RNB ZR CS 198 8:30 -9:50 a.m. ET
September 23, 2024

>> INSTRUCTOR: Good morning. We would like to complete the tale of lecture three and then moved to lecture four. Just to make sure -- maybe, maybe no. A good way to think about dictionaries . A good example is think of memes and grades . So the important thing about dictionaries is they're (Indiscernible). That's the important thing. Every pair has a key . Quite different to the subjects here. A good example is think of names of you guys and the grades in the class so each of these is one person. Can you give me another example of a dictionary?

>> STUDENT: Name and age. Give me another --

>> INSTRUCTOR: Give me another example of a dictionary.

>> STUDENT: (Indiscernible) And location. Student --

>> INSTRUCTOR: (Indiscernible) And location. Give me another example of a dictionary.

>> STUDENT: (Away From Mic).

>> INSTRUCTOR: Name and phone number. Give me something different.

>> STUDENT: (Away From Mic).

>> INSTRUCTOR: Class and professor . Give me another example. Quantities and? (Indiscernible) Give me another example. Email and NetID. Why not? So almost anything that you can think of can be thought of as a dictionary. Now one dictionary by itself is just one piece of information . Name, grade . There could be another dictionary which is name, age, name, email. When you have data, you really have a collection of dictionaries . Is that clear? Now what is the relationship between a collection of dictionaries and tuples? You want to think about how these formalisms help you to process data. What would be a relationship between dictionaries and tuples? Let's make an example. One dictionary, name and NetID. Another one, name and age. Another one name and mail. What in your mind, what can you do with these dictionaries to create tuples? Remember, a tuple is a collection, and I'm trying to put in your mind the idea that from different dictionaries, you can create tuples.

We have a dictionary name, grade. Another dictionary name, NetID. Another dictionary, name, male. What can you do in terms of processing from these dictionaries, and here we would like to have tuples? You identify in those dictionaries a common key . In this

particular example, what do you think is the common key across the 60 nares in this example? Name, right? So name is the common key. Now from the first R(Technical Difficulty) this tuples can be accessed from these dictionaries by accessing these key in getting information. Move across the same key here to get the NetID etc. This will be more clear. I would say that like he's -- say it loud.
>> STUDENT: (Away From Mic) Condensed version of multiple dictionaries .
>> INSTRUCTOR: That is when use of One use of dictionaries. You have dictionaries that are representing a collection of and you have to decide if that is useful to . Raise your hand if this is completely clear. I have a whole bunch of people that this is not clear. A collection of dictionaries, you can extract a collection of tuples. So dictionaries are a more atomic collection of information from which you can build a database of tuples. What about the other way around? Suppose that you have unified a whole bunch of this . You have in your file a collection of tuples. What is the other direction? You can view that collection of tuples . If you need to do some processing, you can do that as a collection of dictionaries. And in fact, what databases do is they take this and say I'm interested in the grades. So the processing of the file will extract the names and the grades and now you have here a dictionary.
Is This completely queer -- clear? The name of the game and data analysis. The name of the game in data analysis is to link these dictionaries. Give me an example of something that you can ask about this collection of dictionaries . Give me an example of a question. That's what we do. We ask queries from the collection of dictionaries or from the database of tuples. Give me an example of a way to link dictionaries. Give me an example of a question that you would like to ask about a collection of dictionaries or a collection of corresponding tuples. Give me an example of a question. Well, this one is example that says name, grade, name . What kind of question.
>> STUDENT: (Away From Mic) Okay. You could ask the net IDs of certain majors. There's that but think about the relationship between names and majors.
>> INSTRUCTOR: It's already here. I'm looking for something that is not here. This is already . Give may the name, you go here to this dictionary and you find the name. We are looking for questions that are not explicitly in the data.
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: For example, you would like to ask the relationship between grades and majors. That is not explicitly here and you like to find that out as a type of query. What kind of question would be interesting between grade and majors? What kind of question?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: What is the average of the grades for this particular major? That's the question. Queries are really questions you would like to ask about the relationships between different pieces of information that are not explicitly in the data. What about the stock market for example? What would be a type of dictionary in the stock market? What would be the key? The key will be a stock ID. What will be associated with it? What is the value of your stock on a particular day, right? So you have all this information and this is happening every single day. What kind of questions would you like to ask? It's going to give

