Transcript for Session 039

Listen to the podcast session, see resources & links:

http://chandoo.org/session39/

Transcript:

Hi and welcome to http://chandoo.org podcast. This is session number 39. http://chandoo.org podcast is designed to make you awesome in data analysis, charting, dashboards and VBA using Microsoft Excel.

Today, I have a very exciting and relevant topic for all of us who are using Excel to do automation, i.e. Visual Basic for Applications (VBA) programming or writing macros or using some code to do some repetitive tasks. We will be talking about how to use For loops, how to write them, when you would use them, some use cases and scenarios and some tips and techniques. But, before we jump into the For loop stuff, I just want to share a couple of announcements with you.

The very first announcement is thank you. I really want to thank you. When I say you, I don't just mean people who listen to http://chandoo.org podcast but everybody in general who visits http://chandoo.org and reads something or watches a video or listens to a podcast. I just want to thank you. I recently closed my online training program on 50 Ways to Analyze Data. I opened it for enrolment on 13th July and I closed it on 20th July. We had such massive success in that. We had more than 200 people joining this program for the second batch and I am now looking forward to engaging with all these wonderful people and teaching them 50 awesome ways to analyze data. The next time that we will re-open this program will be in January 2016 but I couldn't wait to get started with this batch and help them really become awesome Analysts. So, thank you so much for supporting http://chandoo.org and if you are one of those 200 people who enrolled for this program, my heartfelt thanks to you for choosing http://chandoo.org as a partner in your learning and your journey to awesomeness.

The second announcement that I just want to share with you is that on last Saturday (18th July) - many people don't listen to the podcasts immediately and I know that - so, if you are listening a little further in the future, this might not sound so exciting to you. But, on 18th July, I went on a bicycling ride to a hill station - about 1000 metre tall range of hills called Eastern Ghats - near our house. I went there for a cycling ride. It was about a 120 km ride up and down. I left on Saturday morning at 5 am along with a couple of friends and we all rode to the top of the mountain pass. We had a lunch break there and we returned back home by evening. This is actually very exciting for me because I did a similar ride in January this year but around mid-February I injured my knee, and I couldn't do any cycling for more than

three months. After that, even though I was doing a bit of cycling, most of it was for about 20 or 30 kms and so nothing really big. So, I was sceptical whether my knee has recovered fully or not but I could feel in my body that I am feeling a lot better and my knees are good. So, I took up that ride as a challenge to see how far I can go and I was always thinking throughout the ride that I would give up any time if there is a recurring pain in the knee and just take a ride back home but, fortunately, it didn't happen and I had good fun riding with friends and enjoying fresh mountain air and lots of fantastic views from those mountains. It is a strenuous climb but it was worth doing it and I felt good about it. So, I just wanted to share that with you because I feel like sharing these things with the podcast audience.

The third announcement that I want to share with you and which is more relevant for the podcast session that we are doing today is that I have prepared a special bonus Excel workbook that goes along with this podcast. It is like a companion workbook for this podcast because we are talking about For loops which involves a bit of programming and writing macros. Visualizing everything and learning more about For loops can be hard, and so I have prepared a special bonus file for you but you'll have to do something extra to get it. You are listening to the podcast without any extra effort but in order to get that file you must do something extra. What do you need to do? It is very simple. After listening to this podcast, go to iTunes and drop your genuine review about http://chandoo.org podcast there. That's all that you've got to do. Just write a review and then send me an email saying that you have written a review there and I will provide a link to download the workbook for you. You can write any review - whether you like the podcast or not, the things that you enjoy about the podcast, where it can be improved - you can share whatever you feel genuinely about the podcast on iTunes. This will help me in 2 ways. One is that it will introduce our podcast to new listeners because iTunes looks for people who are getting reviews and then they try to feature those podcasts for greater audience. So, I can make more people awesome in Excel and I can give something back to you as well. Apart from the podcast, I get to share this workbook with you. Don't do it now; don't pause the podcast. Finish the podcast, listen to all the topics and when time permits and you have a few minutes, just go and drop this review on your iTunes. Thank you so much for listening to the podcast once again.

