**Functions**

[00:00:00.00] [MUSIC PLAYING]

[00:00:08.21] RYAN AHMED: Hello, everyone, and welcome to this lesson on functions. In Python, functions represent a block of code that performs a specific task. You can call a Python function, send it data, and receive results from it. Functions are super powerful, and they are an essential part of Python programming language.

[00:00:28.85] Functions offer many advantages. They allow for creating a reusable code that can be called multiple times, and executed when needed. Therefore, improves code efficiency.

[00:00:42.07] Functions break down long code scripts into smaller ones for better code structure, readability, management, and debugging. Here are the key learning objectives of this lesson-- understand the syntax of writing functions in person, learn how to set default function arguments, call functions, send them arguments, and receive data from them, develop a simple function that calculates the future value of money, given the present value, annual interest rate, number of years, and the number of compounding periods.

[00:01:17.72] So let's head over to our Jupyter notebook and get started.

[00:01:28.14] All right, so right now, we are in the Jupyter notebook titled Functions. So, in Python, functions represent a block of code that performs a specific task. You can simply define a function, you can call the function, you can send it arguments, send it data, and then you can receive data from that function.

[00:01:51.25] So let me show you an example. Let's assume that I would like to define a Python function that will simply add two values together. So I'm going to send the function two numbers. The function is going to return the sum of those two numbers. Pretty simple, pretty straightforward.

[00:02:09.63] Here is the syntax in a Python function. Simply you say, def-- and this is the defined keyword. You give the function a name. So here, I'm going to call it my\_sum. You open parentheses, and then you specify the function arguments. This is simply the values that the function is going to receive when called.

[00:02:35.34] So here, I'm going to say, this function-- which is my sum function, is going to receive x, and is going to receive y. Please don't forget the colon here at the end, indicating the end of the def line, for the definition of define line.

[00:02:51.36] Next, I'm going to add an indentation here, or whitespace, to indicate the body of the Python function. Next, you specify what you want to do with the x and y. So here, simply, I want it to sum up x plus y. So I'm going to say z is equals to x plus y.

[00:03:10.65] And then, I'm going to say, return z. And these two lines of code are going to be the body of the function. And simply, when I call this function, I'm going to send it x and y values-- can send it maybe, let's say, five and seven, for example. The function is going to return back 12-- so it can return the sum of these two values.

[00:03:34.88] OK, all right. The first point that I would like to note here is when you define a function-- so when you just write these three lines of code in here-- you will notice something quite interesting. You will notice that, well, nothing happens. So when I run the code, which is going to come up next, you will notice that you just defined the function, but you haven't called the function yet. You haven't invoked the function. And that will happen in a different line code, and that's what I'm doing here.

[00:04:07.21] So this is the function definition, these three lines of code here-- one, two, and three. This here, this is the function call. When you call the function, you call the function with its name. So the function name is called my\_sum-- that's why here I'm calling the function with its name, which is my\_sum. And of course, you don't need to write def anymore, because you're not defining the function, you are calling a predefined function, already-- you defined it here already. Here, you're just invoking that function, you're calling the function.

[00:04:43.62] And then, you need to send it arguments. You open parentheses, and then you say one, please sum up five and seven.

[00:04:50.99] So simply what's going to happen behind the scenes is an assignment operation. If you recall, we covered assignment operations in the past. Simply what's going to happen is, the value of five is going to be assigned to x-- it's going to be sent to x. So x is going to be equals to five.

[00:05:08.69] The value of seven is going to be assigned to y. So y is going to be equals to seven. So what's going to happen here, when that function is called is simply, I'm going to sum up x plus y, or five plus seven-- and z is going to be equal to this one. And then I'm going to return 12.

[00:05:26.72] So simply what's going to happen here is when I call this function, this function call is going to be replaced with whatever value I got back from the function. And that is going to be the output 12.

[00:05:39.50] Let me show you how we are going to execute that in code. So what I'm going to do here is, I'm going to zoom in a little bit. And here, first, I'm going to define my function. And here, this function, again, I'm going to call it my sum. So I'm going to say def, space, I'm going to give the function a name, which is my sum. I'm going to open parentheses, and then I'm going to define x and y-- these are the arguments of the function.

[00:06:07.10] I'm going to add colon at the end, I'm going to add the indentation or whitespace, and then I'm going to specify the body of the function. The body of the function is z equals to x plus y, and then I'm going to say to turn z. If you press Shift and Enter, well, nothing happened-- which makes sense because what happened here is we only defined the function, but we haven't invoked it, yet, we haven't called the function, yet. We haven't sent it any values.

[00:06:33.80] Like the function doesn't know, OK, I know that now I can sum up x plus y. But could you please tell me what are the values of x and y? Like it's going to be five, 10, 20. And that's what's going to happen in the next line of code.

[00:06:46.94] So what I'm going to do here I'm going to call this function. So I'm going to say, my sum, I'm going to open parentheses, five and seven, and this is going to be the function call. So if you press Shift and Enter, here you go. You will simply get 12.

[00:07:00.55] And behind the scenes, five has been assigned to x, seven has been assigned to y. I'm going to sum up five plus seven, end up with 12. And then going to return z, or return 12, and this is simply what I got here after I call the function.

[00:07:17.26] OK, so this is what I covered so far is the function syntax, how you define a function, and how you invoke or call that function. Let me show you an example.