you the data. You want to earn money. What kind of questions would you ask? Which one is the stock that is performing better in the last week? For example. Which stock is trending down or which stock is trending up . Is that clear? Almost any question you can think of can be phrased as having this type of data . How can I answer that question officially? When we look at SQL, we'll be looking at example of words like that.
I Wanted to make sure you see these things are not independent of each other . Sometimes you want this or tuples or sets. But the name of the game in the last two decades is this. And with this, you can put them together to ask many, many questions. And the first mini project that was described to you in the recitation section, and that will be announced this evening, the data that you have there is about one point something million what? Anybody? What is the mini project data? Did you go to the recitation? You should have an answer. One point or so million book titles . That's part of the data. The other part of the data is to have people who read the books. Maybe from one to ten (Indiscernible) so here you have two types of data. Book name and (Indiscernible). You could think of that as a dictionary, isn't it? Give me another dictionary that is related to books. Books and (Indiscernible). That's another dictionary. Books and authors . What's another type of dictionary? Books employee genre. We don't have that data in this particular exercise, but that could be another dictionary. For this particular thing besides books and authors and books and ratings, we also have the users that provide the ratings. Now we have three dictionaries. The question that you will be answering is give may the books that are graded what? -- That are rated best among this collection of people that provide the ratings. How are you going to quantify this business of books being rated the best? How? How are you going to quantify? How many ratings we have? About 2000, 10,000? So we have about 10,000 ratings. 1 million books, 10,000 ratings. How from that data are you going to quantify the books that are rated the best? One hint here. In general, when you have some kind of measurement about certain objects in data, in this case the measurement was numbers, I write this book 7, this 18, this 12. A key mathematical idea is lookout for the average. The average of what? Well, this book has been rated by 15 people . Each of them gives a number two this book see look up what is the average of those ratings. Is that clear? Now what is the quantification or what are the (Indiscernible)? Those That have to highest average . But what does it mean those that have the highest average . The second idea, and this is very commonly used today. You may not notice, but you are affected by this daily is the top K. The top K is an important idea . The K you can say I'm going to look out for the top 2% ratings. Or the top 10 ratings are the top three ratings . We usually refer to that as the top K. when you go to one of these sites that sell you stuff, they may say you would be interested in this one or this book because -- they don't tell you the because that maybe they say they have been collected, all the books that you bought in the past. And based on that, they are going to try to target to you certain other books because you may like them. Behind this, there is a process that is rating all these objects, in this case books, and is categorizing them that these are the books that are the top K.
The same thing applies -- give me another example instead of books where the same thinking applies besides books?

>> STUDENT: Restaurants.
>> INSTRUCTOR: Restaurants, same idea. Give me another example besides restaurants and books. Come on guys, you don't buy anything?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: Yeah, but give me something more concrete. Hats . Yeah. Almost any product that you can think of can be subject to this analysis of who bought them, how many times they bought them and based on that because they select a scale. They select the top K and those other things they pushed you so you'll buy them. All this belongs to recommendation systems. This (Indiscernible) recommendation systems started with very simple idea that they want to describe to you that this assignment is a primitive recommendation system for books. But the general template is basically the same. The difference between a good recommendation system for a company and another company is really how much intelligence they put into the process of deciding what are the best . It all depends on how big the company is how much data they have . The company that has a lot more data is going to be able to produce a better rating system and a better recommendation. There are companies now that are using AI and other ways to predict now what are the products the rating is going to go out. The same thing for the stock market. The stock market is very complicated, but recommendation systems and they use all this infrastructure behind the scenes in order to provide you a convincing argument that you should spend your money.
Dictionaries are essential. I hope I gave you the basic idea. They are very easy to create. For example here grades . When you put these curly brackets and here they specified every pair. This is one pair. This is another pair. This is another . Between quotes.[word?] Codon 80. This is saying this is one key pair in this dictionary on grades. This is the second key value pair. This is the third key value pair and then when you have that, you can just refer to that as grades. You can access how many keyvalue pairs etc. The important thing about this notion of keys is that now you can access the data by using the keys. It's a very simple idea, but it has proven to be quite powerful. Before we were speaking of indices and you have a list. We were saying you can take these names dot and then put some brackets there and put some Numbers 7 to access those seven elements. When you were doing that, somehow you have to have the assumption that these items in the list can be accessed by an index which is a number. When you are dealing with dictionaries, the nice thing is that you can refer to a particular item by using the key.
This Is very (Indiscernible) grades . This dictionary so now you can say grades, brackets, and here you give the name of the person that's accessing and that accesses the specific grade for that individual. In general if you have one of these structures you're talking about and when you put the dot here you specify a metal associated with that. This is very important. You can think of metal as a predefined function that the language gives you. For example here, this is saying you're dealing with the dictionary. Dot (Indiscernible) is a method that can be applied to the dictionary and this is saying you are going to pop from the dictionary that particular element with this particular value. This means you're going to extract for this particular individual what is the grade. Sorry I missed something. In this case you are