Now let's get started with the For loop concept. I am not sure how many of you are familiar with VBA programming. If you are new to VBA and if you have never done VBA programming or written a single macro in your life, I would encourage you to listen to the 22nd episode of the http://chandoo.org podcast. If you are listening to this podcast on your iTunes podcast then go back to the archives and get the 22nd episode - CP022. It is an episode called 'What is a Macro - Introduction to VBA'. Go and listen to that podcast for a really good and simple introduction to VBA. Alternatively, you may also go to http://chandoo.org and complete a simple free online crash course on VBA. It is a 5-part course that will introduce VBA to you. I am saying all this because only if you have some knowledge about VBA can you understand what a For loop is. So, I am not really trying to pitch the idea of For loops to a complete newbie in terms of VBA.

Assuming you have some knowledge of VBA and you have probably used a For loop sometime in your life or maybe you have seen or at least heard about it, let's start talking about it. What is a loop? Let's first talk about the concept of a loop. Well, loop is a technical word and it means something but we don't have to go into the technical definition. We will go into the physical real life situational definition of what a loop is. Many times when we are doing some work, we tend to repeat a set of steps or instructions or a separate process a couple of times. For example, let us say that you are making a peanut butter sandwich for breakfast for your family. Let's say that your family is like my family and so you have 4 members - you, your spouse and two kids - and everybody needs to eat one peanut butter sandwich. So, that means that you would have to prepare this peanut butter sandwich - take two slices of bread, apply peanut butter on one, apply jelly on the other and then slap them together and call it a peanut butter sandwich and do this four times. This is nothing but the process of creating four peanut butter sandwiches. Now, the process would not alter between one sandwich or another unless you have somebody very picky like my daughter in your house in which case you might have to cut the edges of the bread or toast it differently. But, assuming that everybody likes their peanut butter sandwiches in the same way, the process is exactly the same. If you know the steps to create the first peanut butter sandwich, you would follow those same steps to create the rest of the three peanut butter sandwiches. This is nothing but a loop. You are doing a set of steps in a loop four times to create four separate sandwiches. So, here, when you are preparing peanut butter sandwiches in your kitchen, essentially, you are writing a loop of creating the peanut butter sandwich. Of course, you are also yourself executing it, and you are making four sandwiches. So, your loop is running four times. This is what a loop is in reality.

You can now relate this to any of the work that you do. For example, in your work, if you are preparing a certain type of report for the regional directors and there are seven regional directors but the report contents are vastly similar and the only thing is that the input data for those reports would change and each region would get their own numbers but the layout and charts and formatting is similar. So, essentially, you are producing seven reports. The process for producing them is the same but you would be doing those same steps seven times, each time producing a different report. So, you are writing or running a loop seven times producing seven different outcomes but the process is the same. This is nothing but a loop.

Now, let's talk about the programmatic or technical definition of a loop**. A loop is nothing but a set of instructions or a separate process repeating certain number of times.** We could refine this definition. You could refer to any kind of programming or VBA book where you might be able to find a more formal definition of a loop but it doesn't really matter. So, if that is what a loop is, then what is a For loop? Well, there are many kinds of loops that can be done. For example, if you are feeding somebody like Hulk who chooses to visit your home and Hulk likes to eat peanut butter sandwiches. Hulk doesn't eat one or two sandwiches. He might be very hungry. He might have finished a long run all the way from South America to home in which case you would have to prepare as many sandwiches as Hulk would eat. So, there is no certain number. It is not 4 or 6. You would have to feed Hulk until he is satisfied in which case you are running a loop of preparing those sandwiches until Hulk's stomach is full. You could prepare 100 sandwiches or if you are a lousy maker of peanut butter sandwiches and Hulk doesn't like them then you

might just stop after 1. Well, jokes aside, this kind of a loop where you are doing something until a condition is satisfied is called Do While or Do Until loop, i.e. do while a certain condition is satisfied or do until a certain condition is met. We are not going to talk about those loops. They are also there and you can understand them. Technically, they are all similar and so we will talk about a more common structure which is a For loop.