[00:07:30.53] And before I cover the example, I just wanted to do a quick review of the future value formula. And let me zoom out a little bit. So the future of that value formula describe the relationship between the present and future values of money for n number of years, and m compounding periods per year. For example, the equation here is, this is a future value, after n number of years. It's going to be equals to the present value at time zero-- that's the money right now-- one plus i, were i is the annual interest rate, divided by m, where m is the number of compounding periods per year. So if you assume that there are no compounding happening within the year, m is going to be assigned to one.

[00:08:24.11] And then m times n, this is simply is going to be the total number of compounding periods. So what I wanted to do right now, is I wanted to simply define a function that is going to calculate the future value, given the present value, interest rate, m-- so I'm going to send it the number of years, send it the annual interest rate, send it how many times I'm compounding per year, and also, the function is simply going to return back the future value.

[00:09:00.60] So let me show you how we can write that in code. Let me zoom in again. So what I'm going to write here is I'm going to define a function-- I'm going to call it calculate\_fv, or calculate future value. So def calculate future. Then, the function is going to receive present value, interest, number of years, n, and m, which is a number of compounding periods per year. And the function is going to simply return PV times one plus i over m, to the power of m times n.

[00:09:32.00] Please note that two asterisks here, or two stars, indicating to the power of. And this is just the formula that I got here.

[00:09:38.27] OK, so if you run this function, what you notice is, well, nothing happens, which makes sense, because we only defined the function, but we haven't called or invoked that function, yet. So let's go ahead and test it out. So what I'm going to do here is I'm going to get from the user all the values that I need to send to the function.

[00:09:58.41] So here, I'm going to get the present value. And get the annual interest rate, number of years, and the number of compounding periods within a year. So if you press Shift and Enter, enter the present value. And here, I gave you a couple of numbers here so you can test it out.

[00:10:13.38] So for example, today, I have $100,000. You press Enter. Enter the annual interest rate-- and I'm assuming that the interest rate is going to be, let's say, 10%-- so 0.1. Press Enter. Enter the number of years. And I'm assuming I want it to see what's going to happen, how that money is going to grow after five years.

[00:10:34.46] And the number of compounding periods within a year-- here, I'm assuming that the interest is compounded annually. So I'm not doing any semiannual compounding, or any monthly compounding. That's why m is going to be equal to one.

[00:10:48.95] If you press Enter, now I got all the values from the user. Next, I'm going to call the function. And this is simply that line of code is the function call. You call the function with its name.

[00:11:02.97] So if you recall, this was my function here. I call it, calculate underscore future value. And if you go down here, this is the function call. And you simply need to call the function and send it arguments, you want to send it values.

[00:11:17.94] So I'm going to send it the present value. And the present value is going to be assigned here to PV variable within the [AUDIO OUT]. I'm going to send it interest-- and interest is going to be assigned to i. Please note that order matters, so here, the order of the function call has to match the actual definition of the function.

[00:11:39.46] I'm going to send it num years-- so num year is going to be assigned to n. And the compounding period's going to be assigned to m. So once you do that, I'm going to get the future value. And here, this is just a simple print operation, I'm going to say the future value of x, after x number of years, and x compounding periods per year, at x interest rate is equal to this value.

[00:12:03.55] And I'm just going to simply print out the present value, num year, compounding periods, interest, and the future value, as well. So right now, if you press Shift Enter, here we go. What you notice is simply the future value of $100,000 after five years, at one compounding periods per year. So annual compounding-- or the interest is compounded $annually at 10% interest rate, is going to be $161,000.

[00:12:31.00] And I also included a link for you here if you would like to do a sanity check. And I actually have that link open. All you need to do is to write down the present value, which is 100,000. A number of periods, I said five years. Interest rate is 10%. And m, which is my compounding times per year.

[00:12:50.33] And if you just click Calculate, you will simply get $161,051. And if you go back, you will simply see that value that we got here. OK, so it looked like it worked.

[00:13:02.11] Finally before we conclude this lesson, I just wanted to show you the default parameter that we can set within the function call. So when I define a function, what I could do, as well, is I have the option to set a default value. For example, here, this is my calculate FV, it's exact same function that we used before. The only difference is I added m equals to one-- this is just the default parameter.

[00:13:28.85] So what I could do is, if I called the function, and I did not set the value for the compounding periods per year-- which is the m parameter-- well, the default value here will prevail. So I can simply kind of use the default value if I do not set a value when I call the function.

[00:13:50.03] For example, if you run the set, what I did here is I set m equals to one. Simply I'm saying, if I do not specify the number of compounding periods per year, let's assume any one compounding. That's pretty much it.

[00:14:04.82] And that's what I'm going to do here, is I'm going to call the function. I'm going to say calculate FV. I'm going to send it present value, the interest, is going to be to i. Number of years, going to be assigned to n. And I did not specify the compounding periods. If I do that, you're not going to get an error message, you are simply going to get the answer-- which the same answer that we got before, when we set m equals to one.

[00:14:30.51] This was the exact same answer that I got here. When I specifically specified the compounding periods. And if you go ahead and you change that venue here and add maybe let's say I wanted to compound it, let's say, monthly. So if I add another parameter to one, 12 is going to be assigned to m. And the default value is going to be ignored-- so I'm going to ignore the default value.

[00:14:55.38] If you run it right now, Shift Enter, here you go. You will simply get $164,000, and that's a little bit more of course compared to the 161 that we had before. And it can do the confirmation here, if you say 12, for example, and you run it, you will simply get $164,530, and that's the exact same number that we got in here.

[00:15:16.86] OK, all right, so that's it. That's all I have for this lesson. I hope you enjoyed it. In the next lesson, we're going to have our practice opportunity. Please stay tuned. Best of luck, and I'll see you in the next lesson.

[00:15:28.05] [MUSIC PLAYING]