popping (Indiscernible). I was thinking something different. No we have given you examples of functions in general and these particular mechanisms for having here a method. Think of these as a function that can be applied to that type of object . This is one method for dictionaries that is very important and necessary for a particular dictionary want to get the element on that dictionary with this particular key value. US -- you specify here you want -- this is not required a.[word?] Value. This is optional.
Here they give you an example of you are organizing a party. You are defining a function. That function has certain parameters. Slices of pizza, soda etc. and then you have a particular function that is going to use this data to compute something about the cost of the pizza . I guess you should have read that by now, but if not, the same thing . There's a function to compute the cost of soda and then at the end you want to determine the cost of the pizza plus the cost of the soda and this is a function. But remember for a function to say DEF, you specify the cost and then you indicate what this code is supposed to give you as output. Then you can use this. Here you have Here you have 14567 and in general you can have plenty for one of them so you can have lots of arguments. It's important that the arguments match the data. If N is first and B a second and S is third, whenever you want to (Indiscernible) something it's important that you remember the order of the arguments. The way to avoid difficulties with that is you use a print function that is going to put the two together. These are examples of defining functions and you have to be careful about the arguments . Again the full function has many parameters, it's hard to remember the order. What you can do, you can use certain mechanisms to avoid making mistakes about the order. Here when you have this and what the heck are the examples? You do this to do stuff . You use the printing function between quotes here.
Inside You can actually write yourself PIAS which is the name of the parameter you have up there and you put this between these brackets . So now confusion is gone because this is going to print this and this is going to put here the value of this and the value of whatever so these are tricks that are important. The way to understand this is that what this does is it puts oldies parameters or functions as a tuple. So you have a function with 25 parameters. And you do this. The name and the value. The name and the value. The name and the value. What the system does, it takes that whole thing else one tuple. And then we print and that system prints the entire tuple together. One of the purposes of this mini project that you are facing now is to get used to managed dictionaries, getting something done (Indiscernible). Now we move into another topic. That's the one that gives you power to do logical computations in these languages. Until now, almost everything that we have mentioned has been reduced to statements. Assignments? There are other kinds of statements which are called conditional statements. This is really what gives you power to process something. This is different to the usual statement that we have assigned to this statement in which you have here a value and here you have some value and that gets assigned to that.
This Is an unconditional statement. Conditional statements are things that look like this. The important keyword here is if. Here there's some condition which is (Indiscernible) in the logic of this is the expression is true, something happens. If the expression is false,