A **For loop** is where you are going to do certain type of steps or a sequence of actions for n number of times. 'n' is a variable and so it could be 4 or it could be 75 or 215 or whatever you fancy. Now that we have kind of understood the meaning of loop and the context of loop and where you would use it, let us go ahead and understand some of the standard For loops that you would use. In VBA there are two types of For loops that we can write - the first one is called For Next and the second is called For Each. We will talk about each of them individually.

Let's talk about the **For Next** loop. The For Next loop is a very simple structure. If you want to repeat a set of steps or instructions or a set of VBA code statements n number of times, you would write For Next loops. You would say 'For sandwich = 1 to 4' and then you'd write the instructions like take the bread, apply peanut butter, take another slice, apply jelly, join both slices and then the last step would be 'Next sandwich'. This is nothing but your For-Next loop.

Now that we have talked enough about this peanut butter sandwich example, let us talk about something more realistic. Let's say that you are looking at some sort of data. You are looking at some customer address data that is imported from a CRM system. Usually, when you export to Excel, sometimes the data is what you want and sometimes it is not. So, let me share an example where the exported data is not meeting our criteria. For example, the customer address information is there but, instead of name, house number, locality, city and zip code in five different columns, each of them are appearing in 5 separate rows. So, if I have 20 customers, I would usually expect this export to look like 20 rows and 5 columns but, instead, we have 100 rows and 1 column. The first row is the name, the second row is the house number, the third row is locality, the fourth row is city and the fifth row is zip code and then it is a repeat. Given all this, you are trying to do some analysis on the kind of distribution we have by zip code. All you care about is the customer zip code which is the fifth item in the list. So, you just want to extract every fifth item and do something with it. In this kind of a case you can use a For loop.

You know that there are let's say 1000 items in the data. Let's use a variable like i or z or m. Let's say that we are going to use the variable i. So, we would write something like:
For i = 1 to 1000

And, then, you could write the logic inside which would be something like:
If we are looking at the 5th row Do this Else do that

So, in our case, inside the For loop, we are focusing on getting the 5th row only and ignoring the first 4 rows and extracting the 5th row item and putting it in a separate place. So, from the original data of all the address details, we could write a For loop that would extract every 5th element and place it in a separate range. For your information, you can access a bonus Excel workbook that contains this example VBA code and you can download and play with it. But, in order to obtain it, you must drop a review on iTunes about http://chandoo.org podcast and drop me an email about your review so that I can give you this file.

Alternatively, if you have been playing with VBA for some time, you may be able to figure out the logic yourself. So, this is a practical scenario where we have a lot of data - 1000 times - and we want to do something to it. We want to treat every fifth item differently and place it somewhere else. This is where a For-Next loop would nicely fit in.

Another example is that if you are producing something like a regional report and you have 7 regions, each region must use a separate dataset and produce a similar report and save it as a PDF. You could write a For loop that could do this for you. You can define the entire process and then write a For loop outside it.
For region = 1 to 7

Do all these steps - load the data, run the calculations, generate the report, take the print area and export it as a PDF with a given name like Region1.pdf or whatever and then continue with the next region.

So, once you finish running this loop 7 times, you would have all the 7 reports generated for you. This is really the power and beauty of For loops. Once you can identify a set of instructions in your long business process that are repeating, you could narrow them down and put a For loop around them so that the For loop can do this work for you. Your macros will be shorter, they will be efficient and they will be error free because anytime that you are repeating a bunch of steps, if you do it manually maybe out of boredom or frustration or lack of attention or any other reason, you might make a mistake and then your results won't be consistent. It is the same with the peanut butter sandwich case. If you are feeding Hulk there is a very high chance that your 1st sandwich and your 75th sandwich will not taste the same. This is because maybe by the time you have prepared 74 sandwiches you might be frustrated and be thinking why this guy is still hungry and you might make a sloppy sandwich. You get the point, right? This is why when we do a manual process where there are a bunch of steps repeating using VBA to automate them could be great. Of course VBA can't make peanut butter sandwiches which could be excellent but unfortunately VBA is not there yet. Maybe in Excel 2019 they will add a function for that!

