilometers run per month. So, the converted values of kilometer will be stored in a new column called km_per_month we have already created one new variable for age as age converted.

So, I am just going to create one more for km_per_month that is also using the same insert function where I have set the position for the kilometer per month variable and the variable name being kilometer per month and I need to fill basically all the values with 0s initially so, I have given 0 here.

(Refer Slide Time: 23:01)

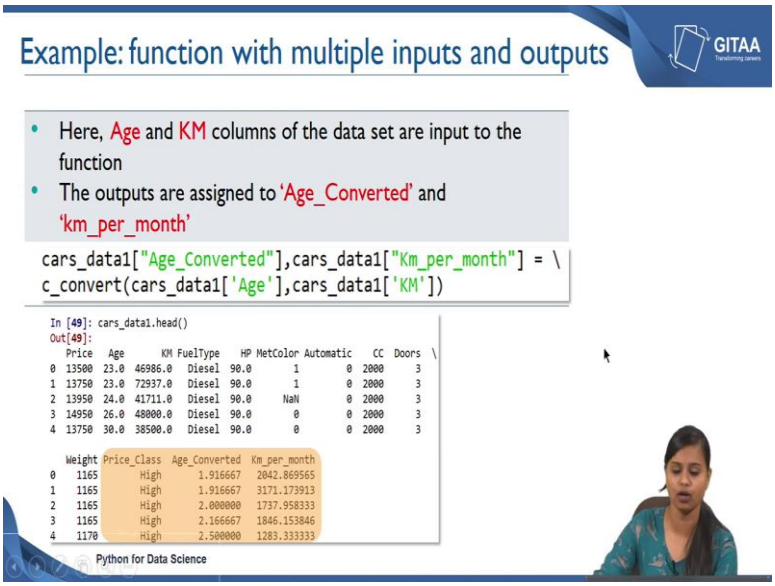


So, now let us define a function which accepts multiple arguments and which will also give us multiple results as a single object. So, here is the function definition. So, as I mentioned a multiple input multiple output function c_convert has been defined, this is the function c_convert and the function also takes in 2 inputs value 1 and value 2.

And, the output is going to be returned in the form of a list that is how I have defined a function, if you see from the start I have defined a function called c_convert I have given 2 inputs here value 1, value 2. And, I am going to get two output; one is value converted from for the age from months to years and the other one is the ratio. So, val1 and val1 divided by 12 basically divides all the observations under the age variable and give you an output and that will be stored in val_converted.

And, now I have created another variable called ratio where my interest is to convert all the kilometers run per month. So, in that case I am going to divide each and every observations of the kilometer by the value 1, where I am going to divide value 2 by value 1. And, it will return both value converted and ratio in a form of list, because I have given the values inside the square brackets so, that the value will be returned in a form of list.

(Refer Slide Time: 24:31)



So, let us see how do we do that? So, here age and kilometer columns of the data set are going to be the input to the function, because I am going to convert age from months to years, I am going to convert kilometers run and I am going to get the km_per_month. And, the outputs are going to be assigned to age converted and kilometer per month where we have created 2 new variables. And, here the outputs are going to be assigned to age converted and kilometer per month, because I am going to save my output simultaneously.

So, as you know in Python you can assign multiple values too by giving multiple variable names. So, that is what I am going to give in here. So, here the variable the first variable name is age converted from the dataframe cars_data1. And, the second variable is kilometer per month from the cars_data1. And, what I am going to save here is the output from the c_convert function, which I have defined it in the previous slide. So, c_convert is the function and the input would be cars_data of age.

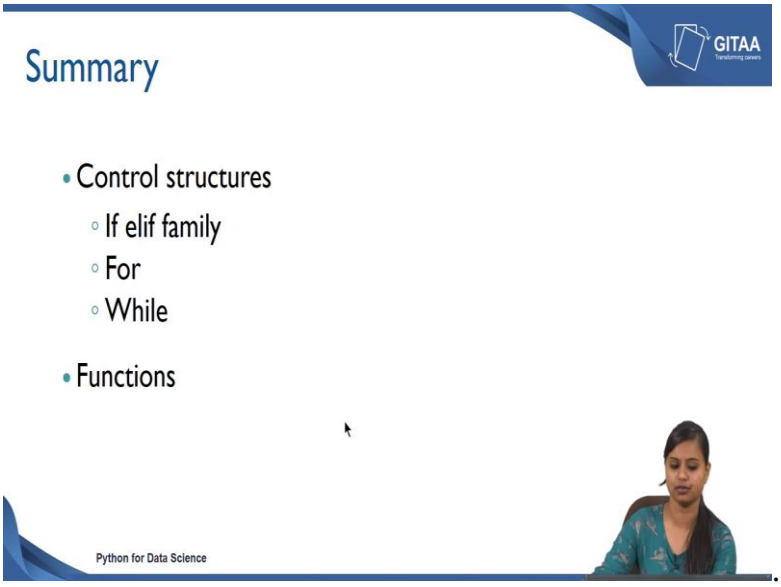
So, age will be the first input and the kilometer would be the second input. And, I have stored whatever value I am getting it using the first argument as age converted to age converted and whatever I will get it using the second variable will be stored into the second variable called kilometer per month. So, let us look at an output after using the c convert function what is the output that we are likely to get. I have used.head function to

basically look the first 5 rows of my dataframe just to see how my variables have been populated.

So, if you see the price class this is using the for loop and while loop which we have done, where we have bucketed all the price values. If, you look at here there are price values like 13500, we have given some conditions where, if it exceeds some value, then keep it as “High” “Low” and “Medium”, then in that case the first 5 rows have been stored as “High”. And, if you see we have converted the age which were in months to years I have just divided every row with 12. So, I got 1.916667.

But, if you see here this is not the rounded off value. So, this snippet just gives you the output with 5 decimal values, you can also round that to 1 decimal point, because this value does not make sense for you can also round off your values to 1 decimal points. So, the value will be 1.9 and you can also get the kilometer run per month. Here, you have the kilometer which is like 46986 and if you want to get it for month, then we have used the function and we have also got how many kilometers that the car has travelled per month.

(Refer Slide Time: 27:33)



So, now we have come to the end of lectures. So, let us summarize whatever we have done till now. We have seen about the control structures where we have covered if elif family followed by that we have seen for and while loops with examples. We have also seen about the functions where the function can accept multiple inputs and give you a

single output, we have also seen how a function can accept multiple inputs and give you multiple output as a single object.

Thank you.