something else happens. Happens to be that the way that these textbooks describe the semantics of this at some point becomes a little bit cumbersome. This is what you should (Indiscernible) anytime you have some kind of statement that becomes convoluted for you to process the logic behind, the right mechanism for you to avoid confusion is create a flowchart. This flowchart is supposed to express the flow of computation of something like this. So there's nothing deeper. deep here. But if you don't map this to the flowchart in the proper way, then the logic that you are interpreting you have may be completely wrong. This is (Indiscernible) . What is the flowchart? In this condition, if this condition is true, whatever is intended after that condition, and this could be in this case true statements. If this condition is true, these true statements are executed. That means if this condition is false, something else is going to be executed.
Question to you. What is the difference between that statement and do the other two statements? What is the difference between that statement with the analysis and the other true statements which read exactly the same . This is exactly the same as that. This is exactly the same as that. PARENT Parentheses this is identical to this and this is identical to this but the three of them are quite different. Answer.
>> STUDENT: Depending on the condition. Statement you get a different (Indiscernible) or input. (Indiscernible) Is outside the conditional statement.
>> INSTRUCTOR: Very good. If this condition is true, what is the value of RES in this statement? What is the value of this RES in this statement? Let me refroze the question. What is RES here? The assignment statement says this whole thing is going to be assigned to that. What is RES? (Indiscernible) Disregarding this but I don't think you guys are understanding this. What is the value of RES here? Is it true, is it false, is it a string, is it a number? What is it? It's a string. What is a string? What is the value of RES?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: It's not X greater than one.
>> STUDENT: True .
>> INSTRUCTOR: No, it's not true.
>> STUDENT: (Indiscernible) Conditional statement.
>> INSTRUCTOR: What is the value of RES right there? Is it.[word?]? Which is three?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: X-ray greater than Y. What is the difference then? With this? What is RES in that case?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: Another string which is.[word?] X (Indiscernible) why this makes sense in this piece of Python code? You as a human when you read this, why this make sense? What is this saying? Look at the if statement. What is this? What is this? What is the condition here? Without analysis . What are you testing?
>> STUDENT: X greater than Y.
>> INSTRUCTOR: If X is greater than Y. What is the value of that condition? I'm getting into this because when you are writing code, if this is not completely clear-cut you'll be writing a bunch of (Indiscernible). What is the value of X greater than Y? What is the value its true or

false. We don't know if it's true or false. That depends . At that point in the code, what is the value of X and what is the value of Y? If I came to it, happens to be that the value of X is greater than the value of Y . That condition is true. Then we are setting RES to be discussed . Listen. I can put here whatever I please . I can put here CS210 is awesome. I can put here the next president of the United States is going to be blah, blah, blah . I can put here whatever I want. This is telling me that you want to print this if this is the case. The point about this exercise is it's important to separate what is the condition that you are evaluating from the action you're taking. This condition is really independent of this. Okay. What about this one now? I hope you understand now this. You understand this. This is (Indiscernible) statement that means that condition was not true so you come here blah, blah, blah. When in this piece of code when you would execute this statement here, when this statement would be executed in this piece of code? ? Only one? 2, 3? Yeah?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: I would like to have a very short answer. When is that piece of code reaching that statement?
>> STUDENT: At the very end of the.[word?].
>> INSTRUCTOR: This piece of code. If it executes (Indiscernible) statement. Dependent of whatever happens between X and Y. If X is greater than Y, then you can go to that part and then after that if x-ray is greater than Y, this will happen. RES will get this string. This will get printed but after this is done the code will go here. If this condition is full, the code will go here and after this is done, this will be printed. This is what is the function is telling you. That's the valid way to interpret this. This is clean, clear. In (Indiscernible) you have a condition if statement. It has two possibilities. Either it's true or it's false. If it's true, this is happening. If it's false, this is happening. And no matter what at the end, this is happening. Now I would ask you then the last question. What will be printed here? Where that mouse is, what will be printed? Explain. Same person? What happened to the other guys? Do you want to give it a try?
>> STUDENT: (Away From Mic).
>> INSTRUCTOR: If the statement is true, it will print X greater than Y. But if the statement is false, It will print the other one. Raise your hand if this is plain, clear. Yeah? Good? Okay. Now happens to be that when you start concatenating these things, the logic may become cumbersome . . there's this so-called.[word?] Operator. It's just a name . Very sophisticated name. If you have a piece of code like this if X is greater than Y, RES would be Joe Biden. (Indiscernible) Would be not Joe Biden. If you have something like this, this can be replaced by one line. Don't take this lightly because the first piece, the expression when condition is telling you that this is the action . This is the action that you would like to take. I also want to take action if the condition is true. I would like to take a different answer. You can look at these examples. Let's look at something more interesting. This is called the ELI parameters statement. Now you understood a we tried to explain in the previous statement. Now look at what this is saying. It's saying if the condition is true, do this.
ELIF Says if that condition is not true, you are going to do this. Again ELIF is saying if this condition is not true, then you're going to do this. And after that then some other ELIF is