Jokes aside, that's your For-Next loop. Now, let's talk about another loop called **For-Each** loop. This is where it is a special case of For loop where we need to do a certain set of actions for every item in a

collection. This is where you have probably heard a lot of jargon words and so let me talk a little bit more in day to day English. Let us say that you are formatting some charts. In your report there are lots of charts and all the charts must be consistently formatted. So, every chart must have a blue color border around it. The gridlines, labels and the axis text must be in a dull gray color. The title should be 18 points big and the series of bars or column should be in blue color. This is your company standard formatting guideline but you've created the charts and they are all over the place with different colors and different schemes and you thought that you could automate the process. But, how many times would you do it? Let's say that the report now has 7 charts and you thought that you would write a For loop for 1 to 7 but tomorrow you might want to apply this formatting for another report where there are 19 charts in which case you will need to go and edit the code again. So, to avoid such kind of macro changes from time to time or situation to situation, Microsoft introduced this For-Each loop construct.

The way this works is that it will take a look at all the charts in your worksheet and for each chart in that worksheet or workbook or in that place, it will do those actions. So, in this set up, for each chart, it will do a certain set of actions which is nothing but 'for each item in a collection'. Here, item refers to the chart and the collection refers to the charts collection. The other places where we could use a similar thing is that let's say that you want to format every worksheet in your workbook in such a way. You might want to format them or set up print settings for them and there are n number of worksheets. You don't have to know how many there are; you could simply say:
For Each worksheet in my workbook do this

So, For-Each is a special type of For loop where we don't have to specify the end point. We will simply have to say 'for each item in the collection'. Many objects or items in VBA offer a collection. So, a workbook is a collection of worksheets. Charts is a collection of all the charts. Pivot tables is a collection of all the pivot tables and so on and so forth. So, we would refer to the right kind of collection and then we would access everything within that collection and do a certain set of actions on them. So, For-Each is a special type of For loop. Again, we could use this is many real life business situation. For example, going back to peanut butter sandwich example, let's say that family is a collection and so, instead of writing 1 to 4, we would simply say:
For Each member in family prepare a peanut butter sandwich

So, this For loop will make 4 sandwiches for my family but it would make 6 sandwiches for your family if you have more people or a couple of pets in your house or whatever. So, that's the For Each loop. Both For Next and For Each are really popular and powerful ways to take a set of actions and automate them. I highly encourage you to play with these things and see how often you can use them in your work because they simplify your VBA code and they will introduce a lot of potential to your automation projects.

Now let's talk briefly about a special thing called **nesting of loops**. This means that sometimes we will have to write one For loop inside another For loop. That concept is called nesting. You could write one loop inside another of you could write one loop inside another and inside another and inside another. This is like deep nesting of For loops. When would you use them? Why would anybody do it like that?

For example, let us say that you want to apply consistent chart formatting to a bunch of reports that are in a folder. So, the folder has several files - let's say 10 files. So, you would write the first For loop for file =1 to 10, open the file and within the file itself there are several worksheets. So, inside that For loop, we are writing a For-Each loop, i.e. for each worksheet in that workbook. Within each worksheet, we would have a bunch of charts. For each chart within worksheet.charts or whatever we will apply the formatting and then go to the next worksheet to do it. Then, once we are done with all the worksheets, we will save it and close the file and then continue with the next file. This is nothing but 3 levels of nesting; the top level talks to individual files. The next level talks to individual worksheets. The deepest level talks to individual charts and does the formatting tasks. This way, you can structure your code elegantly and everything works beautifully provided you have done the right type of coding and there is no chance for errors. It will work in a very powerful and flawless manner. That is where nesting is a very powerful concept and you can use that to do a lot of things by combining one For loop with another.

I will talk a little bit more about the performance issues with For loops and what happens when you have deep nesting towards the end of this podcast. For now, let's move on and talk a little bit about special tips when you are using For loops. By default, For loops (especially For-Next loops) will take numbers. You can't really say For Next for text values or boolean values or things like that. You are essentially talking about whole numbers and so if you write something like For i = 1 to 100, it is going to run 100 times by default. So, going back to the case of For-Next loop that we wrote earlier in the episode where we have the addresses of a lot of people and we only wanted to extract the zip code which is the fifth item, essentially, we don't need to go through all the 5 items. If we could directly go to 5th, 10th, 15th, 20th items then that would be good. In such cases, you could use a special operator called **Step** and tell the For loop how much stepping needs to be done. The default step is 1 but you can tell the For loop to step 5 at a time. So, you could say:
For i = 0 to 1000 Step 5

This will start with 0 and then step to the 5th item, 10th item, 15th item and 20th item. It will kind of skip away all the middle items for you. The step parameter is an optional item in the For loop syntax and when you use it you can control the stepping or the increment value. You can also use it to do a negative looping. The default structure for For loop is:
For i = 1 to 10

So, it will go from 1, 2, 3, 4, 5 and so on. But, if you want, you could do it from 10 to 1 as well. It may be useful in some special scenarios where you need to go from bottom up or whatever. In such cases, we would write:
For i = 10 to 1 Step -1

This will help you to achieve that kind of construct. So, Step is a tip for you especially in some tricky situations where you need to do something special for every fifth row or every fifth item or something special for every odd row or even row or whatever.

The other tip when you are working with For loops is that sometimes in between the loop you might want to exit the For loop completely. This is because if a certain condition is met you don't want to do the rest of the process anymore and you just want to quite it. In such cases, you could use **Exit For**. Keep in mind that if you write Exit For directly inside the For loop it is just not going to run properly the way you would expect. So, ideally, you want to put Exit For inside an If condition. So, you will write something like:
For i = 1 to 10 do something

Then you will write that if an If condition is true then Exit For. This means that if that 'something' condition is met during the For loop then the For loop will stop there. So, if it is running 1 to 10 times and the condition is met the 7th time, it will exit there itself. Whereas, in some other scenarios, if the condition is not met, it will continue running all the way up to 10 times and then it will close the For loop. So, Exit For is a powerful way to stop the loop and bring back the attention to outside the loop and continue running the remaining macro there.

You might be thinking when you would use the Exit For condition. You could understand when you would use Step but what about Exit For. Well, there are many scenarios where I would use Exit For. Talking about the chart formatting macro, let's say that the chart formatting rules are blue color bars or columns or lines. But, if it is a pie chart then there should be no blue color. Leave it to the default format. So, that means that inside your formatting macro VBA For loop, you would check what the chart type is. If the chart type is a pie chart then Exit For because there is no need to do the formatting for that chart and continue with the rest of the worksheet charts or whatever. So, you could use Exit For in a scenario like this. Alternatively, in the regional reports that you are producing, in that example for a certain region you don't need to do all the steps. You could stop it mid-way because it is a smaller region and so you don't need to add the detailed section for the report or whatever. In such a case you could write a special condition that 'If the region is this then Exit For' because we don't want to do the rest of the steps.

So, Exit For is a special type of instruction that tells the For loop that it has done enough and to stop the For loop for now. So, you can use that depending on the business situation or the looping scenario that is going on. Those are some special tips/steps for telling the For loop how to step through the counter values and Exit For if you want to exit the For loop in between.

Just to do a quick recap, we talked a little bit about what a For loop is, what a loop is in reality and how to define it technically and then we talked about For-Next and For-Each loops and then we talked a little bit about nesting For loops one inside another and we talked about the Step and Exit For scenarios.