saying if this condition is not true and if this condition is not true, then you're going to do that. This allows you now to start creating very complex.[word?] Conditions. Anybody can tell may the answer here? two is for X. (Indiscernible) if X is less than Y which happens to be the case here then this is print. But X is not why so you would print this statement. If that is not the case, you will print the Y instead of Z. If that is the case, then you do this, and if that is not the case, then you do this. I think that you will have to play with this. I may explain hours going over this. Combining these things, you can solve (Indiscernible) you want . These are very simple things, but you have to be able to go to the ZY book practice test. This is very useful for things like this. If those scores greater than 90, the greatest A. If that's not the case, if it's greater than 80, then it's B. This is equivalent to saying if the score is between 80 and 90. Instead of saying the scores between 80 and 90, this is saying if it's greater than 90 it's A . If not, you apply this condition. If these two don't happen, then you apply this condition. If this is not happening, then you apply this condition.

This Is very useful and this will happen if none of these things happen. So these are applications about finding for the point on the brain if it's quadrant one or quadrant two or quadrant three if it's testing the condition. Now something more interesting. In order to facilitate when you have a whole bunch of (Indiscernible) items has created something which is called the match case. Think of this as selecting items in the store. You will see what I mean by this. All of these syntactical descriptions are semantics . That is precisely defined by.[word?]. What this is saying is if you like MATCH you are playing a matching game . The matching game says this is what I want. I want a particular expression. Here you can write whatever expression you want. You do the column and here you have a collection of cases. This is saying by -- my matching game is if my expression happens to match pattern one execute this piece of code. If my expression happens to match pattern two, I execute this piece of code etc. And at the end, the last case, none of those happen and that's a default. This is the right way to think about these things. You can think of this as a store, and in this store, they design the [word?] expression and when you are coming into the store, your task is to match this expression with these patterns here. Each of these boxes is labelled by a pattern. This is pattern one, pattern two, pattern three and when you come here, you are going to take the expression, match it with whatever pattern you want and based on that, you're going to execute the different pieces of code.

If None of this happen, then there's something on the back which is a default route. This is exactly the meaning of the match statement. These things are very useful in maintaining systems because in general, when you are checking processes in a system or (Indiscernible) etc., you have a certain state of whatever process you're looking at . If it happens to be that the system has decided that certain numbers mean something, that's what the systems do. They check the status of some process depending on what you are doing and you're going to check that status versus different cases. If this is 400, that's usually a batch request. If it's 404, that's usually something that is not found. 418 etc. These are real, clear messages that you get when you are trying to look at the status of processes in a system and the implementation is very simple and similar to match cases. This is usually what happens when you are requesting some particular website. It's an HTTP request so-called and

sometimes they give you some numbers. That number, what the heck that means? It has different meanings so this is how it's implemented with the match case. Very simple. These very simplistic descriptions here is very powerful because you can actually make your expressions as complicated as you want. For example, this one is one case in which you are saying if it's pattern one, or pattern two, I want to execute this code. If it's pattern three or pattern for, you want to execute this code. And this is usually what miniapplications do . Is pattern detection. Different collections of patterns and based on the pattern that is detected, different pieces of code are executed. Another important thing to understand here is that this matching game between an expression and the cases that you have, this matching game can be applied to very complicated subjects. You can match again a tumble versus a list order versus a dictionary and there are previous rules for that. Let's try to look at these examples. And usually by the way, usually in code, you define the expression that you are checking and the cases. You define that. You put in a function definition and then you can use it in your code. In this particular case, your matches are going to be defined in this here and you can express what you want to express. If you are giving me a pair of values (Indiscernible) the point is exactly 00. That's the only thing.
You may specify a case in which the first parameter is zero but the other is undefined. That means that when the point is zero and something is zero, then that is on the y-axis. If your methods of processing, checking if these (Indiscernible) conditions are computer graphics and when you see these beautiful graphics, each of these pieces is being checked for some condition. Usually these things are referred to as pattern matching, again, whenever you see in the case definition something like this, that means that you are allowing either this pattern or this pattern. In this particular case, if you are dealing with data that's two dimensions versus three dimensions, if it's two dimensions, the (Indiscernible) is 00. If it's (Indiscernible) you can use is 000. If you can combine this this gives you a complicated collection of patterns that you can describe just by playing this game. Forget this pattern. Forget the other one so it allows you to specify a very complicated pattern. Now there's something called safe programming. This is telling the programmer any errors that you can somehow anticipate come make it explicit in your problem that this is an error that you are anticipating. That will make you (Indiscernible) because it's a lot simpler.
People Have invented surnames for this that are called (Indiscernible). Let me give you an example. This is called a guard condition. This is something you can specify. This could happen. I'm going to tell you when that happens. For example, in this particular individual has a name and this is saying it could be that this name start with two letters AZ or it could be that the name contains he letters HAOYAG and if that's the case you want to write out that this particular person is (Indiscernible). Now you can specify in this particular name and particular age and look at here this is very interesting that this expression now has a conditional statement. Applied to this. This is AVH greater than 18, that is the case that you analyze. You're looking at a particular pair . We started today speaking of a dictionary. In a dictionary, there is a name . A corresponding name. And if that age is greater than 18, that's the condition that you're checking. And you want to say I want to welcome that particular individual. In this particular case, there is agreement that the age is under 18 so

this is not a case and in that case that means this age does not satisfy that condition. Therefore you're good to go.
The Flow of this may become complicated if you are not careful in how you write it down. The point that this is illustrating is that you can make these expressions complicated . This is checking pairs in a dictionary for certain conditions. And then you take action. There are more examples of that. Your midterm is coming in three weeks and the best way to prepare for the midterm is to look at these questions and these questions (Indiscernible) usually they are linked to different portions of the CY books and these are usually very short questions that you have there . This point notice that there are certain things here that are essential and we try. For example, if you are going to look at here the same thing and say what the heck that means. If you don't remember, you can go to the CY code. They are still literals. What the heck that means? At six . How you use the functions to find (Indiscernible). How you index a string . All these things, they may be in the midterm inside questions. How you create lists. How you create tuples. How you create sets. How you create dictionaries. How you convert between types. What are the most important things about functions. How you code a function and define a function. What do you do when you have multiple parameters? You find something like this that we have not explicitly discussed in class, ignore it.
We'll take some time and we'll get to that at some point in the class. Mathematical functions. One of the most important mathematical functions that we have? Branching. Branching is this notion of if conditional statements . If else. Else if, what we discussed today. Logical operators . How to put them together in expressions. Nesting if else statements. These are the expected things for everybody in the class to make it to the midterm. Now there is the challenge activities that have been assigned for homework so you are able to do the challenge activities or you have done it. I don't think you should expect any difficulties in the midterm. There are supplemental questions. You want to be challenged more, these are usually a little bit more complicated, but you are not required to do those. These are not required but you want to look at some questions that you may think tricky questions, little bit more tricky.
These Advanced questions are the most tricky ones. If you see participation, that's not supposed to be tricky. The types of questions will be similar to the participation questions you see here. Assistance with the class are posted office hours so you should contact them if you need help. Don't wait to wait because material is going to accumulate within the next four lectures so it's good to started looking at this. Today, this evening we will be posting the mini project and you should be in touch with the members of your group . This is supposed to be a group project . If it happens to be in your group, there's a person who is a free rider, we don't like free riders. Everyone should work . We have prepared this mini project in such a way that all you have to do is you have to look at the template and that template give you the most important parts about the codon you need to develop and that template is a flowchart and in each of the boxes of the flowchart, there is an indication of what is it that you need to do and that piece of code . My suggestion is that you guys should look at this and there's a box there that is painted red . The box that is painted red

is the most difficult part of the call. There's nothing extraordinarily difficult, but that is the most difficult part and that way you can maybe divide among the members of the group. This Is a group mini project. Why is that? Nobody today, nobody writes code in companies individually. Those times have gone. All companies expect you to be able to work in groups. That's how companies go. I think it's a good time to stop.
(End Of Session)