Now that you are familiar with For loops, let us talk a little bit about **performance issues**. Programmatically, For loops are fairly efficient. I said fairly and not highly because anytime that you are

repeating a bunch of steps, it is going to take some amount of resources from your computer. Your computer will spend some time, memory and processing power to do those same steps repetitively. But the beauty of computers is that computers are built for doing repetitive tasks. So, anytime that you ask your computer to do the same steps 5 times, 200 times or 7 million times, your computer can get really fast and efficient. The only problem is if you are doing something that is highly expensive, i.e. the process inside the For loop itself is a highly expensive process then if the For loop is running 10 times, that expensive process is now going to run 10 times whereas if the For loop is 7000 times then you are multiplying the expensive process time and resources by 7000. This can quickly snowball if you are using a lot of Nested loops. With a single loop you could be fairly certain that almost any process that can be manually done will happen a lot faster with a For loop but when you have nesting and it is happening 6 million times on a normal PC or laptop, it can slow down things drastically or it might even crash Excel. So, you need to watch out for a lot of deep nesting and try to avoid it if possible as well as see if there is a way to get rid of some of the loops and improve the coding. So, examine if there is a faster way of doing things. For example, many people do things like selecting individual cells or charts before formatting or changing or modifying or doing something to them inside For loops. Now, it is not necessary to select an item to do certain things. So, that instruction 'Range("A1").Select' is unnecessary in order to process the A1 values or do something to them. So, **avoid steps inside the For loop that are unnecessary** because anything that is unnecessary will get multiplied that many times and can quickly snowball and have more impact on your overall processing time.

So, when you are writing For loops make sure that the inside meat portion of the For loop, i.e. the set of instructions that will repeat, is highly efficient and you have **optimized that code** as much as possible because any kind of shaving that you can do there will have performance impact of 1000 or 100 multiples because you are going to run that instruction those many times in the For loop. So keep that in mind.

One watch-out or gotcha when it comes to For loops is that although it is very difficult to write an **infinite For loop**, i.e. a loop that runs infinite number of times, it is still technically possible. If we write something like For i = 1 to 10 and you are doing some instructions inside the For loop and for some weird reason or for a typo or whatever, you wrote something like i = 1 inside the loop, the For loop will finish all the instructions and then it sees i =1 and so it will reset the i value to 1 every time. This way i stays on 1 forever and you end up writing an infinite loop which can be very cumbersome and it can choke your computer. So, make sure that any kind of code that you are writing inside the For loop is not directly talking to the For loop variable itself, i.e. the counter variable. So, if you are using i= 1 to 10, you should not use i as a variable inside the For loop unless you know what you are doing with it. Of course, once you get used to programming and once you feel comfortable with writing more and more code, you might want to deliberately manipulate i values inside the For loop for some reason. But, for most reasons, try to avoid it so that it doesn't accidentally create an infinite loop and frustrate you. What happens is that when Excel gets into an infinite loop, it pretty much hangs in most cases. Nothing happens - Excel runs this macro for some time and then everything freezes and you get a warning message saying that Excel couldn't run this macro or whatever and when you click OK, Excel just closes.

So, you might even lose your code and all that. So, don't try to manipulate. That's what I am trying to say.

Two things to watch out for are nested If's and nested For loops where the code needs to be really optimum so that you can have a good performance boost and watch out for accidental infinite loops. These are the things that I have to share with you about For loops. As I said earlier on in the podcast title, may the For loop be with you!

Now, let's talk about the **special bonus**. As I said, For loop is a technical concept and so it is difficult for you to visualize what this is like and try to learn all of this just by listening to this podcast. So, I have prepared a companion Excel macro workbook that kind of teaches For loops with a few more examples and if you would like to have it, just drop us a review for this podcast. Just go to iTunes and leave a review there. If you are not listening to this on iTunes or your Apple devices, you can still write a review - if you have an Apple account, you can log in and leave a review there. Alternatively, if you are listening to this podcast on any other application on your phone, you can write a review for the podcast in that application itself so that more people can discover http://chandoo.org podcast and become awesome like you. Thank you so much for listening to this podcast. I hope it helped you understand For loops and hopefully you will write better For loops next time. For all the show notes, information, links and resources about For loops please visit http://chandoo.org/session39. Thank you so much. I'll talk to you again in the next episode.